Algebraic cryptanalysis and contributions to post-quantum cryptography based on error-correcting codes in the rank-metric

Maxime Bros

Laboratoire XLIM : **Axe Mathématiques & Sécurité de l'information**

École doctorale : **Sciences et Ingenierie - Xlim**, ED 653

Algebraic Cryptanalysis and Contributions to Post-Quantum Cryptography based on Error-Correcting Codes in the Rank-metric

Cryptanalyse algébrique et contributions à la cryptographie post-quantique basée sur les codes correcteurs d'erreurs en métrique rang

Thèse de Doctorat

présentée et soutenue publiquement le 08 décembre 2022 par

Maxime Bros

pour obtenir le grade de

Docteur de l'Université de Limoges

Devant le jury composé de :

Rapporteurs	
Alain Couvreur	Directeur de Recherche, *Inria, Saclay*
Ayoub Otmani	Professeur des Universités, *Université de Rouen, Rouen*
Examinateurs	
Magali Bardet	Maître de Conférences, *Université de Rouen, Rouen*
Olivier Blazy	Professeur, *École Polytechnique, Palaiseau*
Pierre-Jean Spaenlehauer	Chargé de Recherche, *Inria, Nancy*
Jean-Pierre Tillich	Directeur de Recherche, *Inria, Paris*
Directeurs de thèse	
Philippe Gaborit	Professeur des Universités, *Université de Limoges, Limoges*
Vincent Neiger	Maître de Conférences, *Sorbonne Université, Paris*

"Give me a nice system of algebraic equations, and a large enough random-access memory to place it, and I shall cryptanalyze the world."

Archimades, 3rd century BC.

Acknowledgments ∗ ∗ ∗ Remerciements

Numerous people have supported me over the course of my PhD. While I have tried to name everyone, please forgive me if I have forgotten to mention you below; I am extremely grateful for your encouragement.

∗ ∗ ∗

Je tiens tout d'abord à remercier les rapporteurs de ma thèse, Alain Couvreur et Ayoub Otmani, pour leurs minutieuses relectures et leurs précieux conseils.

Je remercie également tous les membres du jury, Alain Couvreur, Ayoub Otmani, Magali Bardet, Olivier Blazy, Pierre-Jean Spaenlehauer, Jean-Pierre Tillich, Philippe Gaborit, Vincent Neiger, d'avoir accepté d'être présents pour ce jour si important et stressant pour moi, c'est un grand honneur que vous me faites !

Philippe et Vincent, merci d'avoir accepté de m'encadrer et de m'avoir fait découvrir le merveilleux monde de la recherche et de l'enseignement supérieur. J'ai toujours voulu aller vers une carrière académique ; et après cette expérience mon choix est encore plus certain qu'il ne l'était déjà, et c'est grâce à vous.

Merci pour votre patience, malgré mes interrogations et gaffes, merci de m'avoir poussé, même si c'était parfois un peu rudement, à toujours donner le meilleur de moi-même.

Merci pour votre confiance et vos encouragements, cette thèse fût une belle et forte expérience pour moi, et j'ai appris tellement de choses à vos côtés.

En conclusion, je vous remercie du fond du cœur pour ces 3 années !

∗ ∗ ∗

I would like to thank all of my co-authors; it has been an honor to work with you, and I hope this is only the beginning!

In particular, I would like to thank Ray Perlner and Daniel Smith-Tone for all the discussions we had by email. Furthermore, I am especially grateful for the opportunity offered to me by Ray to work at NIST starting in January 2023. I am very excited and look forward to coming to the US to collaborate with you and your colleagues.

∗ ∗ ∗

Magali Bardet, merci pour toutes les explications et précieux conseils que tu m'as donnés et pour ta relecture attentive de ma thèse.

Merci également de m'avoir permis de faire une visite de recherche au sein du LITIS à Rouen. Bien que ces quelques jours soient passés très vite, ils m'ont appris beaucoup de choses et j'ai beaucoup apprécié de travailler avec toi.

J'ai énormément aimé pouvoir me rendre à presque tous (modulo la sombre période du Covid) les GT Code-Based Cryptography à l'Inria Paris.

Les rencontres et discussions que j'ai pu avoir là-bas ont été extrêmement enrichissantes, les trajets fatiguants[1] depuis Limoges à 6h du mat' en valaient donc vraiment la peine ! Ces journées mensuelles ont été l'occasion d'apprendre à connaître des gens formidables, entre autres Jean-Pierre Tillich[2], Magali Bardet, Manon Bertin, Alain Couvreur, Thomas Debris-Alazard, Maxime Bombar, Matthieu Lequesne et Thibauld Feneuil.

Pendant 3 ans, j'ai eu l'immense honneur de pouvoir enseigner les mathématiques et l'informatique et de travailler dans un environnement extrêmement agréable et convivial. Je remercie donc tous mes collègues limougeauds, ou limougeauds de passage au moment des faits : Joëlle, Jérémy, Léo (pour m'avoir accueilli quand je n'étais qu'un petit stagiaire), Laura, Victor (pour avoir été de supers collègues de bureau), Adrien, Cyprien, Guillaume Goy, Hamza, Mathieu, Neals, Nicolas, Théo, Yann (pour tous les moments sympas passés entre (post-)doctorants), Guillaume Gilet, Maxime M. (pour m'avoir fait découvrir de délicieuses bières chez Orge et Houblon), Benoît, Cristina, Cyrille, Christophe, Damien, Emmanuel, François, Loïc, Mercedes, Moulay, Olivier B., Olivier R., Pierre, Pierre-François, Stéphane, Thibault, Thomas, Tristan (pour les nombreuses discussions, pauses-café et repas que nous avons partagés).

Les indénombrables pauses-café que nous avons passées ensembles sont ancrées dans ma mémoire et m'ont permis de ne pas devenir complètement dingue quand je travaillais trop... Je réalise néanmoins que *indénombrables* n'est pas précis, donc soyons précis :

$$\underbrace{3}_{\text{années}} \times \Big(\underbrace{52.14}_{\text{semaines/an}} \times \underbrace{5}_{\text{jours de travail/semaine}} - \underbrace{35}_{\text{vacances}} \Big) \times \underbrace{3}_{\text{pauses-café/jour}} . \tag{1}$$

[1] D'autant plus que j'ai une incapacité totale à dormir dans le train !

[2] Même si tu as essayé de me tuer avec ton choix d'épices très audacieux chez Chongqing, et que tu m'as détruit au baby-foot.

L'Équation (1) donne une première majoration puisque la période Covid n'est pas prise en compte, ni le fait que parfois, bien que très rarement, nous avions seulement une ou deux pauses café par jour. Puisque l'Équation (1) donne un nombre de pauses-café délirant, je préfère l'exprimer en logarithme binaire : il semble plus raisonnable de dire que j'ai consommé, au plus, environ 11 bits de café pendant ma thèse.

Dire qu'une thèse est un long fleuve tranquille serait faux, mais grâce à vous qui avez su m'écouter, me comprendre, et me remonter le moral lorsque le besoin s'en est fait ressentir, j'adresse de très chaleureux remerciements à Alain, Cristina, Cyrille, Damien, Emmanuel, Jean-Pierre, Olivier B., Olivier R., Philippe, Victor, Vincent. Votre soutien m'a profondément touché, je ne serais pas ici sans vous, et ça je ne l'oublierai jamais !

Durant l'été 2022, j'ai eu le privilège de pouvoir encadrer deux stagiaires : Frédéric Canaud et Romaric Neveu. Ces deux mois ont été très agréables pour moi (malgré la chaleur caniculaire) et vous ne pouvez pas imaginer le bonheur que j'ai ressenti quand vous m'avez dit vouloir continuer dans la recherche en partie grâce à moi.

Olivier Gimenez et Maxime Remaud, je vous ai connus durant le Master Maths CRYPTIS, promotion 2017-2019[3], et quel plaisir que vous ayez également continué en thèse. Vous êtes des amis précieux et je vous remercie pour tous les moments de bonheur passés ensembles.

Je tiens également à remercier Brigitte, Debora, Françoise, Henri, Julie, Monique, Sophie, vous avez toujours été là pour m'aider à travailler dans d'excellentes conditions ! Merci à Patricia et Sandrine, pour m'avoir permis d'avoir un bureau impeccable, et surtout pour toutes nos discussions des lundi matins 8h30, qui, comme vos sourires, vont me manquer.

* * *

Last but not least[4], I would like to thank my parents for their infinite love and support. I owe you absolutely everything! This thesis is dedicated to both of you, the best parents I could ever wish for.

[3]Meilleure promotion de tous les temps au passage.

[4]This is one of my favorite expressions, as you will see if you are brave enough to read the rest of this document.

Katia, I thank God every single day for giving us the chance to meet in Limoges. What are the odds? You... from Iowa, in the middle of nowhere in the US. Me... from Haut-Doubs, in the middle of nowhere in France. You are my everything, and you made me the happiest man in the world when we got married on November 5th, 2020. It is an honor to be your husband and to have you by my side.

I would also like to thank my US family for always encouraging me along the way.

I suspect a lot of people are reading these acknowledgments just to find a mention of Cauchy. Thanks to Olivier B., you are now an international crypto star[5]! I cannot say that I appreciated you waking us up in the middle of the night. However, your playful and spirited nature would always make up for the stressful moments of these past few years.

[5] https://twitter.com/gloupin/status/1472251495005437953?s=20&t=unWyK4pILXQWd5GAvVZU8g

To my parents,

To my wife.

Contents

Notation and Acronyms

All along this document, we use the following notation:

- q is a power of a prime p, $\mathbb{F}_q$ is the finite field with q elements, and $\mathbb{F}_{q^m}$ its extension of degree m.
- for a field $\mathbb{F}$, we denote by $\mathbb{F}^\times$ the multiplicative group of non-zero elements of $\mathbb{F}$.
- $\alpha \in \mathbb{F}_{q^m}$ is a primitive element, and

  $$\beta := (1, \alpha, \ldots, \alpha^{m-1}) \in \mathbb{F}_{q^m}^m$$

  is a basis of $\mathbb{F}_{q^m}$ seen as an $\mathbb{F}_q$-vector space.
- We denote by $\mathbb{K}$ a field, and $\overline{\mathbb{K}}$ its algebraic closure.
- For a field $\mathbb{K}$, we denote by $\mathbb{K}[x_1, x_2, \ldots, x_n]$ the ring of multivariate polynomials in n variables overs $\mathbb{K}$.
- Matrices and vectors are written in boldface font: $\boldsymbol{M}$, $\boldsymbol{v}$. By extension, we identify a point $\boldsymbol{p}$ to its position vector, thus it is also written in boldface font.
- The general linear group of non-singular $n \times n$ matrices with entries in $\mathbb{F}_q$ is denoted $\mathrm{GL}_n(\mathbb{F}_q)$.
- For a positive integer n, $\mathbf{0}^n$ refers to the zero vector of length n. However, when there is no ambiguity, we might just denote it 0.
- The identity matrix of size $n \times n$ is denoted $\boldsymbol{I}_n$.
- The transpose of a matrix $\boldsymbol{M}$ is denoted by $\boldsymbol{M}^\intercal$.
- For a given ring $\mathcal{R}$, the set of matrices with n rows, m columns, and entries in $\mathcal{R}$, is denoted by $\mathcal{R}^{n \times m}$.

- $\{1, 2, \ldots, n\}$ and $\{1..n\}$ stand for the set of integers from 1 to n, we use them indifferently.
- For a subset $I \subset \{1..n\}$, $\#I$ stands for the number of elements in I.
- For two subsets $I \subset \{1..n\}$ and $J \subset \{1..m\}$, $\boldsymbol{M}_{I,J}$ denotes the submatrix of $\boldsymbol{M}$ formed by its rows (resp. columns) with indices in I (resp. J).
- For an $m \times n$ matrix $\boldsymbol{M}$, we use the shorthand notation

$$\boldsymbol{M}_{*,J} = \boldsymbol{M}_{\{1\ldots m\},J}, \boldsymbol{M}_{I,*} = \boldsymbol{M}_{I,\{1\ldots n\}}.$$

- $\boldsymbol{M}_{i,j}$ denotes the entry in $\boldsymbol{M}$ at row i and column j.
- $|\boldsymbol{M}|$ denotes the determinant of a matrix $\boldsymbol{M}$, $|\boldsymbol{M}|_{I,J}$ is the determinant of the submatrix $\boldsymbol{M}_{I,J}$.
- We extend the aforementioned shorthand notation to determinant:

$$|\boldsymbol{M}|_{*,J} = |\boldsymbol{M}_{*,J}|\,, |\boldsymbol{M}|_{I,*} = |\boldsymbol{M}_{I,*}|\,.$$

- We recall the definition and notation for a Gaussian coefficient over q:

$$\begin{bmatrix} m \\ t \end{bmatrix}_q := \prod_{i=0}^{t-1} \frac{q^m - q^i}{q^t - q^i}.$$

- For a set A, the notation $a \xleftarrow{\$} A$ means that the element $a \in A$ is picked uniformly at random in A.
- We say that $\varepsilon(x) \geq 0$ is a negligible function of x if for all positive polynomial p:

$$\exists N_p \in \mathbb{N},\ x > N_p \quad \Longrightarrow \quad \varepsilon(x) < \frac{1}{p(x)}.$$

  For instance, 2^{-x} is a negligible function of x. For the sake of clarity, we might omit the parameter x when there is no ambiguity; indeed, we just say that ε is negligible.
- In this document, we when say that an event happens with *overwhelming probability*, it means that it happens with probability $1 - \varepsilon$, where ε is negligible.

- When the complexity of an algorithm, or an attack, is said to be of n bits, it means that the attacker would have to perform at least 2^n operations (usually elementary operations in the field $\mathbb{F}_q$). In other words, saying that an attack has a cost of 100 bits is just a simpler way of saying that at least 2^{100} operations are required to perform the attack.
- 1^λ is a security parameter in unary which will parameterize in practice the key lengths, group sizes, and complexity for our cryptographic primitives. It affects the success probability of adversaries to win specific security games.
- For a bit string s, we denote its length by $\text{len}(s)$.

Here is a list of the main acronyms used in this document:

Acronym	Refers to
RD	Rank Decoding
NHRD	Non-Homogeneous **RD**
RSL	Rank Support Learning
NHRSL	Non-Homogeneous **RSL**
DRD	Decisional **RD**
DRSL	Decisional **RSL**
DNHRSL	Decisional **NHRSL**
IRD	Ideal **RD**
IRSL	Ideal **RSL**
NHIRSL	Ideal **NHRSL**
DIRD	Decisional **IRD**
DNHIRSL	Decisional **NHIRSL**
PSSI	Product Spaces Subspaces Indistinguishability
IND-CPA	Indistinguishability under Chosen-Plaintext Attack
EUF-CMA	Existential Unforgeability under Chosen-Message Attack
DFR	Decoding *or* Decryption Failure Rate *(depending on the context)*
PPT	Probabilistic Polynomial-Time
KEM	Key Encapsulation Mechanism
HQC	Hamming Quasi-Cyclic
RQC	Rank Quasi-Cyclic
Multi-RQC-AG	Multi-syndrome RQC Augmented Gabidulin
Multi-UR-AG	Multi-syndrome Unstructured Rank Augmented Gabidulin
NH-Multi-RQC-AG	Multi-RQC-AG using Non-Homogeneous errors
NH-Multi-UR-AG	Multi-UR-AG using Non-Homogeneous errors
NIST	National Institute of Standards and Technology
NIST PQSP	NIST Post-Quantum Standardization Process

Chapter 1

Introduction

Contents

1.1 Context

Public key cryptography is a primordial tool used to render digital communications secure. All currently used public key cryptosystems share a property: they rely on two distinct but connected problems: the factorization of integers and the discrete logarithm problems.

The first public key cryptosystem, namely a key exchange algorithm, was invented in 1976 by Diffie and Hellman [DH76], it relies on the discrete logarithm problem. Shortly after, the first public key encryption and signature scheme was invented [RSA78]. It is called RSA after the names of its inventors: Rivest, Shamir, Adleman, and it relies on the hardness of factoring large integers.

1.1.1 Quantum threat

In 1994, Shor described a quantum algorithm [Sho99] that solves both of these problems efficiently. This means that, with a powerful and stable enough quantum computer, Shor's breakthrough algorithm would enable one to break all current cryptosystems.

In the last two decades, there have been a lot of developments in building quantum computers; this is why, in 2017, NIST (National Institute of Standards and Technology) launched its call for cryptosystems that would resist both classical and quantum computers [C+16, AAAS+19]. Such cryptosystems are said to be *quantum-resistant* or *post-quantum*.

The goal of this call for proposals, referred to as the NIST Post-Quantum (Cryptography) Standardization Process, is to select a list of post-quantum cryptosystems, namely encryption and signature schemes, for standardization.

1.1.2 Post-Quantum Cryptography

Surprisingly, the first post-quantum cryptosystem was invented at the same time as RSA, even if it was not said to be *post-quantum* at that time. It is McEliece's cryptosystem [McE78], which was published in 1978.

McEliece's cryptosystem belongs to the so-called *code-based cryptography* since its security relies on the hardness of decoding error correcting codes.

While **code-based cryptography** was the first kind of post-quantum cryptography, it is not the only one. There are currently four other main categories in post-quantum cryptography:

Hash-based cryptography started in 1979 with Lamport's one time signature [Lam79], which evolved with the use of Merkle trees [Mer89].

Multivariate-based cryptography started in 1988 with the C* cryptosystem by Matsumoto and Imai [MI88]. C* has been broken in 1995 by Patarin [Pat95] who later fixed it and created HFE [Pat96].

Lattice-based cryptography started in 1996 with [Ajt96] and the NTRU cryptosystem [HPS98]. The Learning With Error (LWE) problem was later introduced in the seminal paper by Regev [Reg05].

Isogeny-based cryptography started in 2011 with [JF11] by Jao and De Feo using results from Couveignes [Cou06], Rostovtsev and Stolbunov [RS06].

The NIST Post-Quantum Standardization Process stimulated the community to develop new designs and attacks. For instance, very recently some Third Round Finalists and Alternate candidates were attacked, namely Rainbow by Beullens [Beu22], and SIDH/SIKE by Castryck and Decru, Maino and Martindale [CD22, MM22].

In July 2022, NIST released the results of the Third Round [AAC+22], that is to say, the list of candidates to be standardized and the ones advancing to the Fourth round; this is summarized in Table 1.1.

Cryptosystem	**Type**	**Category**	**Decision**
CRYSTALS-Kyber	KEM	Lattice	To be standardized
CRYSTALS-Dilithium	Signature	Lattice	To be standardized
Falcon	Signature	Lattice	To be standardized
SPHINCS+	Signature	Hash	To be standardized
BIKE	KEM	Code	Moving to 4^{th} Round
Classic McEliece	KEM	Code	Moving to 4^{th} Round
HQC	KEM	Code	Moving to 4^{th} Round
SIKE	KEM	Isogeny	Moving to 4^{th} Round

Table 1.1: Third Round results announced by NIST in July 2022. KEM stands for "Key Encapsulation Mechanism".

While the attacks against Rainbow and SIKE do not necessarily mean the end of multivariate or isogeny-based cryptography, they alter the confidence in these approaches. This strengthens the interest for code and lattice-based cryptography.

1.1.3 Rank-based Cryptography

The rank metric was introduced in 1978 by Delsarte [Del78] for matrix codes. In 1985, Gabidulin introduced the first efficiently decodable rank metric $\mathbb{F}_{q^m}$-linear codes, namely Gabidulin codes [Gab85].

In a cryptographic context and for a given security level, rank metric codes have the advantage to usually enable one to have shorter public keys and ciphertexts/signatures compared to classical Hamming metric codes.

This is why, in the last decade, rank metric code-based cryptography, often shortened in *rank-based cryptography*, has evolved to become a real alternative to traditional code-based cryptography using the Hamming metric.

1.1.3.1 Encryption schemes

The original scheme based on rank metric was the GPT cryptosystem, named after its inventors: Gabidulin, Paramonov, and Tretjakov [GPT91].

This is an adaptation of the McEliece scheme in a rank metric context; in other words, in the GPT cryptosystem, Gabidulin codes, which can be seen as rank metric analogs of Reed-Solomon codes, are the masked codes.

However, the strong algebraic structure of these codes was successfully exploited for attacking the original GPT cryptosystem and its variants with the Overbeck attack [Ove05] (see [OTKN18] for the latest developments).

This situation is similar to the Hamming metric setting where most of McEliece cryptosystems based on variants of Reed-Solomon codes have been broken.

Besides McEliece-like schemes where a secret code is masked using permutation, it is possible to generalize the approach by considering public key matrices with a trapdoor.

Examples of such an approach are NTRU [HPS98], and MDPC (Moderate Density Parity-Check) [MTSB13] cryptosystems where the masking consists in knowing a very small weight vector of the given public matrix.

Such an approach was adapted to the rank metric through the introduction of LRPC (Low Rank Parity-Check) codes [GMRZ13], a rank metric analog of MDPC. These codes lead to the ROLLO cryptosystem [ABD+19].

The security of these cryptosystems relies on the general rank decoding problem together with the computational indistinguishability of the public key, *i.e.*, a public matrix, from a random one.

This enables one to obtain very efficient schemes, however this comes at the price of a matrix inversion to compute the public key matrix.

It is worth noticing that Loidreau's encryption scheme [Loi17], which uses Gabidulin matrices in a McEliece context, masked with homogeneous matrices of low rank weight, seems to resist to structural attacks as long as the masking matrix has a sufficiently large rank.

Another approach, pioneered in 2003 by Aleknovich [Ale03], permits to rely solely on random instances of the decoding problem without any masking of a public key. However, such an approach is strongly inefficient in practice.

A few years later, a more optimized approach was proposed with the HQC (Hamming Quasi-Cyclic) scheme [AAB+21b], relying on Quasi-Cyclic codes.

It has been adapted to the rank metric with the RQC (Rank Quasi-Cyclic) scheme [AAB+20].

In these schemes, two types of codes are used: first, a random double circulant code permits to ensure the security of the scheme, second, a public code permits to decode/decrypt the ciphertext.

In RQC scheme, Gabidulin codes are used as public decryption codes.

Besides RQC, some other variations were proposed in [GHPT17, Wan19, GGH+20]. The main advantage of the RQC cryptosystem is the fact that its security reduces to decoding random ideal codes whereas LRPC cryptosystems require an additional indistinguishability assumption; however, this advantage comes at a price since parameters are larger for RQC than for LRPC.

The RQC scheme was proposed to the NIST Post-Quantum Standardization Process with very competitive parameters.

In 2020, algebraic attacks [BBB+20, BBC+20], published during the standardization process, had a dreadful impact on RQC parameters.

In order to limit the impact of these algebraic attacks, that is to say, not to increase too much RQC parameters, non-homogeneous errors were introduced in [AAB+20].

The idea of non-homogeneous errors is to consider errors in three parts of length n such that the error weight is the same for the first two sets, but larger for the third one. Such an approach permits to limit the impact of the security reduction of RQC to decoding random $[3n, n]$ codes rather than $[2n, n]$ codes in LRPC cryptosystems.

The notion of non-homogeneous error led to the introduction of the Non Homogeneous Rank Decoding problem (**NHRD**).

It is worth noticing that for both LRPC and RQC cryptosystems, the weight of the error to attack is structurally $\mathcal{O}(\sqrt{n})$, where n is the length of the code, which is a range of parameters for which algebraic attacks turn out to be very efficient.

Besides the classical **RD** (Rank Decoding) problem, the **RSL** (Rank Support Learning) problem, which consists in having N syndromes whose associated errors share the same support, was introduced in [GHPT17] to construct a RankPKE scheme, and used later in [Wan19].

This problem, which generalizes **RD**, is meaningful to give more freedom while building cryptosystems; for instance, it has been recently used to improve the LRPC scheme [AAD+22].

RSL permits, in particular, to increase the weight of the error to decode from $\mathcal{O}(\sqrt{n})$ to a weight closer to the Rank Gilbert-Varshamov (RGV) bound; this is primordial since for that type of parameters, *i.e.*, close to the RGV bound, algebraic attacks become relatively less efficient than classical combinatorial attacks.

1.1.3.2 Signature schemes

While two rank-metric encryption schemes, ROLLO [ABD+19] and RQC [AAB+20], were selected for the Second Round of the NIST Post-Quantum Standardization Process, designing rank-metric signature schemes is a more challenging task.

Code-based signature schemes in general, and rank metric schemes in particular, can essentially be split in two categories: the hash-and-sign schemes and the proof of knowledge ones.

For building an hash-and-sign signature scheme, one needs to be able to find a low rank error vector associated to a syndrome.

Ranksign [GRSZ14b] was built using this technique together with LRPC codes. However, it has been shown in [DT18] that it is possible to recover the secret LRPC matrix from the public key.

Designing proof of knowledge signature schemes can be done in two ways.

The first one consists in turning a zero-knowledge authentication scheme into a signature scheme using the Fiat-Shamir transformation (also called the Fiat-Shamir heuristic) [FS87]. This approach usually leads to schemes with large signature sizes since the authentication protocol needs to be repeated multiple times, depending on the soundness of the underlying authentication scheme, in order to reach an arbitrary high security level.

Instead of using zero-knowledge authentication schemes, one can build upon the work of Lyubashevsky [Lyu09], which adapts the Schnorr signature scheme [Sch91] to the Euclidian metric.

In the rank metric setting, this idea gave rise to the Durandal [ABG+19] signature scheme, and to the Rank Preserving Signature (RPS) scheme [LT20b], broken in [ABG21].

1.2 Contributions

Our contributions are divided in three parts: cryptanalysis results (Chapter 3), improvements of cryptosystems (Chapter 4), and the introduction of a new problem (Chapter 5). These contributions are summarized in Table 1.2 on page 27.

1.2.1 Cryptanalysis results

Rank-based cryptography is a subfield of code-based cryptography where one uses the rank metric instead of the traditional Hamming metric.

The Rank Decoding (**RD**) and the **MinRank** problems are at the core, respectively, of rank-based cryptography and multivariate-based cryptography. Moreover, **RD** reduces to **MinRank**, thus the **MinRank** problem is important in rank-based cryptography as well.

MinRank. In Section 3.1, we propose a new algebraic attack against the **MinRank** problem.

Unlike previous approaches, such as Kipnis-Shamir's or minors modeling [KS99, FSEDS13, FLdVP08, VBC+19], our system of algebraic equations uses linearization techniques to replace maximal minors.

More specifically, we replace each determinant of degree r with roughly $r!$ terms by a single variable that we denote c_T. By doing so, our system of algebraic equations, *i.e.*, our modeling, is far smaller compared to previous approaches; which leads to better complexities.

This modeling, namely the SupportMinors modeling, has been introduced in [BBC+20], and further improved in [BBB+22].

RD. For a long time, combinatorial attacks were considered to be the only efficient attacks to solve **RD**, see for instance [GPT91, GRS16, AGHT18, AMBD+18]. Roughly, combinatorial attacks rely on picking elements at random before solving a linear system.

For instance, in the case of **RD**, the guessing process concerns vector spaces: one picks vector spaces of a given dimension and tries to solve a linear system. This process eventually ends and one gets the solution when the picked vector space contains the support of the solution error.

In Section 3.2, we present our results [BBB+20, BBC+20, BBB+22]: we show that algebraic attacks can be drastically more efficient than combinatorial attacks

to solve **RD**. More precisely, when the target rank weight r to decode is in $\mathcal{O}(\sqrt{n})$, where n is the length of the code, then algebraic attacks are far more efficient than classical combinatorial attacks.

Since this area is typically the one used in cryptography, these attacks were devastating against the parameters of ROLLO and RQC, two rank-based cryptosystems that made it to the Second Round of the NIST Post-Quantum Standardization Process. Our approach to get this far more efficient algebraic attack against **RD** relies on the same aforementioned technique used to solve the **MinRank** problem.

Generalizations of RD. There exist generalizations of the classical Rank Decoding (**RD**) problem, for instance the Rank Support Learning (**RSL**) problem, the Non-Homogeneous Rank Decoding (**NHRD**) problem, and the Non-Homogeneous Rank Support Learning (**NHRSL**) problem.

These problems enable one to build more advanced or more efficient cryptographic primitives; for instance **RSL** is used in the Durandal signature scheme [ABG+19], and **NHRD** is used in the latest version of RQC [AAB+20].

Thus, studying the complexity of these problems is primordial to evaluate the security of rank-based cryptosystems, with respect to both combinatorial and algebraic attacks.

Recall that an $\mathbb{F}_{q^m}$-linear code in the rank metric is defined by the following parameters: m is the degree of the extension field $\mathbb{F}_{q^m}$, n the length of the code, k its dimension, and r the target rank weight to decode.

In Section 3.3, we describe a combinatorial attack against **RSL**, moreover, this new attack enabled us to refine a fundamental bound on the complexity of this problem. More specifically, in the Rank Support Learning problem, one is given N syndromes, whose corresponding errors all share the same support; when $N = 1$, it is the classical **RD** problem, and it was proven in [GHPT17] that the problem could be solved in polynomial time if $N \geq nr$. Later, in [DT18], it was proven that **RSL** could be solved in subexponential time as long as $N \geq kr$.

It is this bound that we improved upon by showing that any **RSL** instance fulfilling

$$N \geq kr\frac{m}{m-r}$$

can be solved in polynomial time.

We also propose explicit formulas for the algebraic attack against **RSL** [BB21]; and then we give a way to visualize the complexities of the different attacks against **RSL** using graphs.

Then, we adapt the classical combinatorial and algebraic attacks against **RD** to the case of **NHRD**, extending the analysis we made in [AAB+20].

Finally, we propose a combinatorial attack against the **NHRSL** problem.

PSSI. Durandal [ABG+19] is a rank-based signature scheme built using the Schnorr-Lyubashevsky framework [Sch91, Lyu09]. The security of Durandal relies on **RSL**, but also on a problem whose hardness has been less studied, namely the Product Spaces Subspaces Indistinguishability problem (**PSSI**).

In Section 3.4, we give a randomized reduction from **PSSI** to **MinRank**. This reduction enables one to attack any **PSSI** instance using state-of-the-art attacks against **MinRank**. Note that this leads to the first algebraic attack against **PSSI**.

Cryptanalysis of a signature scheme. In Section 3.5, we describe the Rank Preserving Signature scheme (RPS), published in 2020 by Lau and Tan [LT20b]. In a few words, it is a rank-based signature scheme that uses the Schnorr-Lyubashevsky framework [Sch91, Lyu09] together with ephemeral keys, *i.e.*, keys that change with every signature.

Then, we give two attacks of combinatorial nature against the RPS scheme: the first one uses one single valid signature in order to forge a new one, whereas the second attack does not require any signature since it aims at simply forging one.

Since our attacks are combinatorial, they benefit from a quantum speedup using Grover's algorithm; this is why we also give a quantum version of our attacks.

Overall, our attacks, published in [ABG21], show the flaws in the RPS signature scheme, and they break all its proposed parameters, both the classical parameters and the quantum ones, that is to say the ones targeting quantum security levels.

1.2.2 Improvements on rank-based cryptosystems

Rank Quasi-Cyclic. Rank Quasi-Cyclic [AMBD+18] is a rank-based encryption scheme that made it to the Second Round of the NIST Post-Quantum Standardization Process. Its main advantage is that its security relies solely on decoding a

random code, whereas LRPC-like cryptosystems require an additional indistinguishability assumption.

The fact that RQC is more secure comes at a price: its parameters are bigger than the ones of other cryptosystems such as ROLLO. This means that the sizes of RQC's public key and ciphertext are larger.

In Section 4.1, we introduce a new family of efficiently decodable codes in the rank metric, namely the Augmented Gabidulin codes.

Then, using these codes together with techniques coming from [AAD+22, AAB+20, Wan19], we propose two new versions of the RQC scheme.

The first one uses the ideal structure and has very competitive parameters; the second one relies only on random codes, thus it is more secure, yet it still offers competitive parameters.

In order to study the security of these cryptosystems, namely Multi-RQC-AG and Multi-UR-AG, we use the results obtained in Sections 3.2 and 3.3.

1.2.3 Introduction of a new problem

The last chapter of this document, Chapter 5, is dedicated to the introduction of a new problem, namely **SquareSpace**.

Let $E = \langle e_1, e_2, \ldots, e_r \rangle$ be a vector space of $\mathbb{F}_{q^m}$ of dimension r, let U be the vector space of dimension t spanned by the products $e_i e_j$, $1 \leq i \leq j \leq r$; we call this vector space the *square space* of E. In a few words, given $U = E^2$, the **SquareSpace** problem asks one to recover the secret space E.

In Chapter 5, we present this problem and then study its complexity by providing several combinatorial and algebraic attacks against it. Using the results from the aforementioned attacks, we introduce four **SquareSpace** challenges targeting the security level of 128 bits.

Most importantly, we apply this problem to the design of an authentication protocol; then, using the Fiat-Shamir transformation [FS87], we turn this protocol into a signature scheme whose security relies on the **SquareSpace** problem.

Last but not least, we propose a C implementation of this signature scheme.

Contributions	Sec.	Article / Preprint	Impacts
New algebraic attacks against **MinRank**, namely *SupportMinors*	3.1	[BBC+20] [BBB+22]*	Improves the previous attacks, our attack has been used by Beullens in Rainbow's cryptanalysis [Beu22].
New algebraic attacks against **RD**, namely *MaxMinors*	3.2	[BBB+20] [BBC+20] [BBB+22]*	Improves the previous attacks, it caused ROLLO and RQC not to move to the 3rd Round of NIST PQSP.
New attacks against **NHRD**, **RSL**, and **NHRSL**	3.3	[BBBG22]*	These improve the previous attacks, and lead to a better understanding of rank-based cryptography problems.
Randomized reduction from **PSSI** to **MinRank**	3.4	In progress	**PSSI** (problem associated to the Durandal signature scheme) now benefits from all the combinatorial and algebraic attacks against **MinRank**.
Attacks against the RPS signature scheme	3.5	[ABG21]	All proposed RPS parameters, both classical and quantum, are broken.
Improvements of the RQC scheme	4	[BBBG22]*	We propose two new schemes, with or without ideal structure, with competitive parameters.
Introduction of a new problem, namely **SquareSpace**	5	In progress	Analysis of its hardness, description of a few attacks, proposition of 4 challenges, design of a signature scheme, and implementation in C language.

Table 1.2: Summary of our contributions. For each of them: the corresponding section in this document (column "Sec."), our work associated to it, and the impacts it has in a few words. The star symbol by a citation means that it has not been peer-reviewed yet. The acronym NIST PQSP stands for NIST Post-Quantum Standardization Process, see the list of acronyms on page 16.

Chapter 2

Preliminaries

Contents

2.1 Coding Theory

In this section, we give some basic materials in coding theory.

For more information about codes in the Hamming metric, the reader may refer to the very complete book by Pless [Ple98], or to the introduction to coding theory lecture by Couvreur [Cou19].

Concerning the rank metric, the reader may refer to Loidreau's HDR[1] thesis [Loi07].

2.1.1 Coding theory Basics

An error-detecting code or an error-correcting code aims at detect or correct errors that might occur when a message is transmitted through a noisy channel.

In the rest of this document, we will only focus on error-correcting codes, that we simply call *codes* for the sake of simplicity.

The key point in coding theory is to add an extra information to a message, called the redundancy.

Let k, n be integers such that $k \leq n$, let $m \in \mathbb{F}_q^k$ be a message.

The encoding process is an application $\mathcal{E}$ which goes from $\mathbb{F}_q^k$ to $\mathbb{F}_q^n$, we say that $\mathcal{E}(m)$ is the codeword associated to the message m.

In what follows, we will only focus on linear codes, that is to say, codes for which any $\mathbb{F}_q$-linear combination of codewords belongs to the code.

Definition 1 (Linear code (length, dimension))**.** *Let k, n be integers such that $k \leq n$.*

A linear code $\mathcal{C}$ is an $\mathbb{F}_q$-vector space of $\mathbb{F}_q^n$ of dimension k.

The length *of the code is n, and its* dimension *is k, we say that it is a $[n, k]_q$ code, or simply $[n, k]$ when there is no ambiguity.*

A code can be described by a generator matrix or a parity-check matrix.

[1] Acronym of "Habilitation à Diriger des Recherches", highest french academic degree.

Definition 2 (Generator matrix)**.** *Let $\mathcal{C}$ be a $[n,k]_q$ code. A generator matrix of $\mathcal{C}$ is a matrix $\boldsymbol{G}$ in $\mathbb{F}_q^{k\times n}$ whose rows span $\mathcal{C}$:*

$$\mathcal{C} = \{\boldsymbol{m}\boldsymbol{G}, \boldsymbol{m} \in \mathbb{F}_q^k\}.$$

Definition 3 (Parity-check matrix)**.** *Let $\mathcal{C}$ be a $[n,k]_q$ code. A parity-check matrix of $\mathcal{C}$ is a matrix $\boldsymbol{H}$ in $\mathbb{F}_q^{(n-k)\times n}$ such that:*

$$\mathcal{C} = \{\boldsymbol{x} \in \mathbb{F}_q^n, \boldsymbol{H}\boldsymbol{x}^\intercal = (\boldsymbol{0}^n)^\intercal \in \mathbb{F}_q^k\}.$$

For the sake of simplicity, when there is no ambiguity, we might write simply

$$\boldsymbol{H}\boldsymbol{x}^\intercal = \boldsymbol{0}.$$

There are several generator matrices (resp. parity-check matrices) for a given code, we speak about *the* generator matrix (resp. *the* parity-check matrix) when it has an identity block on its left. Such matrices always exist for a given code, up to a permutation of the coordinates.

In that case, we say that the generator matrix (resp. parity-check matrix) is in *systematic* form, for instance

$$\boldsymbol{G} = (\boldsymbol{I}_k|\boldsymbol{A})$$

is a generator matrix in systematic form.

A fundamental point in coding theory is the notion of weight. Let us start with the classical Hamming weight, see Definition 4.

Definition 4 (Hamming weight and distance)**.** *Let $\boldsymbol{x} = (x_1, x_2, \ldots, x_n) \in \mathbb{F}_q^n$, the Hamming weight of $\boldsymbol{x}$ is defined as*

$$w_H(\boldsymbol{x}) := \#\{i, x_i \neq 0\}.$$

The Hamming distance between two vectors $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{F}_q^n$ is defined as

$$d_H(\boldsymbol{x}, \boldsymbol{y}) := w_H(\boldsymbol{x} - \boldsymbol{y}).$$

The minimal distance of a code, denoted d, is the minimal distance between two distinct codeword. By linearity, it is straightforward that it is equal to the minimum weight of a non-zero codeword.

If $\mathcal{C}$ is a code over $\mathbb{F}_q$ of length n, dimension k, and minimal distance d, we say that it is an $[n,k,d]_q$ code, or simply an $[n,k,d]$ when there is no ambiguity.

The notion of minimal distance is very important since it gives the error-correcting capacity of a code, often referred to as the *decoding capacity* of a code, see Proposition 1.

The decoding capacity of a code $\mathcal{C}$ corresponds to the maximum distance t such that any noisy codeword at distance t of a codeword in $\mathcal{C}$ can be *uniquely* decoded.

In other words, it means that if one receives $y = c + e$ where $c \in \mathcal{C}$ and $w(e) \leq t$, then the noisy codeword y cannot be decoded into another codeword $c' \neq c$.

Proposition 1 (Decoding capacity)**.** *Let $\mathcal{C}$ be a $[n, k, d]_q$ code, and let $\boldsymbol{y}$ be a noisy codeword of $\mathcal{C}$ where its associated error has weight t. If*

$$t \leq \left\lfloor \frac{d-1}{2} \right\rfloor,$$

then the problem of decoding $\boldsymbol{y}$ has at most a unique solution.

Proof. Let $\boldsymbol{c}_1 \in \mathcal{C}$, and let us assume that $\boldsymbol{y} = \boldsymbol{c}_1 + \boldsymbol{e}_1$ such that $w(e_1) \leq t$ can be decoded into another codeword $\boldsymbol{c}_2$.
This means that there exists $\boldsymbol{e}_2$ such that $\boldsymbol{y} = \boldsymbol{c}_2 + \boldsymbol{e}_2$, and $w(\boldsymbol{e}_2) \leq t$. One gets that:

$$\begin{aligned} d(\boldsymbol{c}_1, \boldsymbol{c}_2) &= d(\boldsymbol{e}_1, \boldsymbol{e}_2) && \text{(by hypothesis)} \\ &\leq d(\boldsymbol{e}_1, 0) + d(0, \boldsymbol{e}_2) && \text{(triangle inequality)} \\ &\leq 2t && \text{(by hypothesis)} \\ &\leq d - 1 && \text{(by definition).} \end{aligned}$$

Since $\boldsymbol{c}_1, \boldsymbol{c}_2$ both belong to a code of minimal distance d, it means by definition that the distance between them is greater or equal to d or 0.

So, the distance between $\boldsymbol{c}_1$ and $\boldsymbol{c}_2$ is 0, which means that $\boldsymbol{c}_1 = \boldsymbol{c}_2$; thus, $\boldsymbol{y}$ is uniquely decoded into $\boldsymbol{c}_1$. □

Remark 1. *Note that the decoding capacity t mentioned in Proposition 1 does not depend on the metric chosen. For instance, this proposition still holds for the rank metric which will be defined later in this document.*

So far, we have seen that given a $[n, k, d]_{\mathbb{F}_q}$ code $\mathcal{C}$, one can uniquely decode a noisy codeword with at most $t := \left\lfloor \frac{d-1}{2} \right\rfloor$ errors.

However, this decoding process is hard if the code does not have a specific structure.

For example, in the Hamming metric: Reed-Solomon or MDPC (Moderate Density Parity-Check matrix) codes benefit from efficient, *i.e.*, polynomial-time, decoding algorithms, whereas decoding random linear codes is an NP-complete problem, see Definition 5.

Definition 5 (Decoding problem (Hamming metric))**.**
Input : *a matrix* $\boldsymbol{G} \in \mathbb{F}_q^{k\times n}$*, an integer* r*, and a vector* $\boldsymbol{y} \in \mathbb{F}_q^n$*.*
Output : $\boldsymbol{e} \in \mathbb{F}_q^n$*, such that*

$$\boldsymbol{y} - \boldsymbol{e} = \boldsymbol{m}\boldsymbol{G},\ \boldsymbol{m} \in \mathbb{F}_q^k,\ \text{and } w_H(\boldsymbol{e}) \leq r.$$

In 1978, this problem has been proven to be NP-complete by Berlekamp, McEliece, and VanTilborg in [BMvT78].

Using the same notation as in Definition 5, if $\boldsymbol{H}$ is a parity-check matrix of the code generated by the rows of $\boldsymbol{G}$, one has that

$$\boldsymbol{H}\boldsymbol{y}^\intercal = \boldsymbol{H}(\boldsymbol{m}\boldsymbol{G} + \boldsymbol{e})^\intercal = \boldsymbol{H}(\boldsymbol{m}\boldsymbol{G})^\intercal + \boldsymbol{H}\boldsymbol{e}^\intercal = \boldsymbol{H}\boldsymbol{e}^\intercal \tag{2.1}$$

by definition of $\boldsymbol{H}$.

The quantity $\boldsymbol{H}\boldsymbol{e}^\intercal$ is called the *syndrome* associated to the error $\boldsymbol{e}$; using Equation (2.1), one notices that it only depends on the error. In other words, two noisy codewords $\boldsymbol{y} = \boldsymbol{c}_1 + \boldsymbol{e}$ and $\boldsymbol{y} = \boldsymbol{c}_2 + \boldsymbol{e}$ with $\boldsymbol{c}_1 \neq \boldsymbol{c}_2$, lead to the same syndrome.

This yields another version of the decoding problem, namely the *Syndrome Decoding* problem; in this version, one is simply given a syndrome and has to recover an error of small weight, see Definition 6.

Definition 6 (Syndrome Decoding problem (Hamming metric))**.**
Input : *a matrix* $\boldsymbol{H} \in \mathbb{F}_q^{(n-k)\times n}$*, an integer* r*, and a vector* $\boldsymbol{s} \in \mathbb{F}_q^{n-k}$*.*
Output : $\boldsymbol{e} \in \mathbb{F}_q^n$*, such that*

$$\boldsymbol{H}\boldsymbol{e}^\intercal = \boldsymbol{s}^\intercal,\ \text{and } w_H(\boldsymbol{e}) \leq r.$$

Remark 2. *It is worth noticing that the Decoding and the Syndrome Decoding problem are two different formulations of the very same problem. This remains true no matter the metric one chooses. In this document, for the rank metric for instance, we always refers to the Rank Decoding (***RD***) problem, even if one uses a parity-check matrix.*

Last but not least, there is a primordial bound for code in the Hamming metric, namely the *Gilbert-Varshamov* bound.

Definition 7 (Gilbert-Varshamov bound (Hamming metric))**.** *Let n, k be integers such that $k \leq n$. The Gilbert-Varshamov bound* $\mathrm{GV}(q, n, k)$ *for $\mathbb{F}_q$-linear code of length n and dimension k in the Hamming metric is defined as the smallest positive integer t such that*

$$q^{n-k} \leq \sum_{i=0}^{t} \binom{n}{i} (q-1)^i.$$

Remark 3. *A very interesting property of random linear codes is that their minimal distance is given by the Gilbert-Varshamov bound with overwhelming probability. For a proof of this result, the reader may refer to [Cou19].*

2.1.2 Rank metric

In this section, we will consider $\mathbb{F}_{q^m}$-linear codes, that is to say codes that are subspaces of $\mathbb{F}_{q^m}^n$ for a given length n.

Let n, k be integers such that $k \leq n$, given a $[n, k]_{\mathbb{F}_{q^m}}$-code, instead of using the Hamming weight in $\mathbb{F}_{q^m}^n$, in what follows we will consider the rank weight, see Definition 8.

In a few words, while the Hamming metric looks at the number of non-zero entries of a vector $\boldsymbol{v} \in \mathbb{F}_{q^m}^n$, the rank metric looks at the rank of the matrix $\boldsymbol{V} \in \mathbb{F}_q^{m \times n}$ containing the coordinates of every entries of $\boldsymbol{v}$ in a basis β of $\mathbb{F}_{q^m}$ over $\mathbb{F}_q$.

Definition 8 (Rank weight)**.** *Let $\boldsymbol{x} = (x_1, \ldots, x_n) \in \mathbb{F}_{q^m}^n$ be a vector. The rank weight of $\boldsymbol{x}$, denoted $\|\boldsymbol{x}\|$, is defined as the rank of the matrix*

$$\mathrm{Mat}\,(\boldsymbol{x}) := (x_{i,j})_{i,j} \in \mathbb{F}_q^{m \times n}$$

where

$$\forall\, j \in \{1..n\}, \quad x_j = \beta_1 x_{1,j} + \cdots + \beta_m x_{m,j}.$$

Recall that with the Hamming metric, the support of a vector is the set containing the indices of all its non-zero coordinates. Definition 9 adapts this notion to the rank metric.

Definition 9 (Support of a word in $\mathbb{F}_{q^m}^n$)**.** *The support of $\boldsymbol{x} \in \mathbb{F}_{q^m}^n$, denoted* Support($\boldsymbol{x}$)*, is the $\mathbb{F}_q$-linear space generated by the coordinates of $\boldsymbol{x}$, i.e.,*

$$\text{Support}(\boldsymbol{x}) := \langle x_1, \ldots, x_n \rangle_{\mathbb{F}_q}.$$

It follows from the definition that

$$\|\boldsymbol{x}\| = \dim_{\mathbb{F}_q} (\text{Support}(\boldsymbol{x})).$$

These notions can be extended to matrices. The support of a matrix $\mathbf{M} \in \mathbb{F}_{q^m}^{r \times c}$, denoted Support($\mathbf{M}$), is the $\mathbb{F}_q$-vector space spanned by all its rc entries, and the rank weight $\|\mathbf{M}\|$ is defined as the dimension of this support.

Note that the rank weight $\|\mathbf{M}\|$ of a matrix $\boldsymbol{M}$ is very different from the actual rank of $\boldsymbol{M}$; for example $\|\mathbf{I}_n\| = 1$ while $\text{Rank}\,(\mathbf{I}_n) = n$.

We can now define formally an $\mathbb{F}_{q^m}$-linear code, and we naturally extend the notions of generator and parity-check matrices in this context:

Definition 10 ($\mathbb{F}_{q^m}$-linear code)**.** *An $\mathbb{F}_{q^m}$-linear code $\mathcal{C}$ of length n and dimension k is an $\mathbb{F}_{q^m}$-linear subspace of dimension k of $\mathbb{F}_{q^m}^n$ embedded with the rank metric. We say that $\mathcal{C}$ is a $[n,k]_{q^m}$-code, or simply $[n,k]$ when there is no ambiguity.*

A generator matrix of $\mathcal{C}$ is a matrix $\boldsymbol{G} \in \mathbb{F}_{q^m}^{k \times n}$ such that

$$\mathcal{C} = \left\{ \boldsymbol{m}\boldsymbol{G},\ \boldsymbol{m} \in \mathbb{F}_{q^m}^k \right\}.$$

A parity-check matrix of $\mathcal{C}$ is a matrix $\boldsymbol{H} \in \mathbb{F}_{q^m}^{(n-k) \times n}$ such that

$$\mathcal{C} = \left\{ \boldsymbol{x} \in \mathbb{F}_{q^m}^n,\ \boldsymbol{H}\boldsymbol{x}^\intercal = 0 \right\}.$$

We say that $\boldsymbol{G}$ (respectively $\boldsymbol{H}$) is in systematic form if it is of the form $(\boldsymbol{I}_k \mid \boldsymbol{A})$ (respectively $(\boldsymbol{I}_{n-k} \mid \boldsymbol{B})$).

The rank metric is very natural to use when dealing with matrix codes [Del78].

Gabidulin introduced the first efficiently decodable $\mathbb{F}_{q^m}$-linear codes with the rank metric, namely the Gabidulin codes, see Section 2.1.4.

Remark 4. *As noted in [BBC+20], every $\mathbb{F}_{q^m}$-linear code of length n and dimension k can be seen as a very particular matrix code of length nm and dimension km.*

2.1.3 Ideal codes for the rank metric

In a cryptographic context, one wants to reduce keysizes; a typical way to do this in code-based cryptography is to use ideal codes, see Definition 12 below.

Let $P \in \mathbb{F}_q[X]$ denote a polynomial of degree n.

$$\Psi \colon \mathbb{F}_{q^m}^n \longrightarrow \mathbb{F}_{q^m}[X]/\langle P \rangle \tag{2.2}$$

$$\boldsymbol{u} := (u_0, \ldots, u_{n-1}) \longmapsto \sum_{i=0}^{n-1} u_i X^i. \tag{2.3}$$

The linear map Ψ given by Equation (2.2) is a vector space isomorphism between $\mathbb{F}_{q^m}^n$ and $\mathbb{F}_{q^m}[X]/\langle P \rangle$.

We can use it to define a product between two elements $\boldsymbol{u}, \boldsymbol{v}$ in $\mathbb{F}_{q^m}^n$: $\boldsymbol{u} \cdot \boldsymbol{v} := \boldsymbol{w} \in \mathbb{F}_{q^m}^n$ is the only vector such that

$$\Psi(\boldsymbol{w}) = \Psi(\boldsymbol{u})\Psi(\boldsymbol{v}) \mod P.$$

In the rest of this document, for the sake of simplicity and to lighten the formulas, we might omit the Ψ or the $\cdot$ symbols when there is no ambiguity.

Since one has that

$$\begin{aligned}
\Psi(\boldsymbol{u} \cdot \boldsymbol{v}) &= \Psi(\boldsymbol{u})\Psi(\boldsymbol{v}) \mod P \\
&= \sum_{i=0}^{n-1} u_i X^i \Psi(\boldsymbol{v}) \mod P \\
&= \sum_{i=0}^{n-1} u_i \left(X^i \Psi(\boldsymbol{v}) \mod P \right) \\
&= (u_0, \ldots, u_{n-1}) \begin{pmatrix} \Psi(\boldsymbol{v}) \mod P \\ X\Psi(\boldsymbol{v}) \mod P \\ X^2\Psi(\boldsymbol{v}) \mod P \\ \vdots \\ X^{n-1}\Psi(\boldsymbol{v}) \mod P \end{pmatrix},
\end{aligned}$$

it is natural to define the $n \times n$ matrix $\mathcal{IM}(\boldsymbol{v})$ with entries in $\mathbb{F}_{q^m}$ such that:

$$\boldsymbol{u} \cdot \boldsymbol{v} = \boldsymbol{u}\mathcal{IM}(\boldsymbol{v}) \in \mathbb{F}_{q^m}^n.$$

This matrix is called the *multiplication matrix* or the *ideal matrix* generated by $\boldsymbol{v}$ and P, see Definition 11.

Definition 11 (Ideal matrix)**.** *Let $P \in \mathbb{F}_q[X]$ be a polynomial of degree n, and let $\boldsymbol{v}$ be a vector in $\mathbb{F}_{q^m}^n$.*

The ideal matrix *generated by $\boldsymbol{v}$ and P, noted $\mathcal{IM}_P(\boldsymbol{v})$, is defined by:*

$$\mathcal{IM}_P(\boldsymbol{v}) := \begin{pmatrix} \boldsymbol{v} \\ \Psi^{-1}\left(X\Psi(\boldsymbol{v}) \mod P\right) \\ \Psi^{-1}\left(X^2\Psi(\boldsymbol{v}) \mod P\right) \\ \vdots \\ \Psi^{-1}\left(X^{n-1}\Psi(\boldsymbol{v}) \mod P\right) \end{pmatrix} \in \mathbb{F}_{q^m}^{n\times n}.$$

For conciseness, we use the notation $\mathcal{IM}(\boldsymbol{v})$ since there will be no ambiguity in the choice of P in this document.

With ideal matrices, the product of two elements in $\mathbb{F}_{q^m}^n$ is equivalent to the usual vector-matrix product, namely:

$$\boldsymbol{u} \cdot \boldsymbol{v} = \boldsymbol{u}\mathcal{IM}(\boldsymbol{v}) = \mathcal{IM}(\boldsymbol{u})^\intercal \boldsymbol{v} = \boldsymbol{v} \cdot \boldsymbol{u}.$$

An ideal code $\mathcal{C}$ of parameters $[sn, tn]_{q^m}$ is an $\mathbb{F}_{q^m}$-linear code which admits a generating matrix made of $s \times t$ ideal matrix blocks in $\mathbb{F}_{q^m}^{n\times n}$.

Hereafter, we only consider $[sn, n]_{q^m}$-ideal codes, *i.e.*, $t = 1$.

A very important point, regarding the choice of the modulus P, is that if $P \in \mathbb{F}_q[X]$ is *irreducible* of degree n, and if n and m are different *prime* numbers, then an ideal code using P always admits a *systematic* generator matrix made of ideal blocks, see Lemma 1 in [AAB+20].

Definition 12 (Ideal codes)**.** *Let $P \in \mathbb{F}_q[X]$ a polynomial of degree n. An $[ns, n]_{q^m}$-code $\mathcal{C}$ is an* ideal code *if it has a generator matrix of the form*

$$\boldsymbol{G} = \left(\boldsymbol{I}_n \mid \mathcal{IM}(\mathbf{g}_1) \mid \ \ldots \ \mid \mathcal{IM}(\mathbf{g}_{s-1})\right) \in \mathbb{F}_{q^m}^{n\times ns},$$

where $\mathbf{g}_i \in \mathbb{F}_{q^m}^n,\ \forall\ i \in \{1..s-1\}$.

Similarly, $\mathcal{C}$ is an ideal code if it admits a parity-check matrix of the form

$$\boldsymbol{H} = \begin{pmatrix} & \mathcal{IM}(\mathbf{h}_1)^\intercal \\ \boldsymbol{I}_{n(s-1)} & \vdots \\ & \mathcal{IM}(\mathbf{h}_{s-1})^\intercal \end{pmatrix} \in \mathbb{F}_{q^m}^{n(s-1)\times ns}.$$

2.1.4 Gabidulin codes

Gabidulin codes were introduced by Gabidulin in 1985 [Gab85]. These codes can be seen as the rank metric analog of Reed-Solomon codes [RS60], where standard polynomials are replaced by q-polynomials (also called Ore polynomials or linearized polynomials).

Definition 13 (q-polynomials)**.** *The set of q-polynomials over $\mathbb{F}_{q^m}$ is the set of polynomials with the following shape:*

$$\left\{ P(X) = \sum_{i=0}^{r} p_i X^{q^i}, \text{ with } p_i \in \mathbb{F}_{q^m} \text{ and } p_r \neq 0 \right\}.$$

The q-degree of a q-polynomial P is defined as $\deg_q(P) = r$.

Definition 14 (Ring structure)**.** *The set of q-polynomials over $\mathbb{F}_{q^m}$ is a* non-commutative *ring when equipped with the two following operations $(+, \circ)$:*

- *Addition:* $(P + Q)(X) = P(X) + Q(X)$,
- *Composition:* $(P \circ Q)(X) = P[Q(X)]$.

Due to their structure, the q-polynomials are inherently related to decoding problems in the rank metric as stated by the following propositions.

Theorem 1 ([Ore33])**.** *Any $\mathbb{F}_q$-subspace of $\mathbb{F}_{q^m}$ of dimension r is the set of the roots of a unique monic q-polynomial P such that* $\deg_q(P) = r$.

Corollary 1. *Let $\boldsymbol{x} = (x_1, x_2, \ldots, x_n) \in \mathbb{F}_{q^m}^n$ and V be the monic q-polynomial of smallest q-degree such that $V(x_i) = 0$ for $1 \leq i \leq n$, then $\|\boldsymbol{x}\| = r$ if, and only if, if* $\deg_q(V) = r$.

Finally, Gabidulin codes can be seen as evaluation codes using q-polynomials of bounded degree.

Definition 15 (Gabidulin codes)**.** *Let $k, n, m \in \mathbb{N}$ such that $k \leq n \leq m$, and let $\boldsymbol{g} = (g_1, \ldots, g_n)$ be an-$\mathbb{F}_q$ linearly independent family of elements of $\mathbb{F}_{q^m}$. The* Gabidulin code $\mathcal{G}_{\boldsymbol{g}}(n, k, m)$ *is the code of parameters $[n, k]_{q^m}$ defined by*

$$\mathcal{G}_{\boldsymbol{g}}(n, k, m) := \left\{ P(\boldsymbol{g}), \ \deg_q(P) < k \right\}, \text{ where } P(\boldsymbol{g}) := (P(g_1), \ldots, P(g_n)).$$

Thus, a generator matrix for $\mathcal{G}_{\boldsymbol{g}}$ is given by:

$$\boldsymbol{G} = \begin{pmatrix} g_1 & \cdots & g_n \\ g_1^q & \cdots & g_n^q \\ \vdots & \vdots & \vdots \\ g_1^{q^{k-1}} & \cdots & g_n^{q^{k-1}} \end{pmatrix} \in \mathbb{F}_{q^m}^{k \times n}.$$

Proposition 2 (Properties of Gabidulin codes, [Gab85])**.** *Let $\mathcal{G}_{\boldsymbol{g}}(n,k,m)$ be a Gabidulin code, it benefits from an efficient decoding algorithm with can correct up to*

$$\left\lfloor \frac{n-k}{2} \right\rfloor$$

errors.

Moreover, if $m = n$, $\mathcal{G}_{\boldsymbol{g}}(n,k,m)$ satisfies the Singleton bound with a minimal distance $d = n - k + 1$; in that case, they are said to be Maximal Rank Distance *(MRD) codes.*

2.2 Hard problems in rank-based cryptography

2.2.1 Decisional versus Search

An NP-complete problem is by definition a decisional problem; *i.e.*, it asks to decide whether a solution exists or not.

For instance, the **MinRank** problem, see Definition 16, asks whether there exists a non-trivial linear combination of a set of matrices such that the rank of this linear combination is smaller than a threshold.

A *search* problem, sometimes called a *computational* problem, is a problem for which the output is a solution, or a special output such as `FAIL` if there is no solution.

For all the problems described in Section 2.2.2, there is no known reduction from the search to the decision version of the problem.

In other words, the current best way for an adversary to solve an instance of a decisional problem is to try to solve the associated search instance, and to reply `YES` if she finds a solution, and `NO` otherwise.

In this document, we give only the definitions of the search versions, and we refer to the decisional version by adding an uppercase "D" in front of the problem's name.

For instance **DRD** refers to the decisional version of the **RD** problem, see the list of acronyms on page 16.

When we mention the *NP-completeness* of a problem, we obviously refer to its decisional version; even if for the sake of clarity, we might omit to explicitly mention it.

2.2.2 Definitions of the problems

The **MinRank** problem has been introduced in [BFS99] where its NP-completeness was proven.

For another quite more easy to understand proof of its NP-completeness, the reader may refer to [Cou01b].

Definition 16 (**MinRank** problem (search))**.** *The* **MinRank** *problem with parameters* (m, n, K, r) *is given by:*
Input : *an integer* $r \in \mathbb{N}$*, and* K *matrices* $\boldsymbol{M}_1, \boldsymbol{M}_2, \ldots, \boldsymbol{M}_K \in \mathbb{F}_q^{m \times n}$*.*
Output : *elements* $x_1, x_2, \ldots, x_K \in \mathbb{F}_q$*, non all zeros, such that*

$$\text{Rank}\left(\sum_{i=1}^{K} x_i \boldsymbol{M}_i\right) \leq r.$$

Remark 5. *The* **MinRank** *problem is sometimes defined with* $K+1$ *matrices* $\boldsymbol{M}_0, \boldsymbol{M}_1, \boldsymbol{M}_2, \ldots, \boldsymbol{M}_K$ *where one wants* $\boldsymbol{M}_0 + \sum_{i=1}^{K} x_i \boldsymbol{M}_i$ *to be of rank smaller or equal to* r*. This version is sometimes referred to as* affine *or* inhomogeneous **MinRank**.

The **RD** problem given by Definition 17 is at the core of rank-based cryptography.

Definition 17 (Rank Decoding (**RD**) problem (search))**.** *The* **RD** *problem with parameters* (q, m, n, k, r)*, or parameters* (m, n, k, r) *over* $\mathbb{F}_q$*, is given by:*
Input : *an* $\mathbb{F}_{q^m}$*-linear subspace* $\mathcal{C}$ *of* $\mathbb{F}_{q^m}^n$*, an integer* $r \in \mathbb{N}$*, and a vector* $\boldsymbol{y} \in \mathbb{F}_{q^m}^n$ *such that* $\|\boldsymbol{y} - \boldsymbol{c}\| \leq r$ *for some* $\boldsymbol{c} \in \mathcal{C}$*.*
Output : $\boldsymbol{c} \in \mathcal{C}$*, and an error* $\boldsymbol{e} \in \mathbb{F}_{q^m}^n$ *such that* $\boldsymbol{y} = \boldsymbol{c} + \boldsymbol{e}$*, and* $\|e\| \leq r$*.*

The *syndrome* version of the Rank Decoding problem is called the Rank Syndrome Decoding problem (**RSD**). In this version, one is given a parity check matrix $\boldsymbol{H} \in \mathbb{F}_{q^m}^{(n-k)\times n}$ of the code $\mathcal{C}$, and a syndrome $\boldsymbol{s}^\intercal = \boldsymbol{H}\boldsymbol{e}^\intercal \in \mathbb{F}_{q^m}^{n-k}$ where $\|e\| \leq r$, and she is asked to recover $\boldsymbol{e}$. Recall that in this document, we only refer to **RD**, see Remark 2.

The **RD** problem is not known to be NP-complete, however there is a randomized reduction from an NP-complete problem, namely decoding in the Hamming metric, to **RD** in [GZ16].

Decoding generic matrix codes is precisely the **MinRank** problem, see Definition 16, and decoding generic $\mathbb{F}_{q^m}$-linear codes is the Rank Decoding problem, see Definition 17.

By Remark 4 in Section 2.1.2, one readily sees that it is always possible to decode an $\mathbb{F}_{q^m}$-linear code by solving the **MinRank** instance associated to its matrix code; this corresponds to the reduction from **RD** to **MinRank**.
However, this approach is not efficient since it does not take advantage of the $\mathbb{F}_{q^m}$-linearity.

In order to build more advanced cryptosystems, see for instance Section 4.1, one needs more general versions of this problem.

For instance the Rank Support Learning (**RSL**) problem roughly corresponds to **RD** where one is given several syndromes whose associated errors share the same support. The **RSL** problem has been in introduced in [GHPT17], and it is at the core of the Durandal signature scheme [ABG+19], and the recent Multi-LRPC scheme [AAD+22].

Definition 18 (Rank Support Learning (**RSL**) problem (search))**.** *The* **RSL** *problem with parameters* (m, n, k, r, N) *is given by:*
Input : *a full-rank matrix* $\boldsymbol{H} \in \mathbb{F}_{q^m}^{(n-k)\times n}$*, and a matrix* $\boldsymbol{H}\boldsymbol{E}^\intercal \in \mathbb{F}_{q^m}^{(n-k)\times N}$ *where* $\boldsymbol{E} \in \mathbb{F}_{q^m}^{N\times n}$ *has all its entries in a subspace* $\mathcal{V} \subset \mathbb{F}_{q^m}$ *of dimension* $r \in \mathbb{N}$*.*
Output : *the subspace* $\mathcal{V}$*.*

Another generalization of **RD** is the Non-Homogeneous Rank Decoding problem (**NHRD**). In this problem, the rank weight of the error is not the same for different parts of the error. The **NHRD** problem has been introduced in the Second Round update of RQC [AAB+20] in NIST Post-Quantum Standardization process.

Definition 19 (Non-Homogeneous Rank Decoding (**NHRD**) problem (search))**.** *The* **NHRD** *problem with parameters* (m, n, n_1, w_1, w_2) *is given by:*
Input : *a full-rank matrix* $\boldsymbol{H} \in \mathbb{F}_{q^m}^{(n+n_1)\times(2n+n_1)}$*, and a syndrome* $\boldsymbol{s} \in \mathbb{F}_{q^m}^{n+n_1}$*.*
Output : *a vector* $\boldsymbol{e} = (\boldsymbol{e}_1, \boldsymbol{e}_2, \boldsymbol{e}_3) \in \mathbb{F}_{q^m}^{2n+n_1}$ *such that* $\|(\boldsymbol{e}_1, \boldsymbol{e}_3)\| = w_1$*,* $\|\boldsymbol{e}_2\| = w_1 + w_2$*,* $\mathrm{Support}\left((\boldsymbol{e}_1, \boldsymbol{e}_3)\right) \subset \mathrm{Support}\left(\boldsymbol{e}_2\right)$*, and* $\boldsymbol{H}\boldsymbol{e}^\intercal = \boldsymbol{s}^\intercal$*.*

Last but not least, one can somehow combine the **NHRD** and **RSL** problems in order to get the Non-Homogeneous Rank Support Learning problem (**NHRSL**) given by Definition 20.

Definition 20 (Non-Homogeneous Rank Support Learning (**NHRSL**) problem (search))**.** *The* **NHRSL** *problem with parameters* (m, n, n_1, w_1, w_2, N) *is given by:*
Input : *a full-rank matrix* $\boldsymbol{H} \in \mathbb{F}_{q^m}^{(n+n_1)\times(2n+n_1)}$*, and a matrix* $\boldsymbol{H}\boldsymbol{E}^\intercal \in \mathbb{F}_{q^m}^{(n+n_1)\times N}$ *where* $\boldsymbol{E} \in \mathbb{F}_{q^m}^{N\times(2n+n_1)}$ *such that, for all* i *in* $\{1..N\}$*:*

$$\boldsymbol{e}_i = \boldsymbol{E}_{i,*} = (\boldsymbol{e}_{i,1}, \boldsymbol{e}_{i,2}, \boldsymbol{e}_{i,3}), \|(\boldsymbol{e}_{i,1}, \boldsymbol{e}_{i,3})\| = w_1, \|\boldsymbol{e}_{i,2}\| = w_1 + w_2, \text{ and}$$
$$\mathcal{V} := \mathrm{Support}(\boldsymbol{e}_{i,1}, \boldsymbol{e}_{i,3}) \subset \mathcal{W} := \mathrm{Support}(\boldsymbol{e}_{i,2}).$$

Output : *the subspaces* $\mathcal{V}$ *and* $\mathcal{W}$*.*

2.2.3 Gilbert-Varshamov bounds for RD and MinRank

Similarly to the Gilbert-Varshamov bound in the Hamming metric, see Definition 7, there exists a rank metric equivalent of this bound for $\mathbb{F}_{q^m}$-linear codes:

Definition 21 (The Rank Gilbert-Varshamov bound (RGV))**.** *The* Rank Gilbert-Varshamov *bound* $RGV(q, m, n, k)$ *for* $\mathbb{F}_{q^m}$*-linear codes of length* n *and dimension* k *in the rank metric is defined as the smallest positive integer* t *such that*

$$q^{m(n-k)} \leq B_t,$$

where

$$B_t := \sum_{j=0}^{t} \left(\prod_{i=0}^{j-1} (q^n - q^i) \right) \begin{bmatrix} m \\ j \end{bmatrix}_q = \sum_{j=0}^{t} \left(\prod_{i=0}^{j-1} \frac{(q^n - q^i)(q^m - q^i)}{q^j - q^i} \right)$$

is the size of the ball of radius t *in the rank metric.*

Similarly to the Hamming metric, it means that, as long as $w \leq RGV(q, m, n, k)$, a random $\mathbf{RD}(m, n, k, w)$ instance will have at most a unique solution with an overwhelming probability [Loi06].

Concerning the **MinRank** problem, one can use the adaptation of the Gilbert-Varshamov bound to matrix codes.

Definition 22 (The Rank Gilbert-Varshamov bound for matrix codes)**.** *The Gilbert-Varshamov bound for matrix codes of length mn and dimension K over $\mathbb{F}_q$ in the rank metric is defined as the smallest positive integer t such that*

$$q^{mn-K} \leq B_t,$$

where

$$B_t := \sum_{j=0}^{t} \left(\prod_{i=0}^{j-1} (q^n - q^i) \right) \begin{bmatrix} m \\ j \end{bmatrix}_q = \sum_{j=0}^{t} \left(\prod_{i=0}^{j-1} \frac{(q^n - q^i)(q^m - q^i)}{q^j - q^i} \right)$$

is the size of the ball of radius t in the rank metric.

Let w be an integer smaller than the Gilbert-Varshamov bound for matrix codes of length mn and dimension K over $\mathbb{F}_q$.

Applied to **MinRank**, this bound means that if one picks at random K matrices in $\mathbb{F}_q^{m \times n}$, a **MinRank** instance with parameters (q, m, n, K, w) will have, with overwhelming probability, at most a unique solution.

It is common to see the condition $K \leq (m - t)(n - t)$ when dealing with **MinRank** instances [Cou01b, FLdVP08, FSEDS13, VBC+19]; this condition is simply an approximation of the condition given in Definition 22. More precisely, one can replace B_t by an approximation of its greater term:

$$q^{(n+m-t)t},$$

then, applying $\log_q$ to both sides of the inequality, the condition becomes

$$\begin{aligned} mn - K \leq (n + m - t)t \quad &\iff \quad mn - (n + m - t)t \leq K \\ &\iff \quad (m - t)(n - t) \leq K. \end{aligned}$$

Since the Gilbert-Varshamov bound is defined as the *smallest* integer such that this condition is fulfilled, taking $K \leq (m - t)(n - t)$ guarantees that the associated **MinRank** instance will have at most a unique solution with overwhelming probability.

2.2.4 Uniqueness of a solution

In this document, we will focus on instances for which there is always a solution, plus we want this solution to be unique.

This assumption is not very strong since in a cryptographic context, there is always a solution, typically a part of the secret, and it is often unique.

In order to get such instances, *i.e.*, with a unique solution, one uses the bounds given in Section 2.2.3.

For instance, let us apply this to the **RD** problem; let m, n, k be integers, and let r be an integer smaller than $\mathrm{RGV}(q, m, n, k)$, see Definition 21. If one picks a full-rank matrix $\boldsymbol{H}$ in $\mathbb{F}_{q^m}^{(n-k)\times n}$, and a vector $\boldsymbol{e} \in \mathbb{F}_{q^m}^n$ such that $\|\boldsymbol{e}\| = r$, then, with overwhelming probability, the $\mathbf{RD}(m, n, k, r)$ instance given by the couple $(\boldsymbol{H}, \boldsymbol{H}\boldsymbol{e}^\intercal)$ will have a single solution *of rank exactly* r, namely $\boldsymbol{e}$.

The fact that it cannot have another *spurious* solution with overwhelming probability is given by the aforementioned result about random $\mathbb{F}_{q^m}$-linear codes and the Rank Gilbert-Varshamov bound.

In a cryptographic context, the integer r is publicly known since it is usually a parameter of the cryptosystem.

2.3 Algebraic Cryptanalysis

An algebraic attack aims at attacking a cryptosystem by solving a system of algebraic equations. The associated research field is usually called *algebraic cryptanalysis*.

By extension, if a cryptosystem relies on a problem P, we speak about *algebraic attacks against* P.

In this document, what we call *solving* an algebraic system of m equations $f_1, f_2, \ldots, f_m \in \mathbb{F}_q[x_1, x_2, \ldots, x_n]$ refers to finding a point

$$\boldsymbol{c} = (c_1, c_2, \ldots, c_n) \in \mathbb{F}_q^n$$

such that

$$f_1(\boldsymbol{c}) = f_2(\boldsymbol{c}) = \ldots = f_m(\boldsymbol{c}) = 0.$$

A system of algebraic equations one uses to describe a cryptosystem is called a *modeling*.

In this document, we will only focus on algebraic attacks against problems related to public key cryptography. For more general information about algebraic attacks, the reader may refer to [Bar09, Jou09, Spa12, Bar04].

There are several ways to solve a system of algebraic equations; in this document we will use Gröbner bases.

In Section 2.3.1, we recall some basics about Gröbner bases. Then, in Section 2.3.2, we give the reader a general idea of the techniques we will use, in the rest of this document, to compute Gröbner bases.

2.3.1 Gröbner bases

Roughly, computing a Gröbner basis can be seen as applying an elimination algorithm on a system of multivariate polynomials; for instance, a parallel can be made between this process and the Gaussian elimination for linear systems.

For instance, solving the system of polynomial equations over $\mathbb{F}_2[x, y, z, t]$ given by $f_1 = f_2 = \ldots = f_5 = 0$, where

$$\begin{aligned} f_1 &= xz + xt + zt + z + t \\ f_2 &= xy + xz + yt + x + y + z + t + 1 \\ f_3 &= xy + yz + zt + x + t + 1 \\ f_4 &= xz + xt + yz + yt + zt + 1 \\ f_5 &= yz + yt + zt + x + y + z \end{aligned}$$

is not straightforward.

On the contrary, the Gröbner basis of the ideal generated by $f_1, f_2, \ldots, f_5$ is given by

$$\begin{cases} x + 1 \\ y \\ z + 1 \\ t \end{cases},$$

and the solutions of this new system are the very same as the ones of the original system $\mathcal{F}$. Thus, one immediately finds the only solution which is

$$(1, 0, 1, 0) \in \mathbb{F}_2^4.$$

In what follows, we will only give the definitions and results we need, for more details and the proofs, the reader may refer to the very complete book by Cox, Little, and O'Shea [CLO92].

2.3.1.1 Ideals and Varieties

Let $\mathbb{K}$ be a field, recall that $\overline{\mathbb{K}}$ refers to its algebraic closure.

Given a subset S of $\mathbb{K}[x_1, x_2, \ldots, x_n]$, we denote by $\langle S \rangle$ the *ideal generated* by S, *i.e.*,

$$\langle S \rangle := \left\{ \sum_{i=1}^{N} a_i s_i, N \in \mathbb{N}, s_i \in S, a_i \in \mathbb{K}[x_1, x_2, \ldots, x_n], \right\} \subseteq \mathbb{K}[x_1, x_2, \ldots, x_n].$$

As mentioned above, we want to find the solutions to a given system of polynomial equations, that is to say, the points where all the polynomials simultaneously vanish. More formally, this relies on the definition of *affine variety*.

Definition 23 (Affine variety, set of solutions)**.** *Let $\mathbb{L}$ be an extension field of $\mathbb{K}$. The* affine variety *over $\mathbb{L}$ of a set of polynomials $S \subseteq \mathbb{K}[x_1, x_2, \ldots, x_n]$, is defined as*

$$V_{\mathbb{L}}(S) := \{ \boldsymbol{z} \in \mathbb{L}^n, f(\boldsymbol{z}) = 0, \forall f \in S \}.$$

The variety $V_{\overline{\mathbb{K}}}(S)$ is called the (set of) solutions *to the system S.*

Remark 6. *Given $S \subseteq \mathbb{K}[x_1, x_2, \ldots, x_n]$, it is readily seen that a point belongs to $V_{\mathbb{L}}(S)$ if, and only if, if it vanished on every $\mathbb{K}[x_1, x_2, \ldots, x_n]$-linear combination elements of S, hence one has*

$$V_{\mathbb{L}}(S) = V_{\mathbb{L}}(I), \text{ where } I = \langle S \rangle.$$

In practice, we often consider a finite set of polynomials, *i.e.*, $\{f_1, f_2, \ldots, f_m\} \subseteq \mathbb{K}[x_1, x_2, \ldots, x_n]$. For the sake of clarity, we simply write $V_{\mathbb{L}}(f_1, f_2, \ldots, f_m)$ instead of $V_{\mathbb{L}}(\{f_1, f_2, \ldots, f_m\})$.

Hence, by Remark 6 one has

$$V_{\mathbb{L}}(f_1, f_2, \ldots, f_m) = V_{\mathbb{L}}(I), \text{ where } I = \langle f_1, f_2, \ldots, f_m \rangle.$$

Notice that since every ideal is finitely generated by Hilbert's theorem, every affine variety of an ideal is the set of solutions of a finite system of algebraic equations;

more precisely for all $I \subseteq \mathbb{K}[x_1, x_2, \ldots, x_n]$, there exists a finite set of polynomials $f_1, f_2, \ldots, f_m$ such that

$$I = \langle f_1, f_2, \ldots, f_m \rangle ,$$

hence

$$V_{\mathbb{L}}(I) = V_{\mathbb{L}}(f_1, f_2, \ldots, f_m).$$

Definition 24 (Zero-dimensional ideal)**.** *An ideal*

$$I = \langle f_1, f_2, \ldots, f_m \rangle \in \mathbb{K}[x_1, x_2, \ldots, x_n]$$

is said to be zero-dimensional*, or* of dimension zero*, if $V_{\overline{\mathbb{K}}}(I)$ is finite.*

2.3.1.2 Monomial orders and Multivariate division

In order to define the division of multivariate polynomials, see Definition 29, we need to introduce *admissible monomial orders*.

Definition 25 (Admissible monomial order)**.** *An* admissible monomial order *over $\mathbb{K}[x_1, x_2, \ldots, x_n]$ is a total order $<$ on the set of monomials in $\mathbb{K}[x_1, x_2, \ldots, x_n]$ which fulfills the following conditions:*

- *if m_1, m_2 are monomials in $\mathbb{K}[x_1, x_2, \ldots, x_n]$ such that $m_1 < m_2$, then for any monomials $m_3 \in \mathbb{K}[x_1, x_2, \ldots, x_n]$: $m_1 m_3 < m_2 m_3$,*
- *any non-empty set of monomials in $\mathbb{K}[x_1, x_2, \ldots, x_n]$ has a smallest element with respect to $<$; we say that $<$ is a* well-ordering*.*

A classical and very natural admissible monomial order is the lexicographical order, given by Definition 26. In this document, we will mostly deal with graded monomials order, such as the graded reverse lexicographical order, see Definition 27.

Roughly, a graded monomial order is an order where one first compares the degrees of the monomials, and if there are the same, one uses another admissible monomial order.

Definition 26 (Lexicographical monomial order (lex))**.** *The lexicographical monomial order over $\mathbb{K}[x_1, x_2, \ldots, x_n]$ with $x_1 > x_2 > \ldots > x_n$ is denoted $<_{lex}$, and it is defined as follow: let m_1, m_2 be two monomials in $\mathbb{K}[x_1, x_2, \ldots, x_n]$ where*

$$m_1 := x_1^{\alpha_1} x_2^{\alpha_2} \ldots x_n^{\alpha_n}, \quad m_2 := x_1^{\beta_1} x_2^{\beta_2} \ldots x_n^{\beta_n};$$

one has that $m_1 <_{lex} m_2$ if the left-most non-zero term in

$$(\alpha_1 - \beta_1, \alpha_2 - \beta_2, \ldots, \alpha_n - \beta_n)$$

is negative.

Let us have a look at a few examples for the lexicographical order over $\mathbb{K}[x, y, z]$ with $x > y > z$:

- $y <_{\text{lex}} x$ since $(0, 1, 0) - (1, 0, 0) = (-1, 1, 0)$
- $y^2 <_{\text{lex}} x$ since $(0, 2, 0) - (1, 0, 0) = (-1, 2, 0)$,
- $y^2 <_{\text{lex}} xz$ since $(0, 2, 0) - (1, 0, 1) = (-1, 2, -1)$.
- $xz <_{\text{lex}} xy$ since $(1, 0, 1) - (1, 1, 0) = (0, -1, 1)$.

The degree of a monomial

$$m_1 := x_1^{\alpha_1} x_2^{\alpha_2} \dots x_n^{\alpha_n} \in \mathbb{K}[x_1, x_2, \dots, x_n]$$

is naturally defined as

$$\deg(m_1) = \sum_{i=1}^{n} \alpha_i.$$

Definition 27 (Graded Reverse Lexicographical monomial order (grevlex))**.** *The graded reverse lexicographical monomial order over $\mathbb{K}[x_1, x_2, \dots, x_n]$ with $x_1 > x_2 > \dots > x_n$ is denoted $<_{grevlex}$, and it is defined as follow: let m_1, m_2 be two monomials in $\mathbb{K}[x_1, x_2, \dots, x_n]$ where*

$$m_1 := x_1^{\alpha_1} x_2^{\alpha_2} \dots x_n^{\alpha_n}, \quad m_2 := x_1^{\beta_1} x_2^{\beta_2} \dots x_n^{\beta_n};$$

one has that $m_1 <_{grevlex} m_2$ if

$$\deg(m_1) < \deg(m_2),$$

or $\deg(m_1) = \deg(m_2)$, and the right-most non-zero term in

$$(\alpha_1 - \beta_1, \alpha_2 - \beta_2, \dots, \alpha_n - \beta_n)$$

is positive.

As we did for the lexicographical order, let us have a look at a few examples for $<_{\text{grevlex}}$:

- $y <_{\text{grevlex}} x$ since $(0, 1, 0) - (1, 0, 0) = (-1, 1, 0)$,
- $x <_{\text{grevlex}} y^2$ since $\deg(y^2) = 2 > 1 = \deg(x)$,

- $xz <_{\text{grevlex}} y^2$ since $\deg(xz) = \deg(y^2) = 2$, and $(1,0,1) - (0,2,0) = (1,-2,\mathbf{1})$,
- $xz <_{\text{grevlex}} xy$ since $\deg(xz) = \deg(xy) = 2$, and $(1,0,1) - (1,1,0) = (0,-1,\mathbf{1})$.

Overall, from the smallest to the greatest monomial (of degree less or equal to two) over $\mathbb{K}[x,y,z]$ with $x > y > z$ with respect to the lexicographical order, one has

$$1, z, z^2, y, yz, y^2, x, xz, xy, x^2,$$

and for the graded lexicographical order

$$1, z, y, x, z^2, yz, xz, y^2, xy, x^2.$$

Using an admissible monomial order, it is natural to define the leading monomial, coefficient, and term of a multivariate polynomial:

Definition 28. *Let $<$ be an admissible monomial order on a polynomial ring $\mathbb{K}[x_1, x_2, \ldots, x_n]$, and let P be a polynomial in $\mathbb{K}[x_1, x_2, \ldots, x_n]$.*

We will denote by

- $\mathrm{LM}(P)$ *the leading monomial of P, that is to say, the greatest, with respect to $<$, monomials that appear in P,*
- $\mathrm{LC}(P)$ *the leading coefficient of P, that is to say, the coefficient of the monomial* $\mathrm{LM}(P)$*,*
- $\mathrm{LT}(P)$ *the leading term of P, that is to say,* $\mathrm{LC}(P)\,\mathrm{LM}(P)$.

Let m_1, m_2 be two monomials in $\mathbb{K}[x_1, x_2, \ldots, x_n]$, we define $\mathrm{LCM}(m_1, m_2)$ *as the least common multiple of m_1 and m_2; that is to say, the monomial with smallest degree which is a multiple of both m_1 and m_2.*

We naturally extend the degree of a monomial to the degree of a polynomial in $\mathbb{K}[x_1, x_2, \ldots, x_n]$: $\deg(P)$ *is simply the maximum degree of the monomials of P. It is sometimes referred to as the* total degree.

Note that the notions of LCM and the total degree, see Definition 28, do not depend on the choice of any admissible monomial order. For instance the total degree of $x + y^3 \in \mathbb{K}[x,y]$ is always 3, even if its leading term might change according to the order one chooses.

Now that we have defined admissible orders on monomials together with Definition 28, we can define the multivariate division, see Definition 29.

Definition 29 (Multivariate division. [CLO92], Theorem 3, page 64)**.** *Let $<$ be an admissible monomial order over $\mathbb{K}[x_1, x_2, \ldots, x_n]$, and let $F = \{f_1, f_2, \ldots, f_s\}$ be an ordered s-tuple of polynomials in $\mathbb{K}[x_1, x_2, \ldots, x_n]$.*

Then every $f \in \mathbb{K}[x_1, x_2, \ldots, x_n]$ can be written as

$$f = q_1 f_1 + q_2 f_2 + \ldots + q_s f_s + r,$$

where $q_i, r \in \mathbb{K}[x_1, x_2, \ldots, x_n]$, and either $r = 0$ or r is a linear combination, with coefficients in $\mathbb{K}$, of monomials, none of which is divisible by any of $\mathrm{LT}(f_1)$, $\mathrm{LT}(f_2)$, $\ldots$, $\mathrm{LT}(f_s)$.

We call r a reminder *of f on division by F, we will denote it $\overline{f}^F$.*

Remark 7. *It worth noticing that it is not unique since it heavily depends on the order of the f_i's in F. In fact, $\overline{f}^F$ is unique if, and only if, if F is a Gröbner basis of the ideal $< f_1, f_2, \ldots, f_s >$.*

2.3.1.3 Gröbner basis: existence and unicity

Gröbner bases, see Definition 30, were introduced by Buchberger in 1965 in his PhD thesis [Buc65]. He named them as a tribute to his PhD advisor W. Gröbner.

Given a set $S \subseteq \mathbb{K}[x_1, x_2, \ldots, x_n]$, in what follows, we denote by $\mathrm{LM}(S)$ the set of leading monomials of polynomials in S, more precisely:

$$\mathrm{LM}(S) := \{\mathrm{LM}(s),\ \forall\ s \in S\}\,.$$

As mentioned above, the affine variety of an ideal does not depend on the choice of the ideal's basis. A property of Gröbner bases is that they enable one to efficiently compute the solutions, *i.e.*, the affine variety, associated to an ideal, especially when it is zero-dimensional.

Definition 30 (Gröbner basis.)**.** *Let I be an ideal of $\mathbb{K}[x_1, x_2, \ldots, x_n]$, and let $<$ be an admissible monomial order.*

A finite subset $G \subset I$ is a Gröbner basis of I for $<$ if

$$\langle \mathrm{LM}(G) \rangle = \langle \mathrm{LM}(I) \rangle\,.$$

Note that a Gröbner basis of I is not unique; when we speak about *the* Gröbner basis of an ideal, we refer to its reduced Gröbner basis, see Definition 31.

Definition 31 (Reduced Gröbner basis.)**.** *Let I be an ideal of $\mathbb{K}[x_1, x_2, \ldots, x_n]$, and let $<$ be an admissible monomial order. A finite subset $G \subset I$ is a reduced Gröbner basis of I for $<$ if:*

- $\mathrm{LC}(g) = 1$ *for all* $g \in G$*, and*
- *for all* $g \in G$*, none of the monomials of* g *belongs to* $\langle \mathrm{LT}(G \backslash g) \rangle$*.*

Let I be an ideal of $\mathbb{K}[x_1, x_2, \ldots, x_n]$, and let $<$ be an admissible monomial order. A primordial result in the theory of Gröbner bases is that it is always possible to compute a Gröbner basis of I for $<$, for instance using Buchberger's algorithm [Buc65].

Moreover, we have the following result:

Proposition 3 ([Bar04], Proposition 1.2.7, page 8)**.** *Let $I \neq \{0\}$ be an ideal of $\mathbb{K}[x_1, x_2, \ldots, x_n]$, and let $<$ be an admissible monomial order.*

Then I admits a unique reduced Gröbner basis G for $<$, and G is a basis of I. In particular,

$$G = \{1\} \iff V_{\overline{\mathbb{K}}}(I) = \varnothing.$$

2.3.2 Computing Gröbner bases

Historically, the first algorithm to compute a Gröbner basis was described by Buchberger in [Buc65].

It has been drastically improved by Faugère who proposed F4 in 1999 [Fau99], and then F5 in 2002 [Fau02]. Approximately at the same time, Courtois described the algorithm XL in [CKPS00].

2.3.2.1 Macaulay matrices and linear algebra

In a few words, Buchberger's algorithm computes a Gröbner basis of an ideal by computing, and then reducing, S-polynomials, see Definition 32.

Definition 32 (S-polynomial)**.** *Let $<$ be an admissible monomial order, let f_1, f_2 be two non-zero polynomials in $\mathbb{K}[x_1, x_2, \ldots, x_n]$, the* S-polynomial *of f_1 and f_2 is defined by*

$$\mathrm{Spoly}(f_1, f_2) := \frac{\mathrm{LCM}(\mathrm{LM}(f_1), \mathrm{LM}(f_2))}{\mathrm{LT}(f_1)} f_1 - \frac{\mathrm{LCM}(\mathrm{LM}(f_1), \mathrm{LM}(f_2))}{\mathrm{LT}(f_2)} f_2.$$

For example, relatively to the grevlex order induced by $x > y > z$ over $\mathbb{F}_3[x, y, z]$:

$$\begin{aligned}\text{Spoly}(xy + xz + 1, 2x^2 + y) &= \frac{x^2y}{xy}(xy + xz + 1) - \frac{x^2y}{2x^2}(2x^2 + y) \\ &= (x)(xy + xz + 1) - (2y)(2x^2 + y) \\ &= x^2y + x^2z + x - (x^2y + 2y^2) \\ &= x^2z + y^2 + x.\end{aligned}$$

Note that according to Definition 32, one has that

$$\text{Spoly}(2x^2 + y, xy + xz + 1) = 2x^2z + 2y^2 + 2x,$$

so the S-polynomial is unique up to sign; it does not matter though, since the only role of an S-polynomial is to cancel the leading terms.

A key observation is that the computation of S-polynomials, and the computation of their reminder on multivariate division by a set of polynomials, can be done by computing a row echelon form of a particular matrix. Roughly, it is a matrix whose entries are the coefficients of the polynomials multiplied by monomials, and whose columns are indexed by the monomials, usually sorted using an admissible graded monomial order.

This matrix is called a *Macaulay matrix*, and it is denoted $\mathcal{M}_d^{ac}$ for homogeneous systems, or $\mathcal{M}_{\leq d}^{ac}$ for affine systems. The index d corresponds to the maximum degree of the polynomials that are written in the Macaulay matrix.

From this observation, in 1983, Lazard published a seminal paper [Laz83]: in a few words, computing a Gröbner basis can be done using linear algebra, namely performing Gaussian elimination on Macaulay matrices up to a large enough degree d.

All the aforementioned algorithms, namely F4, F5, and XL, such as their variants, are built upon this result, that is why we commonly say that these algorithms are *linear algebra-based algorithms*, see for instance [CG23].

For more information about linear algebra-based algorithms to compute Gröbner bases, the reader may refer to the original papers [Fau99, Fau02, CKPS00] together with, for instance, [Bar04, Spa12, CG23, CLO92].

2.3.2.2 Direct linearization

One can see the aforementioned Gröbner basis algorithms as refined, sophisticated, advanced, *linearization* algorithms.

Indeed, what we call linearization can be described very easily. Given a system $\{f_1, f_2, \ldots, f_m\} \subset \mathbb{F}_q[x_1, x_2, \ldots, x_n]$ where all the polynomials are homogeneous and of degree d, if one looks for an element in the right kernel of its Macaulay matrix $\mathcal{M}_d^{ac}$, one gets values in $\mathbb{F}_q$ for the monomials such that $f_1 = f_2 = \ldots = f_m = 0$.

This is referred to as *linearization techniques* since it relies on considering each monomial as a new variable, and then solving the associated linear system, hence getting possible values for each monomial.

Remark 8. *It is worth noting that a solution to a system of algebraic equations yields a vector in the right kernel of its Macaulay matrix, but the converse is false. More precisely, there could be values of the monomials for which there does not exist any consistent affectation of the variables resulting in these values.*

Let us see this on a toy example over $\mathbb{F}_3[x_1, x_2, x_3]$ with the lexicographical monomial order:

$$\begin{cases} f_1 = x_1^2 + x_2^2 + x_2x_3 + 2x_3^2 \\ f_2 = x_1x_2 + x_2^2 + x_2x_3 + x_3^2 \\ f_3 = x_1^2 + x_1x_2 + x_1x_3 + x_2x_3 \end{cases}.$$

Since this system is made of homogeneous polynomials, if $(x_0, y_0, z_0) \in \mathbb{F}_3^3$ is a solution, then for every $\lambda \in \mathbb{F}_3$, so will $\lambda(x_0, y_0, z_0)$. Thus, $(0, 0, 0) \in \mathbb{F}_3^3$ is solution, but we want to find the other non-trivial solutions.

The Macaulay matrix of this system of homogeneous polynomials at degree 3 is given by the following 9×10 matrix:

$\mathcal{M}_3^{ac} =$

	x_1^3	$x_1^2x_2$	$x_1^2x_3$	$x_1x_2^2$	$x_1x_2x_3$	$x_1x_3^2$	x_2^3	$x_2^2x_3$	$x_2x_3^2$	x_3^3
x_1f_1	1	0	0	1	1	2	0	0	0	0
x_1f_2	0	1	0	1	1	1	0	0	0	0
x_1f_3	1	1	1	0	1	0	0	0	0	0
x_2f_1	0	1	0	0	0	0	1	1	2	0
x_2f_2	0	0	0	1	0	0	1	1	1	0
x_2f_3	0	1	0	1	1	0	0	1	0	0
x_3f_1	0	0	1	0	0	0	0	1	1	2
x_3f_2	0	0	0	0	1	0	0	1	1	1
x_3f_3	0	0	1	0	1	1	0	0	1	0

whose rank is exactly 9. Thus, it has a kernel of dimension 1, spanned by the following vector:

$$(1, 2, 1, 1, 2, 1, 2, 1, 2, 1)^\intercal.$$

This vector gives that $x_1x_2x_3 = 2$, thus none of the variables is zero; using the aforementioned fact about the homogeneity, we can then set $x_3 = 1$. This gives

- $x_2 = 2$ since $x_2x_3^2 = 2$,
- $x_1 = 1$ since $x_1x_3^2 = 1$.

Thus, we found the non-trivial solution $(1, 2, 1)$ which, by homogeneity, gives $(2, 1, 2)$, and the zero vector we already had.

The aforementioned technique can be generalized to non-homogeneous systems of algebraic equations as long as all the constant terms of the polynomials are zero.

In the case where 1 would index a column of the Macaulay matrix, that is to say, if some polynomials had a non-trivial constant term, a solution to the system could still be found by computing a right kernel.

More precisely, if $\mathcal{M}^{ac}_{\leq d}$ is an $M \times N$ Macaulay matrix of a system of algebraic equations at degree d, and such that 1 indexes its last column; the values of the $N-1$ monomials, distinct from 1, will be given by the first $N-1$ entries of $\boldsymbol{v} \in \ker_R(\mathcal{M}^{ac}_{\leq d})$ such that

$$\boldsymbol{v} = (v_1, v_2, \ldots, v_{N-1}, 1).$$

The last entry of $\boldsymbol{v}$ corresponds to the value of the monomial of degree 0, *i.e.*, 1.

This corresponds to the straightforward reduction from solving a linear system $\boldsymbol{A}\boldsymbol{x}^\intercal = \boldsymbol{b}^\intercal$, i.e. with a non-zero right-hand side, to computing $\ker_R((\boldsymbol{A}|\boldsymbol{b}^\intercal))$.

Clearly, these classical linearization techniques have two main limitations when used in order to find a solution to a system of algebraic equations:

- The dimension of the kernel may be large, making it computationally hard for one to recover an actual solution from the values of the system's monomials.
- Sometimes, it is not straightforward to go from the values of the monomials to an actual solution.

2.3.2.3 Complexity

This section is dedicated to the complexity of finding a vector belonging to the right kernel of a matrix.

Let M, N be positive integers, let $\boldsymbol{M}$ be an $M \times N$ matrix with entries in $\mathbb{F}_q$, the complexity, in terms of elementary operations in $\mathbb{F}_q$, to find a vector in its right kernel is given by

$$\mathcal{O}\left(\text{rank}(\boldsymbol{M})^{\omega-2} \times M \times N\right) \subset \mathcal{O}\left(M \times N^{\omega-1}\right), \tag{2.4}$$

where $2 < \omega \leq 3$ is the linear algebra constant, [BH74, Sto00, vzGG13].

In this document, we will consider Strassen's algorithm which yields

$$\omega = \log_2(7) \approx 2.807.$$

However, if the matrix is sparse, that is to say, that it only has τ non-zero terms per row, it is sometimes worth using Wiedemann's algorithm [Wie86] whose complexity is given by

$$\mathcal{O}\left(\tau \times M \times N\right).$$

For a more thorough analysis of Wiedemann's algorithm and its improvements, the reader may refer to [Cop94, Kal95, Vil97].

Remark 9. *One notices that if $\tau = \mathcal{O}(1)$, then the complexity of Wiedemann's algorithm is the same as the one given by Equation (2.4) using $\omega = 2$.*

2.3.2.4 A non-exhaustive list of potential issues

In this introduction about computing Gröbner bases through direct linearization, we did not address a lot of points which make algebraic attacks challenging to perform and study. However, here is a glimpse of these questions one has to consider while looking for an efficient algebraic attack:

- Most importantly: what is the value of d one needs to go up to? Or what is the rank of the Macaulay matrix at a given step degree?
- Should we linearize using a new variable per monomial or could we go further?
- Should we consider homogenized systems? Or instead their homogeneous components of highest degree?

- In order to build the Macaulay matrix, do we multiply by every monomials or by a subset of them?
- Do we add the field equations for $\mathbb{F}_q$, given by

$$\left(x_i^q - x_i\right)_i,$$

  to the initial system?
- Do we use Wiedemann or Strassen's algorithm in order to solve the linear system?
- Is it worth considering only a subset of the rows of the Macaulay matrix at degree d?

Naturally, for the algebraic attacks described in Sections 3.1 and 3.2, we will address, as much as we can, the above items.

2.3.3 Maximal minors and Plücker coordinates

We will see in this document that it is sometimes very beneficial to consider the maximal minors of a matrix as new variables in the aforementioned linearization process.

This corresponds to using the so-called Plücker coordinates, as in [BBC+20, BBB+22].

For more information about determinantal systems, the reader may refer to the book by Bruns and Vetter [BV06], or to the PhD thesis of Spaenlehauer [Spa12].

Let us start with a toy example: we want to compute a basis matrix of the row space of

$$\boldsymbol{M} := \begin{pmatrix} 2 & 2 & 2 & 1 \\ 1 & 2 & 0 & 1 \\ 1 & 1 & 1 & 2 \end{pmatrix} \in \mathbb{F}_3^{3\times 4}.$$

Let us assume that we know that this matrix has rank 2, so the variables will be the entries of the following matrix:

$$\boldsymbol{C} := \begin{pmatrix} x_1 & x_2 & x_3 & x_4 \\ x_5 & x_6 & x_7 & x_8 \end{pmatrix} \in \mathbb{F}_q[x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8]^{2\times 4}.$$

A natural modeling is to cancel all the maximal minors, *i.e.*, 3×3, of

$$\begin{pmatrix} M_{\{i\},*} \\ \boldsymbol{C} \end{pmatrix} \in \mathbb{F}_q[x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8]^{3\times 4}, \quad \forall i \in \{1, 2, 3\}.$$

This system of algebraic equations will contain

$$3\binom{4}{3} = 12 \quad \text{equations.}$$

For example, one of these equations is:

$$\left| \begin{pmatrix} 2 & 2 & 2 & 1 \\ x_1 & x_2 & x_3 & x_4 \\ x_5 & x_6 & x_7 & x_8 \end{pmatrix} \right|_{*,\{1,2,3\}} = 2(x_2x_7 - x_3x_6) - 2(x_1x_7 - x_3x_5) + 2(x_1x_6 - x_2x_5). \quad (2.5)$$

We will not describe the full system nor the associated Gröbner basis; however, here are its properties: for the lexicographical order, the Gröbner basis of the ideal generated by this system to which one appends the field equations $x_i^3 - x_i$ contains 29 polynomials of degree 2 up to 5.

Most importantly, there are 369 solutions, all in $\mathbb{F}_3^8$ since we added the field equations.

This large number of solutions comes from the fact that we did not add a constraint on the rank of $\boldsymbol{C}$, thus any matrix in $\mathbb{F}_q^{2\times 4}$ of rank less or equal to 1 is solution, and there are

$$\frac{(q^4 - 1)(q^2 - 1)}{(q - 1)} + 1 = 321$$

such matrices. On top of these solutions, there are the ones of interest to us, namely the

$$(q^2 - 1)(q^2 - q) = 48$$

solutions corresponding to bases of the row space of $\boldsymbol{C}$.

The total adds up to $321 + 48 = 369$.

Now, let us consider the same toy example, but we replace each maximal minor of $\boldsymbol{C}$ involving $T \subset \{1..4\}$ columns as a new variable that we denote c_T. Note that, for the sake of clarity, we might omit the curly brackets "{}" symbols around the set T.

Thus there are $\binom{4}{2} = 6$ variables c_T's, namely $c_{1,2}, c_{1,3}, c_{1,4}, c_{2,3}, c_{2,4}, c_{3,4}$, and we now deal with the following polynomial ring $\mathbb{F}_3[c_{1,2}, c_{1,3}, c_{1,4}, c_{2,3}, c_{2,4}, c_{3,4}]$.

With this modeling, the above Equation (2.5) becomes:

$$2c_{2,3} - 2c_{1,3} + 2c_{1,2} \in \mathbb{F}_3[c_{1,2}, c_{1,3}, c_{1,4}, c_{2,3}, c_{2,4}, c_{3,4}].$$

This new system in the polynomial ring $\mathbb{F}_3[c_{1,2}, c_{1,3}, c_{1,4}, c_{2,3}, c_{2,4}, c_{3,4}]$ still contains 12 equations, however they are now linear.

The Gröbner basis of the ideal they generate is given by the following polynomials:

$$\begin{cases} c_{1,2} + 2c_{3,4} \\ c_{1,3} + c_{3,4} \\ c_{1,4} + c_{3,4} \\ c_{2,3} + 2c_{3,4} \\ c_{2,4} \end{cases},$$

whose solutions are given by the 3 points in $\mathbb{F}_3^6$:

$$\begin{matrix} (0,0,0,0,0,0) \\ (1,2,2,1,0,1) \\ (2,1,1,2,0,2) \end{matrix} . \tag{2.6}$$

It is immediate to notice that the 2 last points give the solutions we are looking for, in fact they yield a full rank matrix.

Roughly, the first zero solution embeds the 321 aforementioned ones, and the two last ones embed the remaining 48 of interest to us.

The process of going from a solution point to the actual entries of the matrix will be very important in the rest of this document.

Basically, one just picks a non-zero maximal minor, let us say $c_{1,2}$ and set it to 1. This is authorized as long as this minor is non-singular since all multiples of the matrix $\boldsymbol{C}$ by a non-singular 2×2 matrix is still a basis.

The key observation is that some particular maximal minors of the following matrix give its entries; more precisely, if

$$\boldsymbol{U} := \begin{pmatrix} 1 & 0 & x & y \\ 0 & 1 & z & t \end{pmatrix},$$

then

$$\begin{cases} x = -|\boldsymbol{U}|_{*,\{2,3\}} \\ y = -|\boldsymbol{U}|_{*,\{2,4\}} \\ z = |\boldsymbol{U}|_{*,\{1,3\}} \\ t = |\boldsymbol{U}|_{*,\{1,4\}} \end{cases}.$$

Thus, since we set $c_{1,2}$ to 1, we can consider that the first 2×2 block of $\boldsymbol{C}$ contains the identity matrix, then the values of the 4 remaining entries, $\boldsymbol{C}_{3,1}$, $\boldsymbol{C}_{3,2}$, $\boldsymbol{C}_{4,1}$, $\boldsymbol{C}_{4,2}$, are given, respectively, by the values of $-c_{2,3}$, $-c_{2,4}$, $c_{1,3}$, $c_{1,4}$ taken from the second point in Equation (2.6).

This process can be generalized very easily to arbitrary large matrices: as long as one knows the values of all the maximal minors, it suffices to set one to 1, then to put an identity block at its associated spot, and finally to get the remaining entries using some wisely chosen minors.

The aforementioned algebraic modeling is absolutely terrible to find a basis of the row space of a matrix!

However, it is a good illustration of the differences between the two approaches; namely quadratic system with 369 solutions in one case, and linear system with 3 solutions in the other.

This gives one a glimpse of the power of the so-called Plücker coordinates when used for a system which admits solutions up to multiplication by non-singular matrices, typically these are systems whose solutions are given by vector spaces.

Last but not least, let us describe formally what Plücker coordinates are. Roughly, Plücker coordinates are given by a map between vector spaces and the maximal minors of one of their basis matrix, see Definition 33.

It is readily seen that if $\boldsymbol{C} \in \mathbb{F}_q^{r \times n}$ is a basis matrix of a vector space $W \in \mathbb{F}_q^n$ of dimension r, then so is $\boldsymbol{AC}$ for any $\boldsymbol{A} \in \mathrm{GL}_r(\mathbb{F}_q)$. The effect of multiplying $\boldsymbol{C}$ by a non-singular matrix $\boldsymbol{A}$ leads to multiplying all its maximal minors by a non-zero element of $\mathbb{F}_q$, namely by $|\boldsymbol{A}|$. This is why the following definition uses projective space.

Definition 33 (Plücker map and coordinates)**.** *Let $N := \binom{n}{r} - 1$, we denote by $\mathbb{P}^N(\mathbb{F}_q)$ the projective space $\mathbb{P}(\mathbb{F}_q^{N+1})$. Let $T_1, T_2, \ldots, T_{N+1}$ be the sets in $\{1..n\}$ with r elements.*

The Plücker map p *is defined as follow:*

$$p\colon \{W \subset \mathbb{F}_q^n \ :\ \dim(W) = r\} \longrightarrow \mathbb{P}^N(\mathbb{F}_q)$$
$$W \longmapsto (|\boldsymbol{C}|_{*,T_i})_{i\in\{1..N+1\}}.$$

where $\boldsymbol{C} \in \mathbb{F}_q^{r\times n}$ *is a matrix whose rows span the vector space* W*.*

We call Plücker coordinates *of a vector space* W *the homogeneous coordinates* $p(W)$*.*

Notice that the Plücker map p is *well-defined* since if two distinct matrices $\boldsymbol{C}_1$ and $\boldsymbol{C}_2$ contain basis of a vector space W, then

$$p(W) = (|\boldsymbol{C}_1|_{*,T_i})_{i\in\{1..N+1\}} = (|\boldsymbol{C}_2|_{*,T_i})_{i\in\{1..N+1\}}.$$

2.4 Provable Security Basics

Introduced in the eighties with the seminal papers by Bellare, Goldwasser, Micali, Rackoff, and Rogaway [GM82, GMR85, BR93], provable security is a powerful tool which allows designers of primitives and protocols to mathematically prove the security guaranteed by their schemes; and even exactly quantify the loss of security. It enables cryptographic constructions that follow the paradigm of *security by design*.

In this document, we will not go too deep in the notions of provable security, the interested reader may refer to the very complete book by Katz and Lindell, namely "Introduction to modern cryptography" [KL20].

We recall that an encryption scheme is made of 4 algorithms: Setup, KeyGen, Encrypt, Decrypt; and a signature scheme is also made of 4 algorithms: Setup, KeyGen, Sign, Verif. When there is no ambiguity, we might omit the Setup algorithm.

In provable security, we can define security based on *games*, that is to say, that the adversary plays a game against the challenger; or we can use *simulation* based definitions in which the goal is to distinguish between real world (with an adversary) and an ideal world (with a simulator).
In this thesis, we focus on the more prevalent game-based definitions.

In what follows, we first define **IND-CPA** security (for public key encryption schemes) and **EUF-CMA** security (for signature schemes).

Then, we will present the Fiat-Shamir authentication protocol and a mechanism to turn it into a signature scheme, also due to Fiat and Shamir.

Our proofs are in the *Random Oracle security Model* (ROM) due to Bellare and Rogaway [BR93], see the end of Section 2.4.3. This model has some flaws as noted in [CGH04], basically because real world hash functions are not random oracles; however, it is usually sufficient to assume that hash functions behave like random oracles.

However, most schemes that are secure in the ROM provide a reasonable level of security in practice, when instantiated with "good" hash functions, such as SHA256 or Keccak.

2.4.1 IND-CPA and EUF-CMA

2.4.1.1 IND-CPA

Indistinguishability under Chosen-Plaintext Attacks, denoted **IND-CPA**, is a security property for encryption schemes [GM82]; **IND-CPA**-secure encryption schemes reveal no information about the plaintext when given the corresponding ciphertext.

More precisely, an adversary cannot distinguish which of two adversarially-chosen messages is hidden in a given ciphertext.

IND-CPA security is formalized in Definition 34 where the event "$\mathcal{A}$ wins" refers to the security game depicted in Figure 2.1. Recall that the parameter λ gives the security level.

Definition 34 (**IND-CPA** security)**.** *A public key encryption scheme $\mathcal{E}$ is said to be* **IND-CPA***-secure if for any probabilistic polynomial time (PPT) algorithm $\mathcal{A}$, the advantage of $\mathcal{A}$ to win the game $Exp_{\mathcal{E}}^{\textbf{IND-CPA}}$ depicted in Figure 2.1, denoted* $\mathrm{Adv}(\mathcal{A})$*, is smaller than ε, where ε is negligible, and*

$$\mathrm{Adv}(\mathcal{A}) := \left| \mathbb{P}\left(\mathcal{A} \textit{ wins}\right) - \frac{1}{2} \right|.$$

$\underline{\text{Exp}_{\mathcal{E}}^{\textbf{IND-CPA}}(1^\lambda)}$:

$(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$

$b \xleftarrow{\$} \{0,1\}$

$(m_0, m_1) \leftarrow \mathcal{A}\left(1^\lambda, \mathsf{pk}\right)$, s.t. $\text{len}(m_0) = \text{len}(m_1)$

$c^* \leftarrow \mathsf{Encrypt}(\mathsf{pk}, m_b)$

$b' \leftarrow \mathcal{A}\left(1^\lambda, \mathsf{pk}, c^*, m_0, m_1\right)$

The adversary $\mathcal{A}$ wins the game if, and only if, $b = b'$.

Figure 2.1: **IND-CPA** security game for a public key encryption scheme $\mathcal{E}$.

2.4.1.2 EUF-CMA

Existential Unforgeability under Chosen-Message Attacks, denoted **EUF-CMA**, is a security property for signature schemes [GMR85]; **EUF-CMA**-secure signature schemes guarantee that it is impossible for an adversary to forge a valid signature not knowing the secret key, even if she is given access to a signature oracle.

For the sake of simplicity, in this document we always refer to public and secret keys even though signature schemes technically feature signing and verification keys.

More formally, it can also be formalized through the security game depicted in Figure 2.2, where $\mathcal{O}_{\text{sign}}(\cdot)$ is a signature oracle which returns a valid signature $\mathcal{O}_{\text{sign}}(m) := \mathsf{Sign}(\mathsf{sk}, m)$ when given a message m. Note that after each request on a message m, the oracle adds m to a set of signed messages SM.

Definition 35 (**EUF-CMA** security)**.** *A signature scheme $\mathcal{S}$ is* **EUF-CMA***-secure if for any probabilistic polynomial time (PPT) algorithm $\mathcal{A}$, the advantage of $\mathcal{A}$ to win the game $Exp_{\mathcal{S}}^{\textbf{EUF-CMA}}$ depicted in Figure 2.2, denoted* $\text{Adv}(\mathcal{A})$*, is smaller than ε, where ε is negligible, and*

$$\text{Adv}(\mathcal{A}) := \mathbb{P}\left(\mathcal{A} \textit{ wins}\right).$$

In other words, to win the **EUF-CMA** security game, an adversary has to be able to forge a signature for a fresh message m^*, different from all the queries she made to the signing oracle (stored in SM).

$\underline{\mathrm{Exp}_{\mathcal{S}}^{\textbf{EUF-CMA}}(1^\lambda)}$:

$(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$

$(m^*, \sigma^*) \leftarrow \mathcal{A}\left(1^\lambda, \mathsf{pk}, \mathcal{O}_{\mathrm{sign}}(\cdot)\right)$

$b \leftarrow \mathsf{Verif}(\mathsf{pk}, m^*, \sigma^*)$

`IF` $m^* \in SM$ `THEN return` 0

`ELSE return` b.

The adversary $\mathcal{A}$ wins the game if, and only if, $b = 1$ is returned.

Figure 2.2: **EUF-CMA** security game for a signature scheme $\mathcal{S}$.

2.4.2 The Fiat-Shamir authentication protocol

We recall that an authentication protocol allows a *prover* to prove to a *verifier* that she knows a secret associated to her public key.

The verifier ultimately computes a validity bit depending on whether it deems the prover legitimate or not; by contrast, in an identification protocol, the verifier also outputs the identity of the prover.

Remark 10. *The Fiat-Shamir protocol which we present below is often defined in the literature as an identification protocol, as the public key corresponds to an identity; however, in this document, we choose to refer to it as an authentication protocol, as the verifier output is a validity bit.*

The Fiat-Shamir authentication scheme [FS87] is depicted in Figure 2.3.

This protocol has much stronger security properties than standard authentication protocol. Indeed, the scheme is an *interactive zero-knowledge proof of knowledge* and it allows the prover to prove knowledge of her secret without revealing any information about it.

More formally, interactive zero-knowledge proofs of knowledge have to guarantee three properties; in what follows we briefly describe these properties. Then we give a sketch of the proof that the Fiat-Shamir authentication scheme depicted in Figure 2.3 provides all of them:

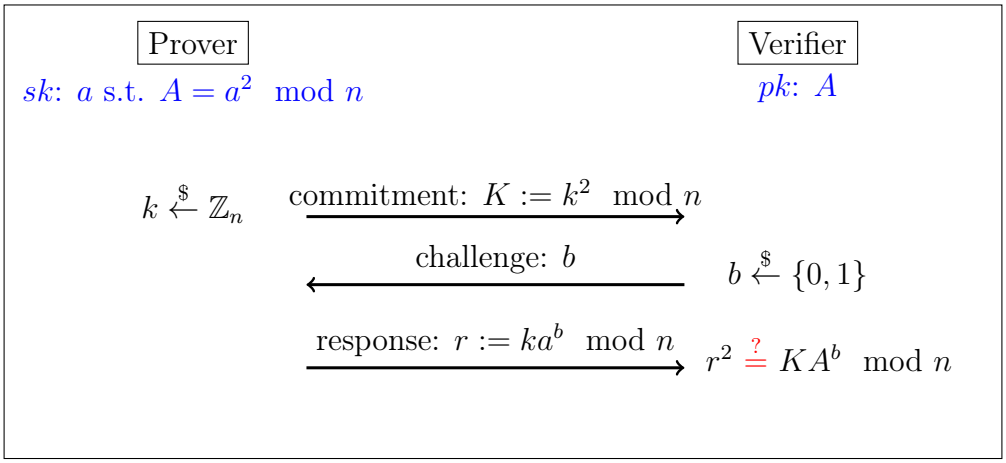


Figure 2.3: Fiat-Shamir authentication scheme.

1. **Completeness.** *Does it work?* An honest verifier following the protocol will always deem an honest prover legitimate.
2. **Soundness.** *How much can an adversary cheat?* A cheater, that is to say, a prover who does not know the secret but wants to prove otherwise, will be able to convince the verifier only with a negligible probability.
3. **Zero-Knowledge.** *Does it leak information about the secret?* The verifier will learn nothing else than the fact that the prover knows the secret, that is to say, no information will leak about the secret during the protocol.

The Fiat-Shamir authentication scheme depicted in Figure 2.3 relies on the hardness of finding square roots, also called quadratic residues, modulo an integer n which is an RSA-modulus, *i.e.*, the product of two distinct large primes p and q.

Assuming the hardness of this problem, the Fiat-Shamir authentication protocol guarantees the three aforementioned properties.

Completeness. If the prover is honest, she sends ka^b, and no matter the value of b, one has that

$$\left(ka^b\right)^2 = k^2 \left(a^2\right)^b = KA^b$$

thanks to the commutativity in $\mathbb{Z}_n$.

Soundness. A cheater, *i.e.*, a prover who does not know the secret a, but wants to prove she does, can predict one of the challenges very easily. That is to say that the cheater picks k and sends $K := k^2$, and then hope to get the challenge $b = 0$.

If the verifier sends $b = 0$, then the cheater responds with $r := k$.

We claim that by using this strategy, the cheater can win and be authenticated as a legitimate prover. Indeed, the verifier expects to receive r such that $r^2 = KA^b = K$, which is the case for the presented strategy.

Since $b = 0$ happens with probability $1/2$, the cheater wins about half the time.

To prove that this trivial winning probability is actually no more than $1/2$, we need to check if a cheater has a non-negligible probability to win if $b = 1$. This is the point of Proposition 4.

Proposition 4. *If an adversary can produce a valid response for both challenges in the Fiat-Shamir authentication protocol, then she knows how to compute a square root of A modulo n.*

Proof. If an adversary can produce a valid response for both challenges, then she can produce y_1 and y_2 such that :

$$\begin{cases} y_1^2 = KA \mod n \\ y_2^2 = K \mod n \end{cases}.$$

Note this does not necessarily mean that $y_1 = ka$ or $y_2 = k$, just that the cheater knows one of the square roots for both.

Then the cheater can easily compute $y_1 y_2^{-1}$ whose square is given by

$$\left(y_1 y_2^{-1}\right)^2 = y_1^2 y_2^{-2} = KAK^{-1} = A \mod n,$$

thus the cheater does know a square root of A. □

Since anticipating both challenges is equivalent to computing a square root modulo n, the soundness of the Fiat-Shamir authentication protocol is $1/2 + \varepsilon$ for a negligible value of ε.

Recall that soundness requires an at most negligible winning probability for the cheater, see above; hence, in practice, to render the Fiat-Shamir protocol sound, one repeats the scheme depicted in Figure 2.3 multiple times in parallel. The soundness then becomes $\frac{1}{2^N}$ where N is the number of repetitions.

Zero-Knowledge. The Zero-Knowledge part is straightforward: the only communication which involves the secret is the response, sent when the challenge $b = 1$,

$$r := ka \mod n.$$

However, since k is chosen uniformly at random in $\mathbb{Z}_n^\times$, it is clear that ka is uniformly distributed in $\mathbb{Z}_n^\times$. This means that the verifier will not learn any information about the secret a from ka.

Moreover, as the protocol executions are independent, the verifier will learn no more from executing it multiple times then it would from executing it once.

2.4.3 The Fiat-Shamir heuristic

The Fiat-Shamir authentication scheme can be turned into a signature scheme by using the Fiat-Shamir heuristic, also described in [FS87].

In what follows, we briefly recall how the Fiat-Shamir heuristic works.

Suppose that we are executing N parallel runs of the Fiat-Shamir protocol; to understand the need for multiple executions, see the soundness argument in the previous section. The prover begins by sending N parallel commitments

$$(K_1, K_2, \ldots, K_N) \in (\mathbb{Z}_n)^N$$

to N independent k_i values.

The verifier responds with N independent challenge bits $(b_i)_i$. Finally the prover must produce N responses

$$(R_1, R_2, \ldots, R_N) \in (\mathbb{Z}_n)^N,$$

such that,

$$\forall i \in \{1..N\}, \quad R_i^2 = K_i A^{b_i}.$$

In order to render this protocol non-interactive, the prover must be able to simulate the verifier's challenges. The Fiat-Shamir heuristic allows the prover to use a hash function, idealized as a random oracle, to produce these challenges.

Specifically, the challenges correspond to the first N bits b_i of

$$\mathcal{H}(K_1||K_2||\ldots||K_N||\mathsf{pk})$$

where $\mathcal{H}$ is the hash function, $||$ is the concatenation operator, the K_i's are the commitments, and $\mathsf{pk} = A$ is the public key.

To further turn this scheme into a signature scheme, one must be able to sign messages. Thus, the signature of the message $\boldsymbol{m}$ will be given by

$$\sigma = (K_1||K_2||\dots||K_N, R_1||R_2||\dots||R_N) \tag{2.7}$$

where R_i is computed as

$$\forall i \in \{1..N\}, \quad R_i = k_i a^{b_i},$$

and b_i is the i^{th} bit of

$$\mathcal{H}(K_1||K_2||\dots||K_N||\mathsf{pk}||\boldsymbol{m}).$$

The use of a hash function modeled as random oracle guarantees the unpredictability of the challenge bits (and their high entropy).

The verification process is straightforward, the verifier uses the message and the public key to compute the challenges, then she checks them one by one using the verification process of the Fiat-Shamir protocol.

If at least one response is not valid, then the signature is not valid; conversely, if all responses are valid, so is the signature.

Pointcheval and Stern proved in [PS96] that the Fiat-Shamir heuristic yields signature schemes that are **EUF-CMA**-secure in the ROM.

More generically, any *interactive zero-knowledge proof of knowledge* can be turned into an **EUF-CMA**-secure signature scheme by using this transformation.

The Random Oracle Model (ROM). Intuitively, the Random Oracle Model assumes that hash functions behave like idealized functions

$$\begin{aligned} \mathcal{RO}\colon \{0,1\}^* &\longrightarrow \{0,1\}^N \\ x &\longmapsto \mathcal{RO}(x) \end{aligned}$$

where $N \in \mathbb{N}^+$ is fixed, and such that:

- **Randomness:** when called on a bit string of arbitrary length x, $\mathcal{RO}$ outputs a random N-bit string $\mathcal{RO}$, totally independent from x,
- **Consistency:** multiple calls to $\mathcal{RO}$ on an input x yields the same output.

Remark 11. *Security proofs in the ROM rely on the random oracle behaving like a black box, which has to be queried, either by the adversary or by honest parties in order to receive the response. This is not the case for the use of hash functions, which parties can use freely as a white box.*

Chapter 3

Cryptanalysis results

Contents

3.1 Algebraic Attack against MinRank

In this Section, we present a new algebraic attack against the **MinRank** problem, see Definition 16, namely the SupportMinors attack; it has been published in [BBC+20], and further improved in [BBB+22].

3.1.1 SupportMinors Modeling

Let $\boldsymbol{M}_1, \boldsymbol{M}_2, \ldots, \boldsymbol{M}_K$ be the K matrices of a **MinRank** instance with parameters (q, m, n, K, r). Recall that in this document, we consider only instances for which there exists only one solution; in this case, it means that there is a single non trivial $\mathbb{F}_q$-linear combination of the matrices $\boldsymbol{M}_i$'s which has rank exactly r.

Remark 12. *The fact that the target rank is exactly r is a very light assumption, see for instance the discussions about the unicity of the solution in Section 2.2.4.*

Usually the value r is publicly known, if it is not, one can simply use the incremental approach described at the beginning of Section 3 in [BBB+20].

First of all, there is a very simple result in linear algebra that we will extensively use in this document: an $m \times n$ matrix $\boldsymbol{M}$ of rank r can be factorized into two matrices $\boldsymbol{S} \in \mathbb{F}_q^{m \times r}$ and $\boldsymbol{C} \in \mathbb{F}_q^{r \times n}$ which are both of full rank r.

The matrix $\boldsymbol{S}$ can be seen as basis of the column space of $\boldsymbol{M}$, and the matrix $\boldsymbol{C}$ as the coordinates of each column of $\boldsymbol{M}$ into this basis. Conversely, one can see $\boldsymbol{C}$ as a basis of the row space of $\boldsymbol{M}$.

This factorization is unique if, and only if, one wants $\boldsymbol{S}$ or $\boldsymbol{C}$ to be in reduced echelon form.

Coming back to our **MinRank** instance, using the fact we just mentioned, one can write that

$$\sum_{i=1}^{K} x_i \boldsymbol{M}_i = \boldsymbol{S}\boldsymbol{C} \tag{3.1}$$

since one wants the matrix $\sum_{i=1}^{K} x_i \boldsymbol{M}_i$ to be of small rank r.

Taking $\boldsymbol{S}$ and $\boldsymbol{C}$ as matrices of unknowns, Equation (3.1) gives an algebraic modeling to solve the **MinRank** problem.

We call the entries of $\boldsymbol{S}$ the *support variables*, and the entries of $\boldsymbol{C}$ the *coefficient variables*. Note that in this system, the variables x_i only occur linearly. As such, we will name them the *linear variables*.

For all $j \in \{1..m\}$, let $\boldsymbol{r}_j$ be the j-th row of $\sum_{i=1}^{K} x_i \boldsymbol{M}_i$. Equation (3.1) implies that each row $\boldsymbol{r}_j$ is in the row space of $\boldsymbol{C}$.

Thus, the following $(r+1) \times n$ matrix

$$\begin{pmatrix} \boldsymbol{r}_j \\ \boldsymbol{C} \end{pmatrix}$$

cannot be of full rank. This means that all its maximal minors, i.e. $(r+1) \times (r+1)$ minors, are all equal to 0.

By using cofactor expansion, also called Laplace expansion, for these maximal minors with respect to their first row, one notices that they can be seen as bilinear forms in the x_i's and in the $r \times r$ minors of $\boldsymbol{C}$.

The core of the SupportMinors modeling is then to replace each of these maximal minors of $\boldsymbol{C}$ by a new variable, this linearization process corresponds precisely to using Plücker coordinates as described in Section 2.3.3.

For these new variables, we use the following notation:

Notation 1. *Let $T \subset \{1..n\}$ where $\#T = r$. Let c_T be the maximal minor of $\boldsymbol{C}$ corresponding to the columns of $\boldsymbol{C}$ that belong to T, i.e.*

$$c_T := |\boldsymbol{C}|_{*,T}\,.$$

We can now describe our modeling to solve **MinRank**, namely the SupportMinors algebraic modeling:

Modeling 1 (Support Minors modeling)**.** *For all $j \in \{1 \dots m\}$, let $\boldsymbol{r}_j$ be the j-th row of the matrix*

$$\sum_{i=1}^{K} x_i \boldsymbol{M}_i.$$

We consider the system of bilinear equations, given by canceling the maximal minors of the m matrices $\begin{pmatrix}\boldsymbol{r}_j \\ \boldsymbol{C}\end{pmatrix}$, $j \in \{1 \dots m\}$, i.e.

$$\left\{ f = 0 \;\middle|\; f \in \textbf{MaxMinors}\begin{pmatrix}\boldsymbol{r}_j \\ \boldsymbol{C}\end{pmatrix},\ j \in \{1..m\} \right\}. \tag{3.2}$$

This system contains:

- *$m\binom{n}{r+1}$ bilinear equations with coefficients in $\mathbb{F}_q$,*
- *$K + \binom{n}{r}$ unknowns: $\boldsymbol{x} = (x_1, \cdots, x_K)$ and the c_T's, $T \subset \{1..n\}$ where $\#T = r$.*

We search for the solutions x_i's, and c_T's in $\mathbb{F}_q$.

In this next section, we will describe how we compute the solution of this system in order to get the solution to the **MinRank** instance, and we will give the complexity to do so.

3.1.2 Complexity

3.1.2.1 Initial system ($b = 1$)

First of all, note that the SupportMinors system, see Modeling 1, is far smaller than the one where one would use the coefficients $C_{i,j}$ of the matrix $\boldsymbol{C}$ instead of the c_T's, roughly by a factor $r!$.

In addition to this, this system has degree 2 whereas it would have degree $r+1$ polynomials using the actual maximal minors of $\boldsymbol{C}$.

It is straightforward to see that this system has at most $K\binom{n}{r}$ monomials of degree 2 (of the form $x_i c_T$, i.e. of bi-degree $(1,1)$), if the number of linearly independent equations is equal to the number of monomials minus 1, then one can solve the system by direct linearization, see Section 2.3.2.2.

More precisely, given a **MinRank**(m,n,K,r) instance, if the condition:

$$m\binom{n}{r+1} \geq K\binom{n}{r} - 1, \tag{3.3}$$

is fulfilled, then one expects to solve the system by linearization.

We did a lot of experiments and it appears that, as long as the condition given by Equation (3.3) is fulfilled, the rank of the Macaulay matrix of the SupportMinors system at degree 2 is $K\binom{n}{r} - 1$. Thus, one can get the values of all the monomials in the system by finding a vector in the right kernel of the Macaulay matrix, see Section 2.3.2.2.

However, even knowing this kernel vector, one only has the values of all the monomials of the system, i.e. degree 2 monomials of the form

$$x_i c_T.$$

In order to recover the actual solution to the **MinRank** instance, recall that it means getting the values of the K linear variables x_i, one can follow the procedure given in Section 3.1.2.6.

Remark 13. *The SupportMinors modeling, see Modeling 1, is an algebraic system of bi-degree $(1,1)$ in the x_i's and c_T's variables; since for higher degrees, see Section 3.1.2.2, we write $(b,1), b>1$, the aforementioned system is referred to as the $b=1$ system.*

3.1.2.2 Solving at higher degree ($b>1$)

When the condition given by Equation (3.3) is not fulfilled, i.e. when there are more variables than equations, one wants to increase the number of equations in order to linearize the system.

More precisely, when the condition given by Equation (3.3) is not fulfilled, we can get new equations by multiplying the original degree 2 equations by monomials involving *only* the linear variables, i.e. the x_i's.

By multiplying the initial equations by monomials of degree $b-1$ with $b \geq 2$, one gets a bilinear system of bi-degree $(b, 1)$ in the x_i's and c_T's variables.

It is straightforward that this SupportMinors algebraic system at bi-degree $(b, 1)$ has

$$m\binom{n}{r+1}\binom{K+b-2}{b-1} \tag{3.4}$$

equations containing at most

$$\binom{K+b-1}{b}\binom{n}{r} \tag{3.5}$$

distinct monomials of bi-degree $(b, 1)$.

Remark 14. *All along this section, we suppose that $q > b$, the particular case $q = 2$ is discussed in Section 3.1.2.3.*

However, among these

$$m\binom{n}{r+1}\binom{K+b-2}{b-1}$$

equations at bi-degree $(b, 1)$, there are some linear dependencies; in order to use the linearization techniques described in Section 2.3.2.2, we need to compute the number of *linearly independent* equations, which corresponds to the rank of the Macaulay matrix of the SupportMinors system.

Before going further, we need to recall the definition of a (symmetric) tensor, then we give a lemma which will be useful in what is to follow.

Definition 36 (Tensor, Symmetric tensor)**.** *Let m, b be integers such that $2 \leq b \leq m$.*
A tensor *of dimension m and order b over $\mathbb{F}_q$ is a set*

$$(S_{i_1,i_2,\ldots,i_b})_{1\leq i_1,i_2,\ldots,i_b\leq m} \in \mathbb{F}_q^{m^b}.$$

Such a tensor is called a symmetric tensor *of dimension m and order b over $\mathbb{F}_q$ if for any permutation σ in the symmetric group $\mathcal{S}_b$, one has*

$$S_{i_1,i_2,\ldots,i_b} = S_{i_{\sigma(1)},i_{\sigma(2)},\ldots,i_{\sigma(b)}}.$$

In other words, a symmetric tensor is a tensor which is invariant under the action of the symmetric group on its indices.

Remark 15. *The $\mathbb{F}_q$-vector space of symmetric tensors of dimension m and order b over $\mathbb{F}_q$ is isomorphic to the vector space of homogeneous polynomials of degree b in m variables over $\mathbb{F}_q$.*

The latter admits the homogeneous monomials of degree b in m variables as a basis, thus it is of dimension

$$\binom{m+b-1}{b}.$$

Hence, the dimension of the vector space of symmetric tensors of dimension m and order b over $\mathbb{F}_q$ is given by $\binom{m+b-1}{b}$ as well.

Lemma 1. *For all $j \in \{1..m\}$, let $\boldsymbol{r}_j$ be the j-th row of $\sum_{i=1}^{K} x_i \boldsymbol{M}_i$.*

Let $b \geq 1$ be an integer, let J be a subset of $\{1..n\}$ such that $\#J = r + b$, then there are

$$\binom{n}{r+b}\binom{m+b-1}{b} \tag{3.6}$$

relations of the form

$$\sum_{i_1=1}^{m}\sum_{i_2=1}^{m}\cdots\sum_{i_b=1}^{m} S_{i_1,i_2,\ldots,i_b}\left|\begin{pmatrix}\boldsymbol{r}_{i_1}\\ \boldsymbol{r}_{i_2}\\ \cdots\\ \boldsymbol{r}_{i_b}\\ \boldsymbol{C}\end{pmatrix}\right|_{*,J} = 0 \tag{3.7}$$

where the $S_{i_1,i_2,\ldots,i_b}$'s are symmetric tensors of dimension m and order b over $\mathbb{F}_q$.

Proof. First of all, it is clear that the minor

$$\left|\begin{pmatrix}\boldsymbol{r}_{i_1}\\ \boldsymbol{r}_{i_2}\\ \cdots\\ \boldsymbol{r}_{i_b}\\ \boldsymbol{C}\end{pmatrix}\right|_{*,J}$$

will vanish if two rows are identical, i.e. if $i_j = i_k$ for any $1 \leq j, k \leq b$ such that $j \neq k$.

Thus, the left-hand side of Equation (3.7) can be restricted to a sum over all indices that are pairwise distinct since all the other terms are zero. This means that we can rewrite Equation (3.7) as a sum over sets of distinct indices and their permutations:

$$\sum_{\{i_1,i_2,\ldots,i_b\}\subset\{1..m\}} \sum_{\sigma\in\mathcal{S}_b} S_{i_{\sigma(1)},i_{\sigma(2)},\ldots,i_{\sigma(b)}} \left| \begin{pmatrix} \boldsymbol{r}_{i_{\sigma(1)}} \\ \boldsymbol{r}_{i_{\sigma(2)}} \\ \ldots \\ \boldsymbol{r}_{i_{\sigma(b)}} \\ \boldsymbol{C} \end{pmatrix} \right|_{*,J} = 0 \tag{3.8}$$

where $\mathcal{S}_b$ denotes the set of permutations over b symbols, also called the symmetric group of degree b.

By definition, a symmetric tensor is a tensor such that:

$$S_{i_1,i_2,\ldots,i_b} = S_{i_{\sigma(1)},i_{\sigma(2)},\ldots,i_{\sigma(b)}},\ \forall\sigma\in\mathcal{S}_b.$$

Thus, Equation (3.8) can be rewritten:

$$\sum_{\{i_1,i_2,\ldots,i_b\}\subset\{1..m\}} S_{i_1,i_2,\ldots,i_b} \sum_{\sigma\in\mathcal{S}_b} \left| \begin{pmatrix} \boldsymbol{r}_{i_{\sigma(1)}} \\ \boldsymbol{r}_{i_{\sigma(2)}} \\ \ldots \\ \boldsymbol{r}_{i_{\sigma(b)}} \\ \boldsymbol{C} \end{pmatrix} \right|_{*,J} = 0. \tag{3.9}$$

Applying a permutation σ on rows of a square matrix switches the sign of its determinant if, and only if, the signature of σ is -1.

Since the symmetric group $\mathcal{S}_b$ contains as many permutations whose signature is -1 as permutations whose signature is 1, the sum

$$\sum_{\sigma\in\mathcal{S}_b} \left| \begin{pmatrix} \boldsymbol{r}_{i_{\sigma(1)}} \\ \boldsymbol{r}_{i_{\sigma(2)}} \\ \ldots \\ \boldsymbol{r}_{i_{\sigma(b)}} \\ \boldsymbol{C} \end{pmatrix} \right|_{*,J}$$

equals 0; which ends the proof of Equation (3.7).

By Remark 15, about the dimension of the space of symmetric tensors, one has that there are

$$\binom{m+b-1}{b}$$

such relations; this holds for any set $J \subset \{1..n\}$ of size $r+b$, hence the result. □

With Lemma 1, we now have the tools to count the number of linearly independent equations in the SupportMinors system at bi-degree $(b,1),\ b \geq 2$.

Let us start with the $b = 2$ case. Recall that the SupportMinors system at bi-degree $(2,1)$ contains

$$Km\binom{n}{r+1}$$

equations containing at most

$$\binom{K+1}{2}\binom{n}{r}$$

distinct monomials of bi-degree $(2,1)$, see Equations (3.4) and (3.5).

Lemma 1 gives explicit linear dependencies between these equations. More precisely, expanding the minors in

$$\left|\begin{pmatrix}\boldsymbol{r}_i\\ \boldsymbol{r}_j\\ \boldsymbol{C}\end{pmatrix}\right|_{*,J} + \left|\begin{pmatrix}\boldsymbol{r}_j\\ \boldsymbol{r}_i\\ \boldsymbol{C}\end{pmatrix}\right|_{*,J} = 0$$

with respect to their first rows, one gets a non trivial linear combination of the equations of bi-degree $(1,1)$ multiplied by linear variables x_i's which vanishes.

These are precisely syzygies in the system at bi-degree $(2,1)$, and there are

$$\binom{n}{r+2}\binom{m+1}{2}$$

of them by Lemma 1.

This means that there are *at most*

$$Km\binom{n}{r+1} - \binom{n}{r+2}\binom{m+1}{2} \tag{3.10}$$

linearly independent equations in the SupportMinors system at bi-degree $(2,1)$.

Experimentally, we often obtained precisely the number given by Equation (3.10) of linearly independent equations. This is why we assume that we managed to identify all the linear dependencies between the equations of the SupportMinors system at bi-degree $(2,1)$.

Counting the number of linearly independent equations when $b = 3$, namely at bi-degree $(3,1)$, is a little bit trickier.

Using Equations (3.4) and (3.5), the SupportMinors system at bi-degree $(3,1)$ contains

$$\binom{K+1}{2} m \binom{n}{r+1}$$

equations containing at most

$$\binom{K+2}{3}\binom{n}{r}$$

distinct monomials of bi-degree $(3,1)$.

Recall that the system at bi-degree $(3,1)$ is obtained by multiplying all the equations of the system at bi-degree $(2,1)$ by the K linear variables x_i's.

Hence, the syzygies at the *previous* degree, i.e. $(2,1)$, will give K times more syzygies at degree $(3,1)$.

However, this time the syzygies are not linearly independent. In other words, it means that there exist non-trivial linear dependencies among the syzygies themselves. These linear dependencies come from relations of the form

$$\left|\begin{pmatrix} \boldsymbol{r}_i \\ \boldsymbol{r}_j \\ \boldsymbol{r}_k \\ \boldsymbol{C} \end{pmatrix}\right|_{*,J} + \left|\begin{pmatrix} \boldsymbol{r}_i \\ \boldsymbol{r}_k \\ \boldsymbol{r}_j \\ \boldsymbol{C} \end{pmatrix}\right|_{*,J} + \left|\begin{pmatrix} \boldsymbol{r}_j \\ \boldsymbol{r}_i \\ \boldsymbol{r}_k \\ \boldsymbol{C} \end{pmatrix}\right|_{*,J} + \left|\begin{pmatrix} \boldsymbol{r}_j \\ \boldsymbol{r}_k \\ \boldsymbol{r}_i \\ \boldsymbol{C} \end{pmatrix}\right|_{*,J} + \left|\begin{pmatrix} \boldsymbol{r}_k \\ \boldsymbol{r}_i \\ \boldsymbol{r}_j \\ \boldsymbol{C} \end{pmatrix}\right|_{*,J} + \left|\begin{pmatrix} \boldsymbol{r}_k \\ \boldsymbol{r}_j \\ \boldsymbol{r}_i \\ \boldsymbol{C} \end{pmatrix}\right|_{*,J} = 0. \tag{3.11}$$

In fact, by expanding the determinants in Equation (3.11) with respect to their first row, one gets precisely non-trivial relations *between the syzygies* that appear at degree $(3,1)$.

Like we did before, using Lemma 1, one gets exactly the number of such dependencies:

$$\binom{n}{r+3}\binom{m+2}{3}.$$

Overall, at degree $(3,1)$, there are $\binom{K+1}{2}m\binom{n}{r+1}$ equations among which there are $K\binom{n}{r+2}\binom{m+1}{2}$ syzygies; and last but not least, there are $\binom{n}{r+3}\binom{m+2}{3}$ linear dependencies among these syzygies.

Thus, the number of linearly independent equations in the SupportMinors system at bi-degree $(3,1)$ is at most

$$\binom{K+1}{2}m\binom{n}{r+1} - K\binom{n}{r+2}\binom{m+1}{2} + \binom{n}{r+3}\binom{m+2}{3}. \tag{3.12}$$

One more time, experimentally, we often had a Macaulay matrix whose rank was exactly given by Equation (3.12).

Thus, it is natural to extend the results given by Equation 3.12 to higher values of b using Lemma 1 and an alternating sum.

More precisely, let b be an integer such that $1 \leq b < r+2 < q$, we expect the number of linearly independent equations in the SupportMinors system at a bi-degree $(b,1)$ to be given by:

$$\sum_{i=1}^{b}(-1)^{i+1}\binom{n}{r+i}\binom{m+i-1}{i}\binom{K+b-i-1}{b-i},$$

of course as long as this value is smaller than the total number of monomials minus 1.

This means that if

$$\sum_{i=1}^{b}(-1)^{i+1}\binom{n}{r+i}\binom{m+i-1}{i}\binom{K+b-i-1}{b-i} \geq \binom{K+b-1}{b}\binom{n}{r} - 1, \tag{3.13}$$

we expect the rank of the Macaulay matrix of the system to be precisely $\binom{K+b-1}{b}\binom{n}{r} - 1$, thus one could find the values of each monomials using techniques described in Section 2.3.2.2.

Once one has the values of each monomials, she recovers the solution to the **MinRank** instance using the procedure described in Section 3.1.2.6.

Thus, since the most costly step in this attack is to find a vector in the right kernel of the Macaulay matrix, we can derive a complexity for our SupportMinors attack using Equation (3.13) together with the complexity formulas given in Section 2.3.2.3.

Overall, let b be the smallest positive integer such that $1 \leq b < r+2 < q$, and let E_b and U_b be defined as follow:

$$E_b := \sum_{i=1}^{b} (-1)^{i+1} \binom{n}{r+i} \binom{m+i-1}{i} \binom{K+b-i-1}{b-i}, \tag{3.14}$$

$$U_b := \binom{K+b-1}{b} \binom{n}{r}. \tag{3.15}$$

If $U_b - 1 \leq E_b$, and if the Macaulay matrix of the SupportMinors system at degree $b+1$ has rank $U_b - 1$, then the complexity to solve a **MinRank** instance with parameters (m, n, K, r) using the SupportMinors attack at bi-degree $(b, 1)$ is in

$$\mathcal{O}\left(\min\left(E_b U_b^{\omega-1}, K(r+1)E_b U_b\right)\right) \tag{3.16}$$

operations in $\mathbb{F}_q$, where ω is the linear algebra constant.

The minimum in Equation (3.16) comes from the use of Wiedemann or Strassen's algorithms, see Section 2.3.2.3.

3.1.2.3 Discussions about the cases $q = 2$ and $b \geq r + 2$

In the previous section, there were two conditions on b, namely $b < r + 2$ and $q < b$. Let us start with the former.

When dealing with a **MinRank** over $\mathbb{F}_2$, one wants to find the solution vector with entries in $\mathbb{F}_2$, which means using the field equations $x_i^2 = x_i$ for every linear variable x_i.

When $b \geq 2$, this leads to the collapse of some monomials to lower degree, hence the system is no longer homogeneous.

Thus, in this case it is most profitable to combine the equations obtained at a given value of b with those produced using all smaller values of b. Similar considerations to the general case imply that as long as $b < r + 2$ we will have

$$E_b := \sum_{j=1}^{b} \sum_{i=1}^{j} (-1)^{i+1} \binom{n}{r+i} \binom{m+i-1}{i} \binom{K}{j-i}. \tag{3.17}$$

equations with which to linearize the

$$U_b := \sum_{j=1}^{b} \binom{n}{r} \binom{K}{j}$$

monomials that occur at a given value of b.

Since our experiments support the number of linearly independent equations given by Equation (3.17), we therefore expect to be able to solve by linearization when $b < r + 2$ and b is large enough so that

$$U_b - 1 \leq E_b. \tag{3.18}$$

Similarly to the general case for any q described in the previous section, when $q = 2$ and $b < r + 2$, the complexity to find the kernel vector is given by

$$\mathcal{O}\left(K(r+1)E_bU_b\right) \tag{3.19}$$

in terms of binary operations, where b is the smallest positive integer so that the condition (3.18) is fulfilled.

Like in the previous Section, once one has this kernel vector, finding the solution to the **MinRank** instance can be done using Section 3.1.2.6.

Last but not least, the condition $b < r + 2$ comes from the fact that we can build additional syzygies at this degree using Theorem 1 in [BBB+20] from [FSEDS11]. Note that Theorem 14 in this document is the same theorem but applied to generic matrices containing elements in $\mathbb{F}_q$ instead of linear forms.

However, such high degrees are not relevant in a cryptographic context.

Indeed, most cryptanalysis are performed at bi-degree $(b, 1)$ where b is small before r, in particular with the techniques described in Sections 3.1.2.4 and 3.1.2.5.

For instance, Beullens performed his cryptanalysis on the Rainbow signature scheme at bi-degree $(2, 1)$ when the target rank was $r = 5$, see [Beu22].

3.1.2.4 Hybrid case

For a given **MinRank** instance with parameters (q, m, n, K, r), it sometimes happens that the linearization condition $U_b - 1 \leq E_b$, where U_b and E_b are defined, respectively, by Equations (3.15) and (3.14), is not fulfilled for any $b \in \{1..r + 1\}$.

In that case, it is possible to use an hybrid approach by guessing $K_{\text{hyb}} > 0$ linear variables x_i at an exponential cost of $q^{K_{\text{hyb}}}$, as described in [BBC+20].

Doing so, one is left with a **MinRank** instance with parameters

$$(q, m, n, K - K_{\text{hyb}}, r),$$

which will be solvable at bi-degree $(b, 1)$ with $b < r + 2$; see Table 3.1 for some examples.

Note that, even if a SupportMinors system is solvable at bi-degree $(b, 1)$ for some $b < r + 2$, it is sometimes worth guessing a few linear variables in order to reach smaller values of b. In fact, there is a tradeoff between the exponential cost of guessing linear variables and the cost to solve a "smaller" **MinRank** instance.

3.1.2.5 Improvements

All along this section, one is given a **MinRank** instance with parameters

$$(q, m, n, K, r).$$

First of all, note that for non-square **MinRank** instances, it is sometimes worth considering the transposed matrices, in fact it could lead to smaller complexities.

Overdetermined case. In Section 3.1.2.4, we considered the case where the linearization condition is not fulfilled; in the opposite case, it could be fulfilled *too much*.

More precisely, if the ratio between the number of *linearly independent* equations and the number of monomials that appear in the system is above 1, it is sometimes worth considering only $n' < n$ columns of the matrices.

In fact, if there is a unique solution to a **MinRank** instance where the matrices have size $m \times n$, and if there is also one solution to the instance one gets from keeping only the first $n' < n$ columns of the matrices, then these solutions are the same.

Generically, we will keep a unique solution in the smaller instance as long as we respect the Gilbert-Varshamov bound for matrix codes, see Definition 22.

This improvement of the attack corresponds to puncturing the matrix code given by the **MinRank** instance. It is worth noticing that a tradeoff might be found, like in Section 3.1.2.4, between the degree b and the number of kept columns n'.

More precisely, it is always beneficial for the attacker to reduce n to the minimum value n' allowing linearization at a given degree b, however, it can sometimes lead to an even lower complexity to reduce n further and solve at a higher degree b.

New hybrid approach. In [BBB+22], we introduced an hybrid strategy different from the one presented in Section 3.1.2.4; in a few words, for $a > 0$, at an exponential cost of q^{ar}, one is left with solving a smaller **MinRank** instance with parameters

$$(q, m, n - a, K - am, r).$$

This approach may allow one to reach smaller complexities than the ones obtained with the aforementioned specialization technique of [BBC+20], especially for instances with larger q. As an illustration, we present some parameter sets in Table 3.1. The first one comes from Table 24.7.1 in [Cou01b], and the following ones are obtained by increasing the value of q from 2 to 64 but by keeping the same parameters (m, n, K, r).

(q, m, n, K, r)	SM 20	b	K_{hyb}	SM 22	b	a	comb
$(2, 19, 19, 81, 10)$	**115**	1	66	**75**	1	4	45
$(4, 19, 19, 81, 10)$	**180**	2	62	**115**	1	4	90
$(8, 19, 19, 81, 10)$	**219**	11	34	**155**	1	4	135
$(16, 19, 19, 81, 10)$	**253**	11	34	**195**	1	4	180
$(64, 19, 19, 81, 10)$	**321**	11	34	**256**	1	4	270

Table 3.1: Comparison between the SupportMinors hybrid approach of [BBC+20] (specialization of K_{hyb} linear variables), see Section 3.1.2.4, in column "SM 20", and the approach of [BBB+22], see Section 3.1.2.5, in column "SM 22". The "comb" column refers to the complexity of the best combinatorial attacks against **MinRank**, see §4.2 in [Cou01a] for a broader description of these attacks.

Note that for $q = 64$, and also for even larger values which are not presented here, the SupportMinors attack starts beating the best combinatorial attacks, and the greater q, the bigger the gap.

The **MinRank**-based sigma protocol by Courtois [Cou01a] has been recently improved in [BESV22]. The parameters of this scheme have been computed by relying on our analysis and they can be found in Table 1 in [BESV22].

3.1.2.6 Last step of the attack

Using the SupportMinors attack described in Sections 3.1.2.1 to 3.1.2.5, one gets the values of each monomials involved in the system; they have the following form

$$x_{i_1} x_{i_2} \dots x_{i_b} c_T,$$

and are thus of degree $b+1$ for some $1 \leq b \leq r+1$.

Recall that if $\boldsymbol{v}$ is a non-zero vector which gives the values of each monomials in $\mathbb{F}_q$ (*i.e*, $\boldsymbol{v}$ is in the right-kernel of the Macaulay matrix, see Section 2.3.2.2), then for any $\alpha \in \mathbb{F}_q^\times$, $\alpha\boldsymbol{v}$ still gives valid values for the monomials.

Thus, picking an index i_0 and a set T_0 such that the monomial

$$x_{i_0}^b c_{T_0}$$

is not zero enables one to set $x_{i_0} = 1$. Then one gets the values of all the linear variables by computing the following ratios

$$x_i = \frac{x_i}{x_{i_0}} = \frac{x_i x_{i_0}^{b-1} c_{T_0}}{x_{i_0}^b c_{T_0}}, \quad \forall\, i \in \{1, 2, \dots, K\}.$$

3.1.2.7 Complexity results on some cryptosystems

In this section, we apply the complexity formulas for our SupportMinors algebraic attack to some multivariate-based cryptosystems, namely GeMSS [CFMR+19], and Rainbow [DCP+19], see in Table 3.2.

In the case of GeMSS, one sees that our attack against **MinRank** drastically improved the previous ones. However, it is not sufficient to go below the security levels.

As for the Rainbow signature scheme, our attack does improve on the previous **MinRank** attack, and for the two last sets of parameters, it also beats by a few bits the best attack against the scheme, namely the Direct Algebraic (DA) attack. Note that the acronym RBS stands for Rainbow Band Separation attack.

It is worth noticing that since the first paper [BBC+20] where we presented the SupportMinors attack, it has been used for some multivariate-based signatures cryptanalysis, for instance in [BBC+22] and [Beu22].

						Complexity		
GeMSS(D, n, Δ, v)	n	K	r	n'	b	New	Previous	Type
GeMSS128$(513, 174, 12, 12)$	174	162	34	61	2	**154**	522	**MinRank**
GeMSS192$(513, 256, 22, 20)$	265	243	52	94	2	**223**	537	**MinRank**
GeMSS256$(513, 354, 30, 33)$	354	324	73	126	3	**299**	1254	**MinRank**
RedGeMSS128$(17, 177, 15, 15)$	177	162	35	62	2	**156**	538	**MinRank**
RedGeMSS192$(17, 266, 23, 25)$	266	243	53	95	2	**224**	870	**MinRank**
RedGeMSS256$(17, 358, 34, 35)$	358	324	74	127	3	**301**	1273	**MinRank**
BlueGeMSS128$(129, 175, 13, 14)$	175	162	35	63	2	**158**	537	**MinRank**
BlueGeMSS192$(129, 265, 22, 23)$	265	243	53	95	2	**224**	870	**MinRank**
BlueGeMSS256$(129, 358, 34, 32)$	358	324	74	127	3	**301**	1273	**MinRank**

Rainbow$(GF(q), v_1, o_1, o_2)$	n	K	r	n'	b	New	Previous	Best / Type
Ia$(GF(16), 32, 32, 32)$	96	33	64	82	3	155	161	145 / RBS
IIIc$(GF(256), 68, 36, 36)$	140	37	104	125	5	**208**	585	215 / DA
Vc$(GF(256), 92, 48, 48)$	188	49	140	169	5	**272**	778	275 / DA

Table 3.2: Complexity comparison between the new and the previous attacks against GeMSS [CFMR+19], and Rainbow [DCP+19].

Last but not least, the very last improvement of our SupportMinors attack, namely the hybrid strategy described in [BBB+22], see Section 3.1.2.5, has been used to compute the parameters of the MR-DSS signature scheme, see Table 1 in [BESV22].

3.2 Algebraic Attack against RD

There are two general classes of attacks against **RD**, see Definition 17: the combinatorial and the algebraic attacks.

Combinatorial attacks can be seen as the equivalent of ISD-based (Information Set Decoding) attacks in a rank metric context. The current best combinatorial

attacks against **RD** are given by [GRS16, AGHT18].

In a few words, they state that, if one is given an **RD** instance with parameters (q, m, n, k, r), then one can solve it using

$$\widetilde{\mathcal{O}}\left(\min\left(q^{(w-1)\left\lfloor\frac{(k+1)m}{n}\right\rfloor}, q^{w\left\lceil\frac{(k+1)m}{n}\right\rceil-m}\right)\right) \tag{3.20}$$

elementary operations in $\mathbb{F}_q$.

In what follows, we describe an algebraic attack against **RD**, namely the MaxMinors attack.

3.2.1 MaxMinors Modeling

In this section, we consider an $\mathbf{RD}(m, n, k, r)$ instance over $\mathbb{F}_q$, let $\boldsymbol{G} \in \mathbb{F}_{q^m}^{k\times n}$ be a generator matrix of the code associated to this instance, that we call $\mathcal{C}$.

It means that one is given a noisy codeword

$$\boldsymbol{y} = \boldsymbol{x}\boldsymbol{G} + \boldsymbol{e} \in \mathbb{F}_{q^m}^n$$

where $\|e\| \leq r$, and the goal is to recover $\boldsymbol{e}$, hence $\boldsymbol{x}$. As mentioned in Remark 12, in what follows we will consider that $\|e\| = r$.

First of all, let us add the word $\boldsymbol{y}$ to the code $\mathcal{C}$, this creates a new augmented code $\widetilde{\mathcal{C}}$ of length n, dimension $k+1$, generated by the following matrix:

$$\begin{pmatrix}\boldsymbol{G}\\ \boldsymbol{y}\end{pmatrix} \in \mathbb{F}_{q^m}^{(k+1)\times n}.$$

Let $\boldsymbol{H} \in \mathbb{F}_{q^m}^{(n-k-1)\times n}$ be a parity check matrix of $\widetilde{\mathcal{C}}$, then $\boldsymbol{e}\boldsymbol{H}^\intercal = \boldsymbol{0}^{n-k-1}$.

We say that we *homogenized* the problem, because now we are left with a syndrome equation whose right-hand side is zero.

Remark 16. *Note that not only $\boldsymbol{e}$ belongs to $\widetilde{\mathcal{C}}$, but all its $\mathbb{F}_{q^m}^\times$-multiples.*

Although this last remark was very important in our previous attack against **RD**, described in [BBB+20], it will not be as important for this attack. We will just use it in the last step, see the end of the proof of Proposition 5.

The error vector $\boldsymbol{e}$ can be written

$$\beta\,\mathrm{Mat}\,(\boldsymbol{e})\,;$$

recall that β is a fixed basis of $\mathbb{F}_{q^m}$ seen as a vector space over $\mathbb{F}_q$, and $\mathrm{Mat}\,(\cdot)$ has been defined in Definition 8.

By definition, the matrix $\mathrm{Mat}\,(\boldsymbol{e})$ has rank r; thus it can be factorized exactly like we did in Equation (3.1).

Overall,

$$\boldsymbol{e} = \beta \boldsymbol{S} \boldsymbol{C},$$

where $\boldsymbol{S} \in \mathbb{F}_q^{m \times r}$, and $\boldsymbol{C} \in \mathbb{F}_q^{r \times n}$ both have full rank r.

By replacing this expression for $\boldsymbol{e}$ in the aforementioned homogenized syndrome equation, one gets

$$\beta \boldsymbol{S} \boldsymbol{C} \boldsymbol{H}^\intercal = \boldsymbol{0}^{n-k-1}. \tag{3.21}$$

One of the primordial points in our attack is given by the following theorem.

Theorem 2. *The maximal minors of the matrix $\boldsymbol{C}\boldsymbol{H}^\intercal \in \mathbb{F}_{q^m}^{r \times (n-k-1)}$ are equal to zero.*

Proof. The key point is to notice that the vector $\beta \boldsymbol{S}$ is not zero by construction, thus there is at least one non trivial element in the left kernel of the matrix $\boldsymbol{C}\boldsymbol{H}^\intercal \in \mathbb{F}_{q^m}^{r \times (n-k-1)}$. This means that its rank is smaller than r, thus all its maximal minors will vanish. □

Theorem 2 already enables one to have an algebraic modeling to solve the **RD** instance: let $\boldsymbol{C}$ be an $r \times n$ matrix containing variables:

$$\{f = 0 \mid f \in \mathrm{MaxMinors}\,(\boldsymbol{C}\boldsymbol{H}^\intercal)\}. \tag{3.22}$$

More precisely, the solutions in $\mathbb{F}_q^{rn}$ of the algebraic system given by Equation (3.22) are possible values for the entries of the matrix $\boldsymbol{C}$.

It is important to notice that despite the fact the maximal minors of the matrix $\boldsymbol{C}\boldsymbol{H}^\intercal$ have coefficients in $\mathbb{F}_{q^m}$, the variables in the matrix $\boldsymbol{C}$ belong to $\mathbb{F}_q$. This is why we can get m times more equations by considering each equation $f = 0$ as m equations over $\mathbb{F}_q$. We call this process the *unfolding*, and it is formally defined by the following definition.

Definition 37 (Unfolding)**.** *Let $\mathcal{S} := \{\sum_j a_{ij} m_{ij} = 0, 1 \leq i \leq N\}$ be a set of polynomial equations where the m_{ij}'s are the monomials in the unknowns that are assumed to belong to $\mathbb{F}_q$, whereas the a_{ij}'s are known coefficients that belong to $\mathbb{F}_{q^m}$.*

We define the a_{ijk}'s as $a_{ij} = \sum_{k=0}^{m-1} a_{ijk}\alpha^k$, where the a_{ijk}'s belong to $\mathbb{F}_q$. From this we can define the unfolding *of the system over $\mathbb{F}_q$ as*

$$\text{Unfold}(\mathcal{S}) := \left\{ \sum_j a_{ijk} m_{ij} = 0, 1 \leq i \leq N, 0 \leq k \leq m-1 \right\}.$$

The important point is that the solutions of $\mathcal{S}$ over $\mathbb{F}_q$ are exactly the solutions of Unfold($\mathcal{S}$) over $\mathbb{F}_q$; in that sense the two systems are equivalent.

The second primordial point of our attack is to write each maximal minor of the matrix $\boldsymbol{C}\boldsymbol{H}^\intercal$ as a linear combination of maximal minors of $\boldsymbol{C}$ seen as new variables.

This linearization process is the same as the one described in Section 3.1.1, i.e. one considers $\binom{n}{r}$ variables c_T's that are the maximal minors of the matrix $\boldsymbol{C}$.

In order to write the maximal minors of the matrix $\boldsymbol{C}\boldsymbol{H}^\intercal$ as linear combinations of maximal minors of $\boldsymbol{C}$, we need the following theorem:

Theorem 3. *Let $J \subset \{1..n-k-1\}$ such that $\#J = r$, then*

$$|\boldsymbol{C}\boldsymbol{H}^\intercal|_{*,J} = \sum_{T \subset \{1..n\},\ \#T=r} |\boldsymbol{H}^\intercal|_{T,J}\, c_T$$

Proof. The Cauchy-Binet formula generalizes the formula

$$\det(AB) = \det(A)\det(B)$$

to non-square matrices.

Theorem 4 (Cauchy-Binet formula)**.** *Let m be an integer such that $1 \leq m \leq n$, let $A \in \mathbb{K}^{m\times n}$, and $B \in \mathbb{K}^{n\times m}$, then*

$$|AB| = \sum_{T \in \{1,2,\ldots,n\},\ \#T=m} |A|_{*,S}\, |B|_{S,*}\,.$$

The Cauchy-Binet formula is then readily adapted to minors of a product of two matrices. Theorem 3 is a straightforward application of this formula to the maximal minors of the matrix $\boldsymbol{C}\boldsymbol{H}^\intercal$. □

We have now described all the elements that are necessary to define the so-called MaxMinors modeling:

Modeling 2 (MaxMinors)**.** *With the previously defined notation, the MaxMinors modeling is the linear system of algebraic equations defined by*

$$\text{Unfold}\left(\{f = 0 \mid f \in \text{MaxMinors}\left(\boldsymbol{C}\boldsymbol{H}^{\intercal}\right)\}\right).$$

This system contains:

- $m\binom{n-k-1}{r}$ *linear equations with coefficients in* $\mathbb{F}_q$*,*
- $\binom{n}{r}$ *variables* c_T *that are searched in* $\mathbb{F}_q$*.*

3.2.2 Complexity

As noted at the beginning of Section 3.1.2.1, using the variables c_T's instead of degree r polynomials for the maximal minors of $\boldsymbol{C}$ enables one to drastically reduce the size of the system, roughly by a factor $r!$.

In addition to this, the MaxMinors modeling is a linear system.

This choice of linearization corresponds to using Plücker coordinates as described in Section 2.3.3.

Conjecture 1. *With overwhelming probability, as long as*

$$m\binom{n-k-1}{r} \geq \binom{n}{r} - 1, \tag{3.23}$$

the matrix of the linear system given by the MaxMinors modeling, see Modeling 2, will have rank $\binom{n}{r} - 1$*.*

We extensively checked Conjecture 1 on **RD** instances generated at random and it was always verified.

Proposition 5 (Complexity of the MaxMinors attack)**.** *If* $m\binom{n-k-1}{r} \geq \binom{n}{r} - 1$*, and if the matrix of the linear system given by the MaxMinors modeling has rank* $\binom{n}{r} - 1$*, then the complexity to solve the associated* **RD** *instance is in*

$$\mathcal{O}\left(m\binom{n-k-1}{r}\binom{n}{r}^{\omega-1}\right) \tag{3.24}$$

operations in $\mathbb{F}_q$*, where* ω *is the linear algebra constant.*

Proof. If the matrix associated to the linear system given by the MaxMinors modeling has rank $\binom{n}{r} - 1$, its single non-zero right kernel vector gives the values of all the maximal minors c_T's, see Section 2.3.2.2 for more details about this so-called linearization process.

Computing this kernel requires linear algebra on a $\left(m\binom{n-k-1}{r}\right) \times \binom{n}{r}$ matrix with entries in $\mathbb{F}_q$, this yields the complexity given by Equation (3.24). This step is clearly the one that concentrates the majority of the complexity since the last step only involves linear algebra on far shorter matrices.

More precisely, to recover the entries of $\boldsymbol{C}$ from the values of all the c_T's, one uses the straightforward procedure described in Section 2.3.3.

Last, knowing the entries of the matrix $\boldsymbol{C}$, one only has to solve a linear system to recover the coefficients of the matrix $\boldsymbol{S}$, hence the error vector $\boldsymbol{e}$. This last step corresponds to the end of Algorithm 1 in [BBC+20]. □

3.2.2.1 Overdetermined case

In case the ratio

$$\frac{m\binom{n-k-1}{r}}{\binom{n}{r} - 1} \tag{3.25}$$

is strictly greater than 1, which means that there are more equations than unknowns in the linear system given by the MaxMinors modeling, one can reduce the complexity by puncturing the code $\widetilde{\mathcal{C}}$.

Note that, by removing p coordinates, the code $\widetilde{\mathcal{C}}$ will have length $n - p$ and so will be the number of columns of the matrix $\boldsymbol{C}$. This has an impact on the number of variables but also on the number of equations; this is very convenient since it leads to a smaller linear system, thus it decreases the complexity to solve it.

More precisely, one wants to find the biggest integer p such that

$$m\binom{n-k-1-p}{r} \geq \binom{n-p}{r} - 1, \tag{3.26}$$

then the complexity to solve the associated **RD** instance, as long as the linear system has rank $\binom{n-p}{r} - 1$, becomes

$$\mathcal{O}\left(m\binom{n-k-1-p}{r}\binom{n-p}{r}^{\omega-1}\right).$$

This is referred to as the *overdetermined* case.

3.2.2.2 Hybrid case

On the contrary, if the condition given by Equation (3.23) is not fulfilled, i.e. if the ratio given by Equation (3.25) is smaller than 1; one can guess, formally one says *specialize*, $a > 0$ columns of the matrix $\boldsymbol{C}$ at an exponential cost of q^{ar}, and solve a new system with less variables but as many equations, i.e. it increases the aforementioned ratio.

More precisely, one wants to find the smallest integer a such that

$$m\binom{n-k-1}{r} \geq \binom{n-a}{r} - 1, \tag{3.27}$$

then the complexity to solve the associated **RD** instance, as long as the linear system has rank $\binom{n-a}{r} - 1$, becomes

$$\mathcal{O}\left(q^{ar} m \binom{n-k-1}{r} \binom{n-a}{r}^{\omega-1}\right).$$

This is referred to as the *hybrid* case.

Specializing a columns in $\boldsymbol{C}$. Since we know the value of a columns of the matrix $\boldsymbol{C}$, it seems pretty natural that one is left with a system containing only

$$\binom{n-a}{r}$$

variables c_T's.

However, getting this number of variables is not a straightforward process. This is why this section is dedicated to explaining it. Let us start with a toy example in characteristic 2 to avoid sign issues, here is a small 2×5 matrix $\boldsymbol{C}$:

$$\begin{pmatrix} \lambda & a & b & c & d \\ \mu & e & f & g & h \end{pmatrix},$$

it contains 8 variables $(a, b, c, \ldots, h)$ and 2 known coefficients $\lambda, \mu \in \mathbb{F}_2^2$.

There are $\binom{5}{2} = 10$ maximal minors, i.e. 2×2 minors, in this matrix:

- $c_{\{1,2\}} = \lambda e + \mu a$
- $c_{\{1,3\}} = \lambda f + \mu b$

- $c_{\{1,4\}} = \lambda g + \mu c$
- $c_{\{1,5\}} = \lambda h + \mu d$
- $c_{\{2,3\}} = af + be$
- $c_{\{2,4\}} = ag + ce$
- $c_{\{2,5\}} = ah + de$
- $c_{\{3,4\}} = bg + cf$
- $c_{\{3,5\}} = bh + df$
- $c_{\{4,5\}} = ch + dg$

One notices that even if we did specialize one column, i.e. the actual values of λ and μ are known, there are still 10 variables c_T's, which is different from the expected $\binom{5-1}{2} = 6$. And there is no way of expressing a variable c_T in terms of the others without introducing new variables; recall that we do not deal with the variables $a, b, c, \ldots, h$, only with the c_T's.

The trick is to pick a c_T, and put an identity block in $\boldsymbol{C}$ on the T coordinates. Without loss of generality, let us assume that here $T = \{1, 2\}$. Thus $c_{\{1,2\}} = 1$, and the matrix $\boldsymbol{C}$ becomes

$$\begin{pmatrix} 1 & 0 & \lambda & a & b \\ 0 & 1 & \mu & c & d \end{pmatrix},$$

and its maximal minors now are:

- $c_{\{1,2\}} = 1$
- $c_{\{1,3\}} = \mu$
- $c_{\{1,4\}} = c$
- $c_{\{1,5\}} = d$
- $c_{\{2,3\}} = \lambda$
- $c_{\{2,4\}} = a$
- $c_{\{2,5\}} = b$

- $c_{\{3,4\}} = \lambda c + \mu a$
- $c_{\{3,5\}} = \lambda d + \mu b$
- $c_{\{4,5\}} = ad + bc$.

This time we have only 6 variables left, the ones in blue above. This is precisely

$$\binom{5-1}{2} = 6.$$

More precisely, while computing the MaxMinors linear system, only the 6 blue variables will appear, the others will be replaced by $\mathbb{F}_2$-linear combinations of the blue variables using the following rules:

- $c_{\{1,3\}} = \mu c_{\{1,2\}}$
- $c_{\{2,3\}} = \lambda c_{\{1,2\}}$
- $c_{\{3,4\}} = \lambda c_{\{1,4\}} + \mu c_{\{2,4\}}$
- $c_{\{3,5\}} = \lambda c_{\{1,5\}} + \mu c_{\{2,5\}}$.

Remark 17. *This specialization depends on the choice $c_{\{1,2\}} = 1$, if it does not work, that is to say, if the linear system at the end is inconsistent, it simply means that this particular maximal minor is singular in any solution matrix $\boldsymbol{C}$.*

Then one picks another set T and start again, eventually this works since $\boldsymbol{C}$ is of full rank by construction.

In practice, the number of attempts before it works is about the same as the inverse of the probability for an $r \times r$ matrix to be non-singular over $\mathbb{F}_q$. that is to say, around 4 attempts over $\mathbb{F}_2$ and it increases exponentially fast up to 1 as q increases.

This specialization process is generalized with Proposition 6.

Proposition 6. *Let a be an integer such that $1 \leq a < n - r$.*

Let $\boldsymbol{C}$ be an $r \times n$ matrix which is made of an identity block of size r, a matrix $\boldsymbol{A} \in \mathbb{F}_q^{r \times a}$, and variables on the remaining $r(n - r - a)$ entries, i.e.

$$\boldsymbol{C} := (\ \mathbf{I}_r \mid \boldsymbol{A} \mid (C_{i,j})_{i,j}\ ).$$

One can express all the maximal minors of $\boldsymbol{C}$ *as* $\mathbb{F}_q$*-linear combinations of only*

$$\binom{n-a}{r}$$

variables c_T*, where* $c_T := |\boldsymbol{C}|_{*,T}$ *.*

Proof. Let us split the sets $T \subset \{1..n\}$ of size r in two sets:

$$S_1 := \{T \subset \{1..n\} \mid \#T = r,\ T \cap \{r+1..r+a\} \neq \emptyset\},$$

and

$$S_2 := \{T \subset \{1..n\} \mid \#T = r\} \backslash S_1.$$

that is to say that the set S_2 contains all the sets $T \subset \{1..n\}$ of size r which do not contain any indices in $\{r+1..r+a\}$.

The set S_2 clearly contains $\binom{n-a}{r}$ elements.

It is trivial to see that every maximal minor of $\boldsymbol{C}$ on a set of columns indexed by $T \in S_2$ can be written as an $\mathbb{F}_q$-linear combination of the following variables:

$$\{c_T \mid T \in S_2\}.$$

The rest of the proof consists in showing that every maximal minors of $\boldsymbol{C}$ on a set of columns indexed by $T \in S_1$ can also be written as an $\mathbb{F}_q$-linear combination of the following variables:

$$\{c_T \mid T \in S_2\}.$$

Let c_T be such that $T \in S_1$, then it is possible to write

$$T \cap \{r+1..r+a\} = \{i_1, i_2, \ldots, i_s\}$$

for an integer s smaller or equal to r.

Using several times Laplace expansion, one has that

$$\begin{aligned}
c_T &:= |\boldsymbol{C}|_{*,T} && (3.28)\\
&= \sum_{J \subset \{1..r\},\ \#J = r-s} \lambda_J \, |\boldsymbol{C}|_{J,\ T\backslash\{i_1,i_2,\ldots,i_s\}} && (3.29)\\
&= \sum_{J \subset \{1..r\},\ \#J = r-s} \lambda_J \left| \left((\mathbf{I}_r)_{*,\ \{1..r\}\backslash J} \mid \boldsymbol{C}_{*,\ T\backslash\{i_1,i_2,\ldots,i_s\}} \right) \right|_{*,*} && (3.30)\\
&= \sum_{J \subset \{1..r\},\ \#J = r-s} \lambda_J c_{\{J \,\cup\, T\backslash\{i_1,i_2,\ldots,i_s\}\}} && (3.31)
\end{aligned}$$

The λ's are elements in $\mathbb{F}_q$, and the key observation is to notice that the last variable

$$c_{\{J \cup T \setminus \{i_1, i_2, \ldots, i_s\}\}}$$

is a c_T where T belongs to S_2, which finishes the proof. □

Last but not least, there is another way to perform an hybrid MaxMinors algebraic attack.

We described this new process in [BBB+22]; it is based on a randomized algorithm with success probability q^{-ar}; with this algorithm, one is then left with solving a smaller **RD** instance. Namely, the instance will have parameters $(q, m, n-a, k-a, r)$ instead of (q, m, n, k, r).

Plugging these new values in the complexity of the MaxMinors approach, see Equation (3.24), one precisely gets the hybrid complexity given above.

Note that this way of performing the hybrid attack is the same as the one mentioned in Section 3.1.2.5, basically it applies to both **MinRank** and **RD**, see Section 5 of [BBB+22].

3.2.2.3 Complexity results on some cryptosystems

In this Section, we apply the MaxMinors algebraic attack to some cryptosystems.

Before that, in the next paragraph, we explain how to actually compute complexities for the MaxMinors attack.

Find the optimal complexity for MaxMinors. While computing the complexity of our MaxMinors attack against an **RD** instance, it is not always in the overdetermined or in the hybrid case; actually, the best complexity is sometimes obtained by combining both.

Recall that when the ratio given by Equation (3.25) is smaller than 1, one needs to specialize $a > 0$ columns of $\boldsymbol{C}$, this corresponds to the *hybrid case*, see Section 3.2.2.2.

However, when it becomes greater than 1, it is sometimes large enough so that one could puncture the code, this is the *overdetermined case*, see Section 3.2.2.1.

Let us see that on a concrete example: for an **RD** instance with parameters $[q = 2,\ m = 50,\ n = 50,\ k = 25,\ r = 12]$, the ratio is around 10^{-3}, so one definitely has to specialize a few columns. The smallest integer a such that the ratio is bigger than 1 is 20, however with $a = 20$, the ratio reaches 1.56.

Thus, one could puncture the code on 2 coordinates, i.e. set $p = 2$, to bring the ratio down to 1.06.

In conclusion, the couple $(a, p) = (20, 2)$ is somehow optimal, and the complexity of the attack is then 310 bits instead of 315 if one had only considered the hybrid attack with $a = 20$ and $p = 0$.

Our attack on some cryptosystems. Table 3.3 gives the complexity of our attack together with the (a, p) couples against some rank-based cryptosystems. They are compared with the claimed security levels of the cryptosystems and with our previous algebraic attack against **RD**, namely [BBB+20].

We can see that our attack improves the previous one by a great deal, and it is clearly below the security level for some cryptosystems.

Note that the ROLLO and RQC parameters considered are the ones before the Second Round updates.

Cryptosystem	**RD**(m, n, k, r)	Sec.	[BBB+20]	MaxMinors	(a, p)
Loidreau [Loi17]	$(128, 120, 80, 4)$	256	98	**65**	$(0, 3)$
ROLLO-I	$(79, 94, 47, 5)$	128	117	**71**	$(0, 9)$
ROLLO-I	$(89, 106, 53, 6)$	192	144	**87**	$(0, 0)$
ROLLO-I	$(113, 134, 67, 7)$	256	197	**158**	$(8, 0)$
ROLLO-II	$(83, 298, 149, 5)$	128	134	**93**	$(0, 40)$
ROLLO-II	$(107, 302, 151, 6)$	192	164	**111**	$(0, 18)$
ROLLO-II	$(127, 314, 157, 7)$	256	217	**170**	$(6, 0)$
ROLLO-III	$(101, 94, 47, 5)$	128	119	**70**	$(0, 12)$
ROLLO-III	$(107, 118, 59, 6)$	192	148	**88**	$(0, 4)$
ROLLO-III	$(131, 134, 67, 7)$	256	200	**138**	$(5, 0)$
RQC-I	$(97, 134, 67, 5)$	128	123	**77**	$(0, 18)$
RQC-II	$(107, 202, 101, 6)$	192	156	**101**	$(0, 10)$
RQC-III	$(137, 262, 131, 7)$	256	214	**144**	$(3, 0)$

Table 3.3: Complexity of the MaxMinors algebraic attack against some rank-based cryptosystems. The complexities (in bits) are compared to the claimed security levels (column "Sec."), and to our previous algebraic attack [BBB+20].

3.3 Attacks against NHRD, RSL, NHRSL

3.3.1 Attacks against the NHRD problem

This section is dedicated to the first cryptanalysis of **NHRD**, see Definition 19, for which we propose two attacks exploiting the non-homogeneous structure of the error.

3.3.1.1 A new combinatorial attack.

Before explaining our new combinatorial attack against **NHRD**, let us recall how the combinatorial attacks [GRS16, AGHT18] against **RD** work.

Given an $\mathbf{RD}(m, n, k, w_1)$ instance whose solution is an error $\boldsymbol{e}$ of rank weight w_1, the main idea is to pick a vector space $V \subset \mathbb{F}_{q^m}$ of dimension $r \geq w_1$ such that $\text{Support}(\boldsymbol{e}) \subset V$.

If it is the case, one is basically left with solving a linear system.

This process can be seen as an adaptation of the Information Set Decoding, used in the Hamming metric, to the rank metric.

The probability that a vector space of dimension $r \geq w_1$ picked at random in $\mathbb{F}_{q^m}$ contains $\text{Support}(\boldsymbol{e})$, is given by

$$\frac{\begin{bmatrix} m - w_1 \\ r - w_1 \end{bmatrix}_q}{\begin{bmatrix} m \\ r \end{bmatrix}_q} \approx q^{-w_1(m-r)} \tag{3.32}$$

where the approximation is obtained from the classical approximation of a Gaussian coefficient:

$$\begin{bmatrix} x \\ y \end{bmatrix}_q \approx q^{y(x-y)}.$$

For more details, the reader may refer to [GRS16].

Then, the complexity of the attack is derived from choosing the optimal value of r to get a single solution in the linear system.

In the case of a non-homogeneous error, our combinatorial approach follows the same path, roughly, we will pick 2 vector spaces V and Z of dimension, respectively, r and ρ, and we hope that $\text{Support}((\boldsymbol{e}_1, \boldsymbol{e}_3)) \subset V$, and that $\text{Support}(\boldsymbol{e}_2) \subset V \oplus Z$.

More formally, let $(\boldsymbol{e}_1, \boldsymbol{e}_2, \boldsymbol{e}_3) \in \mathbb{F}_{q^m}^{n+n_1+n}$ be a non-homogeneous error of rank weight (w_1, w_2), see Definition 19; one is given a noisy codeword $\boldsymbol{y} = \boldsymbol{c} + \boldsymbol{e}$ where $\boldsymbol{c}$ belongs to an $[2n+n_1, n_1]_{\mathbb{F}_{q^m}}$-code $\mathcal{C}$, let us assume that $\boldsymbol{y}$ is uniquely decodable, i.e. there is a single solution to this **NHRD** instance with parameters (m, n, n_1, w_1, w_2).

Let S_1 be Support $((\boldsymbol{e}_1, \boldsymbol{e}_3))$, and let S_2 be Support $(\boldsymbol{e}_2)$. This means that S_1 and S_2 are two vector spaces of $\mathbb{F}_{q^m}$ of dimension w_1 and $w_1 + w_2$, respectively. By definition, recall that $S_1 \subset S_2$.

Let $\boldsymbol{H}$ be the parity check matrix of the augmented code $\mathcal{C} + \langle \boldsymbol{y} \rangle$, this means that $\boldsymbol{H}$ is an $(n + n_1 - 1) \times (2n + n_1)$ matrix in $\mathbb{F}_{q^m}$, and that we want to solve

$$\boldsymbol{H}(\boldsymbol{e}_1, \boldsymbol{e}_2, \boldsymbol{e}_3)^\intercal = \boldsymbol{0}^{n+n_1-1}. \tag{3.33}$$

If we know two spaces V and Z in $\mathbb{F}_{q^m}$ of dimension, respectively, r and ρ, such that $S_1 \subset V$ and $S_2 \subset V \oplus Z$; then, we can express each coefficient of $\boldsymbol{e}_1$ as an $\mathbb{F}_q$-linear combination of the r elements in a basis of V. Doing so for each of its n coefficients, it means that we can write $\boldsymbol{e}_1$ using nr unknowns in $\mathbb{F}_q$.

Doing the same for $\boldsymbol{e}_2$ and $\boldsymbol{e}_3$, we need a total of

$$2nr + n_1(r + \rho)$$

unknowns in $\mathbb{F}_q$.

The Equation (3.33) gives $n + n_1 - 1$ linear equations over $\mathbb{F}_{q^m}$, hence m times more equations over $\mathbb{F}_q$.

Overall, once can solve this linear system and get its single solution as long as

$$2nr + n_1(r + \rho) \leq m(n + n_1 - 1). \tag{3.34}$$

In order to get the complexity of our combinatorial attack, we need to know the probability that two random spaces contain the secret spaces; more precisely: if one picks a subspace V of dimension $r \geq w_1$ and a random subspace $Z \subset \mathbb{F}_{q^m}/V$ of dimension $\rho \in \{w_2..m - r\}$, what is the probability that $S_1 \subset V$ and $S_2 \subset V \oplus Z$?

This probability is approximately

$$p := q^{(w_1+w_2)(r-m)+w_2\rho},$$

the proof is given in the appendix of [BBBG22].

Last but not least, we can use the same trick as in [AGHT18] to raise this probability. More precisely, one can take advantage of the $\mathbb{F}_{q^m}$-linearity by considering the probability

$$\mathbb{P}_{V,Z}\left(\exists \alpha \in \mathbb{F}_{q^m}^*,\ \alpha S_1 \subset V,\ \alpha S_2 \subset V \oplus Z\right) \approx \frac{q^m - 1}{q - 1} p. \tag{3.35}$$

In other words, picking V and Z such that

$$\alpha S_1 \subset V,\ \alpha S_2 \subset V \oplus Z$$

enables one to decode the word $\alpha \boldsymbol{e}$ instead of $\boldsymbol{e}$.

Overall, the complexity of our combinatorial attack against **NHRD**, in terms of operations over $\mathbb{F}_q$, and discarding the polynomial terms that are not relevant for us, is given by:

$$\widetilde{\mathcal{O}}\left(q^{(w_1+w_2)(m-r)-w_2\rho-m}\right) \tag{3.36}$$

where $r, \rho \in \mathbb{N}$ are chosen to maximize $(w_1 + w_2)r + w_2\rho$ under the following constraints:

$$\begin{cases} w_1 \leq r, \\ w_2 \leq \rho, \\ r + \rho \leq m - 1 \\ (2n + n_1)r + n_1\rho \leq m(n + n_1 - 1). \end{cases} \tag{3.37}$$

Remark 18. *The optimization problem given by Equation (3.37) is over the integers and there is a finite and "small" set of possible pairs (r, ρ) when given cryptographic parameters (m, n, k, w_1, w_2); thus it is really easy to solve it by performing an exhaustive search over all these pairs and taking the one maximizing the quantity $(w_1 + w_2)r + w_2\rho$.*

3.3.1.2 Algebraic attack against NHRD.

In this section, we adapt the algebraic attack against **RD** [BBC+20], see Section3.2, to the case of non-homogeneous errors.

Recall that the core of the algebraic attack **RD**, namely MaxMinors, is to consider the system made of the maximal minors of the matrix $\boldsymbol{C}\boldsymbol{H}^\intercal$, then one linearizes looking at the maximal minors of $\boldsymbol{C}$ as new variables, see Section 3.2.1 for more details.

This approach would obviously work against an **NHRD**(m, n, n_1, w_1, w_2) instance, however one would have to see the non-homogeneous error $\boldsymbol{e}$ as a vector of rank weight $w_1 + w_2$, which would lead to an inefficient attack.

In the case of a non-homogeneous error $\boldsymbol{e} = (\boldsymbol{e}_1, \boldsymbol{e}_2, \boldsymbol{e}_3) \in \mathbb{F}_q^{n+n_1+n}$, one has

$$\beta \boldsymbol{S}\boldsymbol{C}\boldsymbol{H}^\intercal = 0$$
$$\text{where } \boldsymbol{S} = (\boldsymbol{S}_1 \,|\boldsymbol{S}_2) \in \mathbb{F}_q^{m\times(w_1+w_2)}, \boldsymbol{C} = \begin{pmatrix} \boldsymbol{C}_1 & \boldsymbol{C}_2 & \boldsymbol{C}_3 \\ 0 & \boldsymbol{C}'_2 & 0 \end{pmatrix} \in \mathbb{F}_q^{(w_1+w_2)\times(n+n_1+n)}$$
$$\text{with } \boldsymbol{C}_1,\ \boldsymbol{C}_3 \in \mathbb{F}_q^{w_1\times n},\ \boldsymbol{C}_2 \in \mathbb{F}_q^{w_1\times n_1},\ \boldsymbol{C}'_2 \in \mathbb{F}_q^{w_2\times n_1},\ \boldsymbol{S}_1 \in \mathbb{F}_q^{m\times w_1},\ \boldsymbol{S}_2 \in \mathbb{F}_q^{m\times w_2}. \tag{3.38}$$

In the factorization of $\boldsymbol{e}$ given by Equation (3.38), the columns of the matrix $\boldsymbol{S}_1$ gives a basis of Support $((\boldsymbol{e}_1, \boldsymbol{e}_3))$, and the columns of the matrix $\boldsymbol{S}$ gives a basis of Support $(\boldsymbol{e}_2)$; recall that Support $((\boldsymbol{e}_1, \boldsymbol{e}_3)) \subset$ Support $(\boldsymbol{e}_2)$.

The advantage of writing the matrix $\boldsymbol{C}$ as in Equation (3.38) is that

$$M := \sum_{i=0}^{w_2-1} \binom{n_1}{i} \binom{2n}{w_1+w_2-i} \tag{3.39}$$

of its maximal minors will be zero; basically they are the minors $|\boldsymbol{C}|_{*,T}$ such that $T \cap \{n+1..n+n_1\} \leq w_2 - 1$.

Since the maximal minors of $\boldsymbol{C}$ are the variables in the MaxMinors modeling, it yields a more efficient algebraic attack which does take into account the non-homogeneous structure.

Moreover, in order to get an hybrid algebraic attack like we did in Section 3.2.2.2, one can guess entries of a columns in the matrix $\boldsymbol{C}$.

One more time, it is possible to take advantage of the particular structure of $\boldsymbol{C}$ given by Equation (3.38) by fixing a columns containing only w_1 non-zero coordinates.

This leads to a smaller exponential factor of q^{aw_1} in the final cost, instead of the naive $q^{a(w_1+w_2)}$.

This algebraic attack against **NHRD** was first described in 2020, in the Second Round update of RQC [AAB+20]; then a more thorough analysis of its complexity was proposed in [BBBG22], see Theorem 5.

Theorem 5 (Algebraic attack against **NHRD**. [BBBG22], Theorem 5, page 17)**.** *Let $a \geq 0$ be the smallest integer such that*

$$\mathcal{N}_{\mathbb{F}_q} \geq \binom{2n + n_1 - a}{w_1 + w_2} - M_a - \nu_{\mathbb{F}_q} - 1,$$

where

$$\mathcal{N}_{\mathbb{F}_q} = m \sum_{i=w_2}^{w_1+w_2} \binom{n_1 - 1}{i} \binom{n}{w_1 + w_2 - i}, \quad \nu_{\mathbb{F}_q} = m \binom{n_1 - 1}{w_2 - 1} \binom{n - 1}{w_1},$$

and

$$M_a := \sum_{i=0}^{\omega_2 - 1} \binom{n_1}{i} \binom{2n - a}{\omega_1 + \omega_2 - i}.$$

The hybrid MaxMinors attack adapted to **NHRD** *costs*

$$\mathcal{O}\left(q^{a w_1} \mathcal{N}_{\mathbb{F}_q} \left(\tbinom{2n+n_1-a}{w_1+w_2} - M_a - \nu_{\mathbb{F}_q} \right)^{\omega - 1} \right)$$

operations in $\mathbb{F}_q$, where ω is a linear algebra constant.

3.3.2 Attacks against the RSL problem

In this section, we consider an **RSL**(m, n, k, r, N)-instance, say N distinct **RD** instances whose errors share the same support of dimension r, see Definition 18.

This number N is a crucial parameter to estimate the hardness of **RSL**, in particular to compare it to **RD**.

For instance, this problem can be solved in polynomial time when $N \geq nr$, due to [GHPT17]. A more powerful attack was later found in [DT18]; it suggests that secure **RSL** instances must satisfy a stronger condition: $N < kr$.

In what follows, we give a new combinatorial attack against **RSL**, it is more efficient than the previous combinatorial attacks, plus it enables us to decrease the threshold on N where the **RSL** problem starts to be solvable in polynomial time. In addition to this, we give more explicit formulas to clarify the recent algebraic attack of [BB21].

Last but not least, we propose a graph so that one could visualize how the hardness of an **RSL** instance changes when N grows.

3.3.2.1 New combinatorial attack on RSL.

Theorem 6 (Combinatorial attack on **RSL**). *There exists a combinatorial attack against an* $\mathbf{RSL}(m, n, k, r, N)$ *instance whose complexity is*

$$\widetilde{\mathcal{O}}\left(q^{r\left(m-\left\lfloor\frac{m(n-k)-N}{n-a}\right\rfloor\right)}\right)$$

operations in $\mathbb{F}_q$*, where* $a := \left\lfloor\frac{N}{r}\right\rfloor$.

Proof. Let $\boldsymbol{s}_i \in \mathbb{F}_{q^m}^{n-k}$, $1 \leq i \leq N$, denote the N syndromes from an $\mathbf{RSL}(m, n, k, r, N)$ instance.

By definition, there exist

$$\boldsymbol{e}_i \in \mathbb{F}_{q^m}^n,\ \|\boldsymbol{e}_i\| = r,\ \boldsymbol{H}\boldsymbol{e}_i^\intercal = \boldsymbol{s}_i^\intercal,\ \forall\ i \in \{1, 2, \ldots, N\},$$

where $\boldsymbol{H} \in \mathbb{F}_{q^m}^{(n-k)\times n}$ is a parity-check matrix, and where $\mathrm{Support}(\boldsymbol{e}_i)$ does not depend on i.

Similarly to [GRSZ14a, BB21], this last property enables us to use the fact that there exists an $\mathbb{F}_q$-linear combination $(\ \mathbf{0}^a \mid \widetilde{\boldsymbol{e}}\ ) \in \mathbb{F}_{q^m}^n$ of the $\boldsymbol{e}_i$'s which is all-zero on its first $a := \left\lfloor\frac{N}{r}\right\rfloor$ coordinates.

This error corresponds to a *secret* linear combination of the syndromes, more precisely

$$\exists \lambda_1, \lambda_2, \ldots, \lambda_N \in \mathbb{F}_q, \quad \boldsymbol{H}(\ \mathbf{0}^a \mid \widetilde{\boldsymbol{e}}\ )^\intercal = \sum_{i=1}^{N} \lambda_i \boldsymbol{s}_i^\intercal.$$

By setting $\widetilde{\boldsymbol{H}} := \boldsymbol{H}_{*,\{a+1\ldots n\}}$, this is equivalent to

$$\widetilde{\boldsymbol{H}}\widetilde{\boldsymbol{e}}^\intercal = \sum_{i=1}^{N} \lambda_i \boldsymbol{s}_i^\intercal. \tag{3.40}$$

Similarly to what was done in Section 3.3.1.1, we can pick a vector space $V \subset \mathbb{F}_{q^m}$ of dimension $r_1 \geq r$ such that $\mathrm{Support}(\widetilde{\boldsymbol{e}}) \subset V$.

Knowing such a vector space V, Equation (3.40) can be seen as $n - k$ linear equations over $\mathbb{F}_{q^m}$ involving:

- $(n - a)r_1$ unknowns in $\mathbb{F}_q$ to write each coefficient of $\widetilde{\boldsymbol{e}}$ in a basis of V,
- N unknowns over $\mathbb{F}_q$, namely the λ_i's.

Note that since all the unknowns are in $\mathbb{F}_q$, one can project the equations over $\mathbb{F}_q$ in order to get a total of $m(n-k)$ equations.

Thus, one can expect to solve this linear system as long as:

$$(n-a)r_1 + N \leq m(n-k) \quad \Longrightarrow \quad r_1 \leq \frac{m(n-k)-N}{n-a}. \tag{3.41}$$

Using the result from [GRS16] that we described at the beginning of Section 3.3.1.1, the probability that a vector space V of dimension r_1, taken uniformly at random in $\mathbb{F}_{q^m}$, contains the support of $\widetilde{\boldsymbol{e}}$ which has dimension r can be approximated by:

$$q^{-r(m-r_1)}.$$

In order to maximize this probability, according to the constraint given by Equation (3.41), one sets

$$r_1 = \left\lfloor \frac{m(n-k)-N}{n-a} \right\rfloor,$$

hence the result. □

Thanks to Theorem 6, we can derive a new bound on N so that an **RSL** instance with N syndromes can be solved in polynomial time:

Corollary 2 (Polynomial bound)**.** *Using the attack of Theorem 6, an* **RSL** *instance with parameters* (m, n, k, r, N) *can be solved in polynomial time as long as*

$$N > kr\frac{m}{m-r}.$$

Proof. The exponential factor in the cost of the attack given by Theorem 6 is

$$q^{r(m-\delta)}$$

where

$$\delta := \left\lfloor \frac{m(n-k)-N}{n-a} \right\rfloor, \text{ and } \quad a := \left\lfloor \frac{N}{r} \right\rfloor.$$

Since $r > 0$ by definition, this attack will be polynomial as long as $\delta > m$.

Without loss of generality, let us assume that both $\frac{N}{r}$ and δ are integers.
One readily has that $\delta > m$ is equivalent to

$$N > kr\frac{m}{m-r},$$

hence the result. □

Note that as long as

$$\frac{m}{m-r} < \frac{n}{k},$$

which is almost always the case for cryptographic parameters, our bound is lower than the former one given in [GHPT17], namely $N > nr$.

3.3.2.2 Algebraic Attack of [BB21].

This attack consists in solving a bilinear system at some bi-degree $(b, 1)$ for $b \geq 1$ by using linearization techniques such as the ones used in [BBC+20], see Section 2.3.2 for more details.

The two cases "$\delta = 0$" and "$\delta > 0$" presented below correspond to two different specializations of this bilinear system which lead to different costs.

In what follows, we provide explicit formulas to compute these two complexities (for the binary field $\mathbb{F}_2$).

In particular, we also include the values of α_R and α_λ which correspond to the hybrid approach mentioned in [BB21].

Finally, note that these formulas are valid only when $N > n - k - r$.

First case: $\delta = 0$. Let a be the unique integer such that

$$ar < N \leq (a+1)r,$$

and set $N' := ar + 1$.

For $1 \leq b \leq r + 1$, the number of variables for linearization is

$$\mathcal{M}_{\leq b}^{\mathbb{F}_2} := \sum_{i=1}^{b} \binom{n-a-\alpha_R}{r} \binom{N'-\alpha_\lambda}{i}, \tag{3.42}$$

where $0 \leq \alpha_R < n - a - r$, and $0 \leq \alpha_\lambda < N' - b$.

The number of linearly independent equations is equal to $m\mathcal{N}_{\leq b}^{\mathbb{F}_2}$ where

$$\mathcal{N}_{\leq b}^{\mathbb{F}_2} := \sum_{i=1}^{b} \sum_{d=1}^{i} \sum_{j=1}^{n-k} \binom{j-1}{d-1} \binom{n-k-j}{r-d+1} \binom{N'-\alpha_\lambda-j}{i-d}. \tag{3.43}$$

The complexity is given by

$$\mathcal{O}\left(\min\left(2^{r\alpha_R+\alpha_\lambda} m\mathcal{N}_{\leq b}^{\mathbb{F}_2} (\mathcal{M}_{\leq b}^{\mathbb{F}_2})^{\omega-1}, \right.\right. \\ \left.\left. 2^{r\alpha_R+\alpha_\lambda}(N'-\alpha_\lambda)\binom{k-a+1+r}{r}(\mathcal{M}_{\leq b}^{\mathbb{F}_2})^2 \right)\right) \quad (3.44)$$

provided that $m\mathcal{N}_{\leq b}^{\mathbb{F}_2} \geq \mathcal{M}_{\leq b}^{\mathbb{F}_2} - 1$, and where the values of b, α_R, and α_λ are chosen to minimize the complexity.

Second case: $\delta > 0$**.** Let δ be a positive integer such that

$$N \geq \delta(n-r+\delta),$$

let a be the greatest integer such that

$$N > \delta(n-r+\delta) + a(r-\delta),$$

and set $N' := \delta(n-r+\delta) + a(r-\delta)$.

To find the complexity of this attack, one replaces r by $r-\delta$ in the expressions of $\mathcal{M}_{\leq b}^{\mathbb{F}_2}$ and $\mathcal{N}_{\leq b}^{\mathbb{F}_2}$, respectively in Equations (3.42) and (3.43).

The complexity is finally obtained with Equation (3.44), and its minimal value now depends on $\delta > 0$ as well as on b, α_R, α_λ as above.

3.3.2.3 Visualization of the attacks against RSL.

Last but not least, in order to visualize how the number of syndromes N affects the complexity of an **RSL** instance, we drew a graph of the complexities of different attacks with respect to N.

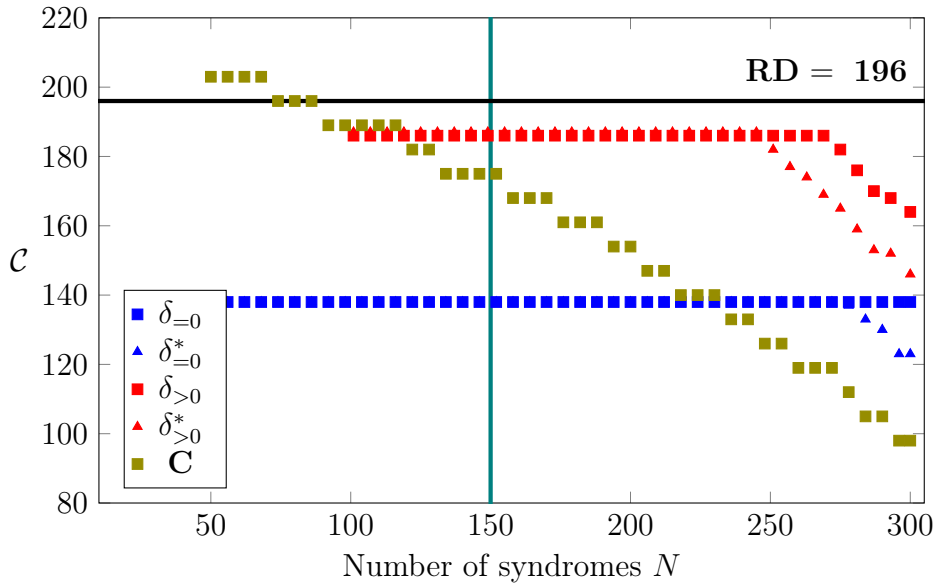
More precisely, on Figure 3.1, the x-axis indicates the value of N, the number of syndromes of an **RSL** instance with parameters

$$[m, n, k, r, N] = [61, 100, 50, 7, N],$$

and the y-axis gives the complexity of different attacks.

The blue/red squares and triangles refer to the different cases of the algebraic attack described in Section 3.3.2.2, and the green squares refer to our new combinatorial attack described in Section 3.3.2.1.

Figure 3.1: Complexity $\mathcal{C}$ (in bits) of the best known attacks against an **RSL** instance with parameters $[m, n, k, r] = [61, 100, 50, 7]$ in terms of the number $N > n - k - r$ of syndromes. In the legend: **C** stands for our combinatorial attack (see Theorem 6), all the other symbols correspond to the 2 cases of the algebraic attack [BB21] where the "*" indicates the use of Wiedemann algorithm instead of Strassen's.



The complexity to solve the associated **RD** instance, i.e. taking only $N = 1$ syndrome, using the algebraic attack MaxMinors, see Section 3.2, is 196 bits; it corresponds to the horizontal black thick line on Figure 3.1.

Starting with $N = 44$ syndromes, recall that it is the threshold for the algebraic attack against **RSL** (see Section 3.3.2), one sees that it already beats the **RD** attack.

It is worth noticing that with approximately $N = 225$ syndromes, our new combinatorial attack against **RSL**, see Section 3.3.2.1, starts to beat the algebraic attack of [BB21].

Finally, one notices that, with a lot of syndromes, all the aforementioned attacks complexities drop down, which is quite logical.

The parameters $[m, n, k, r] = [61, 100, 50, 7]$ were not chosen at random: they precisely correspond to attacking our scheme NH-Multi-RQC-AG-128, described in Section 4.2.4, see Table 4.1 for the parameters.

More details are given about the links between Figure 3.1 and our new scheme in Section 4.2.7.

3.3.3 Attack against the NHRSL problem

In this section, we adapt the combinatorial attack against **RSL**, given in the proof of Theorem 6, to the case of non-homogeneous errors, i.e. to the **NHRSL** problem, see Definition 20.

For the sake of simplicity, and since it is the case for all cryptographic parameters studied in this document, we focus only on **NHRSL** instances where $n_1 < n$.

Theorem 7 (Combinatorial attack against **NHRSL**)**.** *There exists a combinatorial attack against an* **NHRSL** *instance with parameters*

$$(m, n, n_1, w_1, w_2, N)$$

whose complexity, in terms of elementary operations in $\mathbb{F}_q$*, is given by*

$$\widetilde{\mathcal{O}}\left(q^{(w_1+w_2)(m-r)-w_2\rho}\right),$$

where r, ρ *are integers chosen to maximize the quantity*

$$(w_1 + w_2)r + w_2\rho$$

under the following constraints:

$$\begin{cases} w_1 \leq r, \\ w_2 \leq \rho, \\ r + \rho \leq m - 1, \\ N_1, N_2 \in \mathbb{N}, \\ N_1 + N_2 = N, \\ a := \left\lfloor \frac{N_1}{w_1} \right\rfloor \leq n_1, \\ b := \left\lfloor \frac{N_2}{w_1+w_2} \right\rfloor \leq 2n, \\ m(n + n_1) \geq (n_1 - b)(r + \rho) + (2n - a)r + N. \end{cases} \tag{3.45}$$

Proof. Straightforward adaptation of the attack in the proof of Theorem 6, combined with the probability results given in the appendix of [BBBG22]. □

3.4 Attacks against PSSI (Durandal signature)

Section 1.1.3.2, and the beginning of Section 3.5 give an overview of rank-based signature schemes in general, and about Durandal [ABG+19] in particular.

In this section, we will only focus on the Product Spaces Subspaces Indistinguishability (**PSSI**) problem, see Definition 40, which is central for the security of Durandal signature scheme.

First, we will give a randomized reduction from **PSSI** to **MinRank**, see Definition 16; then we will use this reduction to give some complexity estimations.

Throughout this section, $\mathbf{Gr}(d, \mathbb{F}_{q^m})$ refers to the set of the $\mathbb{F}_q$-vector subspaces in $\mathbb{F}_{q^m}$ of dimension d. Hence, $V \xleftarrow{\$} \mathbf{Gr}(d, \mathbb{F}_{q^m})$ means that $V \subset \mathbb{F}_{q^m}$ is a vector space picked uniformly at random among the vector spaces of dimension d in $\mathbb{F}_{q^m}$.

3.4.1 The PSSI problem

Before introducing the **PSSI** problem, we need the two following definitions.

Definition 38 (PSSI distribution $\mathcal{D}_{\mathsf{PSSI}}$)**.** *Let E be a $\mathbb{F}_q$-subspace of $\mathbb{F}_{q^m}$ of dimension r. Let $\mathcal{D}_{\mathsf{PSSI}}(E)$ be the distribution that outputs N samples $(F_i, Z_i)_{1 \leq i \leq N}$ defined as follows:*

- $F_i \xleftarrow{\$} \mathbf{Gr}(d, \mathbb{F}_{q^m})$
- $U_i \xleftarrow{\$} \mathbf{Gr}(rd - \lambda, EF_i)$
- $W_i \xleftarrow{\$} \mathbf{Gr}(w, \mathbb{F}_{q^m})$
- $Z_i = W_i + U_i$

The pairs of subspaces $(F_i, W_i)_{1 \leq i \leq N}$ are picked independently. When there is no ambiguity on the vector space E, we use $\mathcal{D}_{\mathsf{PSSI}}$ as a shorthand.

Definition 39 (Random distribution $\mathcal{D}_{\mathsf{Random}}$)**.** *Let $\mathcal{D}_{\mathsf{Random}}$ the distribution that outputs N samples $(F_i, Z_i)_{1 \leq i \leq N}$ where F_i and Z_i are independent random variables picked uniformly in, respectively, $\mathbf{Gr}(d, \mathbb{F}_{q^m})$ and $\mathbf{Gr}(w + rd - \lambda, \mathbb{F}_{q^m})$.*

The **PSSI** problem, see Definition 40, consists in distinguishing between random spaces of dimension $w + rd - \lambda$ and subspaces of codimension λ in EF_i (where F_i is public and E is a fixed secret space) masked with random subspaces of dimension w.

Definition 40 (Product Spaces Subspaces Indistinguishability (**PSSI**) problem)**.** *Let E be a $\mathbb{F}_q$-subspace of $\mathbb{F}_{q^m}$ of dimension r. The* **PSSI** *problem with parameters (m, r, d, λ, w, N) consists in distinguishing samples $(Z_i, F_i)_{i\in\{1..N\}} \xleftarrow{\$} \mathcal{D}_{\mathsf{PSSI}}(E)$ and samples $(Z_i, F_i)_{i\in\{1..N\}} \xleftarrow{\$} \mathcal{D}_{\mathsf{Random}}$.*

Remark 19. *In the original paper of Durandal [ABG+19], the* **PSSI** *problem was defined with a mandatory filtering on U_i, i.e. $\{ef, e \in E, f \in F_i\} \cap U_i = \{0\}$. In this version, we define* **PSSI** *as a more general problem in which weak instances are those with a subspace U_i for which $\{ef, e \in E, f \in F_i\} \cap U_i \neq \{0\}$. More details about these weak instances are given right below.*

Weak instances. The **PSSI** problem is related to the decoding of LRPC codes [GMRZ13]. Indeed we can consider a subspace $Z = U + W$ as the noisy support of a syndrome for an LRPC code, the noise corresponding to W. Consequently, it is natural to try and apply techniques used for decoding LRPC codes in order to solve the **PSSI** problem.

The first idea is to use the basic decoding algorithm, see [GMRZ13]. It consists in computing intersections I of the form $f^{-1}Z \cap f'^{-1}Z$ with $(f, f') \in F^2$.

This technique only works when U contains product elements of the form ef with $e \in E$ and $f \in F$. We categorize these occurrences as *weak instances*. A strong instance of **PSSI** is thus an instance in which all the subspaces U_i are *filtered*, meaning that $\{ef, e \in E, f \in F_i\} \cap U_i = \{0\}$.

Clearly, in the Durandal signature scheme, only strong **PSSI** instances are used.

3.4.2 Reduction to MinRank

In this section, we will need the following assumption:

Assumption 1. *For integers r, d, w, λ of the same order of magnitude as the parameters in Table 3.4. When a pair (Z_i, F_i) is picked uniformly at random in*

$$\mathbf{Gr}(w + rd - \lambda, \mathbb{F}_{q^m}) \times \mathbf{Gr}(d, \mathbb{F}_{q^m}),$$

then there does not exist, with overwhelming probability, a vector space E such that

$$Z_i = W_i + U_i, \quad \text{where } U_i \subset EF_i, \quad \dim(U_i) = rd - \lambda,$$

for any space W_i of dimension w.

Roughly, Assumption 1 states that a vector space picked uniformly at random will correspond to a **PSSI** instance with a negligible probability. This assumption has been supported by our experiments.

Proposition 7. *There is a randomized polynomial-time reduction from* **PSSI** *to* **MinRank**. *More precisely, there exists a probabilistic polynomial-time algorithm $\mathcal{A}$ such that:*

- $(Z_i, F_i) \leftarrow \mathcal{D}_{\mathsf{PSSI}} \implies \mathbb{P}(\mathsf{MinRank.Solve}(\mathcal{A}(Z_i, F_i)) = 1) \geq 1/4$,
- $(Z_i, F_i) \leftarrow \mathcal{D}_{\mathsf{Random}} \implies \mathbb{P}(\mathsf{MinRank.Solve}(\mathcal{A}(Z_i, F_i)) = 0) \geq 1 - \varepsilon$, *where ε is negligible.*

Proof. Note that in the following reduction, one only uses a single couple (Z_i, F_i), that is to say that $i \in \{1..N\}$ is fixed all along this proof.

Let (Z_i, F_i) be an instance of **PSSI**, i.e. coming either from $\mathcal{D}_{\mathsf{PSSI}}$ or $\mathcal{D}_{\mathsf{Random}}$, see Definitions 38 and 39.

Let $\boldsymbol{Z}_i$ be an $m \times (w + rd - \lambda)$ matrix with entries in $\mathbb{F}_q$ whose columns form a basis of the vector space Z_i written over $\mathbb{F}_q$.

Let $\widetilde{\boldsymbol{E}}$ be an $m \times r$ matrix containing $(m - r)r$ variables $\widetilde{e_{i,j}}$ with $1 \leq i \leq r$, and $r + 1 \leq j \leq m$, and an identity block as on Equation (3.46):

$$\widetilde{\boldsymbol{E}} := \left(\begin{array}{ccc} & \boldsymbol{I}_r & \\ \hline \widetilde{e_{1,r+1}} & \cdots & \widetilde{e_{r,r+1}} \\ \widetilde{e_{1,r+2}} & \cdots & \widetilde{e_{r,r+2}} \\ \vdots & \vdots & \vdots \\ \widetilde{e_{1,m}} & \cdots & \widetilde{e_{r,m}} \end{array}\right). \tag{3.46}$$

Without loss of generality, we put the identity block at the first $r \times r$ spot in the matrix $\widetilde{\boldsymbol{E}}$, see the remark at the very end of this proof.

This matrix can be seen as a symbolic basis over $\mathbb{F}_q$ of a vector space of dimension at most r, namely E.

The fact that we consider this matrix $\widetilde{\boldsymbol{E}}$ in systematic form ensures that its column space is of dimension exactly r.

From the matrix $\widetilde{\boldsymbol{E}}$, since the vector space F_i is known, one can easily build a symbolic basis matrix $\widetilde{\boldsymbol{EF}_i}$ of the product vector space EF_i; this $m \times rd$ matrix has entries which are polynomials of degree at most 1 in the polynomial ring

$$\mathbb{F}_q[\widetilde{e_{1,r+1}}, \widetilde{e_{1,r+2}}, \ldots, \widetilde{e_{r,m-1}}, \widetilde{e_{r,m}}].$$

Note that these polynomials might have non-zero constant term due to the systematic form of the matrix $\widetilde{\boldsymbol{E}}$. Let

$$\widetilde{\boldsymbol{M}} := \left(\boldsymbol{Z}_i \mid \widetilde{\boldsymbol{EF}_i}\right)$$

be the matrix obtained by appending horizontally $\widetilde{\boldsymbol{EF}_i}$ to $\boldsymbol{Z}_i$; $\widetilde{\boldsymbol{M}}$ is an $m \times (w + rd - \lambda + rd)$ matrix with entries in $\mathbb{F}_q[\widetilde{e_{1,r+1}}, \widetilde{e_{1,r+2}}, \ldots, \widetilde{e_{r,m-1}}, \widetilde{e_{r,m}}]$.

We define the matrix $\boldsymbol{M}_0 \in \mathbb{F}_q^{m \times (w+rd-\lambda+rd)}$, and for $1 \leq i \leq r$, and $r+1 \leq j \leq m$, the matrices $\boldsymbol{M}_{i,j} \in \mathbb{F}_q^{m \times (w+rd-\lambda+rd)}$ such that:

$$\widetilde{\boldsymbol{M}} = \boldsymbol{M}_0 + \sum_{i=1}^{r} \sum_{j=r+1}^{m} \widetilde{e_{i,j}} \boldsymbol{M}_{i,j}.$$

These matrices are uniquely defined since they contain, for each corresponding entry in $\widetilde{\boldsymbol{M}}$, the coefficient of the $\widetilde{e_{i,j}}$ variables, and $\boldsymbol{M}_0$ contains the constant term.

These matrices form the **MinRank** instance, see Definition 16, i.e.

$$\mathcal{A}(Z_i, F_i) := (\boldsymbol{M}_0, \boldsymbol{M}_{1,1}, \boldsymbol{M}_{1,2}, \ldots, \boldsymbol{M}_{r,m-1}, \boldsymbol{M}_{r,m}),$$

and its parameters are given by:

$$(m, w + 2rd - \lambda, (m-r)r + 1, w + rd).$$

The reason for the choice of the target rank, namely $w + rd$, is explained below.

We will now prove that $\mathcal{A}$ is a suitable algorithm for the reduction of **PSSI** to **MinRank**, by considering the two cases where (Z_i, F_i) comes from either $\mathcal{D}_{\mathsf{PSSI}}$ or $\mathcal{D}_{\mathsf{Random}}$.

First case. Let us assume that (Z_i, F_i) comes from the **PSSI** distribution generated by some subspace E, i.e. $\mathcal{D}_{\mathsf{PSSI}}(E)$.

Since the vector space E is picked uniformly at random, there is a probability bounded from below by $1/4$ that the first $r \times r$ block in a basis matrix of E, written over $\mathbb{F}_q$, is non-singular, see for instance the aforementioned Lemma 7 in [BGL03].

So, let $\boldsymbol{E}$ be a basis matrix of E of the following form:

$$\boldsymbol{E} := \begin{pmatrix} & & \\ & \boldsymbol{I}_r & \\ & & \\ \hline & & \\ e_{1,r+1} & \cdots & e_{r,r+1} \\ e_{1,r+2} & \cdots & e_{r,r+2} \\ \vdots & \vdots & \vdots \\ e_{1,m} & \cdots & e_{r,m} \end{pmatrix} \in \mathbb{F}_q^{m \times r}, \tag{3.47}$$

Similarly to the previous transformation, since the vector space F_i is known, one can easily build a basis matrix $\boldsymbol{EF}_i$ of the product vector space EF_i using the basis of E in a systematic form.

The matrix defined as

$$\boldsymbol{M} := (\boldsymbol{Z}_i \mid \boldsymbol{EF}_i)$$

has a rank less or equal to $w + rd$. The proof is straightforward from the fact that $U_i \subset EF_i$.

It is also readily seen that the matrix $\boldsymbol{M}$ corresponds to the evaluation of the matrix $\widetilde{\boldsymbol{M}}$ with the variable assignments $\widetilde{e_{i,j}} \leftarrow e_{i,j}$. In other words,

$$\boldsymbol{M} = \boldsymbol{M}_0 + \sum_{i=1}^{r} \sum_{j=r+1}^{m} e_{i,j} \boldsymbol{M}_{i,j}.$$

This shows that the **MinRank** instance given by $\mathcal{A}(Z_i, F_i)$ admits a solution

$$(e_{1,r+1}, e_{1,r+2}, \ldots, e_{r,m-1}, e_{r,m}),$$

and consequently, the solver MinRank.Solve will return 1.

Due to the probability that the first $r \times r$ block is not singular, one gets the first probability of Proposition 7.

Second case. Let us now assume that (Z_i, F_i) comes from the uniform distribution, i.e. $\mathcal{D}_{\mathsf{Random}}$.

By Assumption 1, using the same notation as in Definition 40, with overwhelming probability, there does not exist a solution vector space E of dimension r such that $Z_i = W_i + U_i$.

Let $\mathcal{M} = \mathcal{A}(Z_i, F_i)$ be the **MinRank** instance built using the procedure described above.

If $\mathcal{M}$ has a solution, by construction and more specifically thanks to the systematic form, it is readily guaranteed that there exists a solution E of dimension r to the associated **PSSI** instance.

Since by hypothesis no such solution vector space exists, it means that the **MinRank** instance $\mathcal{M}$ will not have any solution. Hence, MinRank.Solve is bound to return 0.

The second probability in Proposition 7 simply comes from the $1 - \varepsilon$ probability in Assumption 1.

Turning a **PSSI** instance into a **MinRank** instance following the aforementioned procedure is probabilistic polynomial-time because the $r \times r$ systematic block is chosen at random, see Equation (3.46), and all the subsequent computations are clearly polynomial. This concludes the proof. □

Remark 20. *Recall that Assumption 1 is supported by experiments. Furthermore, recall that the* **MinRank** *instance coming from the reduction in the proof of Proposition 7 has parameters*

$$(m, w + 2rd - \lambda, (m - r)r + 1, w + rd).$$

By applying these parameters to the bound given by Definition 22, one gets the condition:

$$(m - r)r + 1 < (m - (w + rd))(w + 2rd - \lambda - (w + rd)). \tag{3.48}$$

And it is worth noticing that this last condition is always fulfilled for parameters of interest, see for instance Table 3.4, which is consistent with Assumption 1.

3.4.3 Application to Durandal parameters

Table 3.4 gives the parameters of the two **PSSI** instances associated to Durandal sets of parameters [ABG+19], and the complexities to attacking a **MinRank** instance coming from the aforementioned reduction. Note that the algebraic attacks we considered are the ones described in Section 3.1.

m	$r = d$	w	λ	[ABG+19]	**MinRank** param	Comb.	Alg.
241	6	57	12	193	$(241, 117, 1411, 93)$	558	803
263	7	56	14	193	$(263, 140, 1793, 105)$	736	990

Table 3.4: **PSSI** parameters where for each set $q = 2$, and the targeted level of security is 128 bits. The column "[ABG+19]" refers to the complexity of the distinguisher against **PSSI** described in this paper, the column "Comb." (resp. "Alg.") refers to the combinatorial attacks (resp. the algebraic attacks) against generic **MinRank**.

One notices that the reduction does not enable one to beat the previous attack against **PSSI** using state-of-the-art attack against generic **MinRank** instances. However the comparison is somehow biased for the reasons explained below.

It is worth noticing that the previous attack against **PSSI** in [ABG+19] is a *distinguisher* that does not enable one to retrieve the secret space E unlike the **MinRank** attack.

Moreover, as mentioned in Chapter 6, there is a lot of room for improvements. First, these **MinRank** instances are very structured and one could take advantage of that by adapting the **MinRank** attacks. Second, the distinguisher makes use of several connected **PSSI** instances, whereas here we only used one in the reduction to **MinRank**.

3.5 Cryptanalysis of the RPS signature scheme

As seen in Section 1.1, code-based signature schemes can belong to two categories: the hash-and-sign schemes and the proof of knowledge ones. The latter is due to Schnorr-Lyubashevsky [Sch91, Lyu09].

In the rank-metric, it gave rise to Durandal [ABG+19] and RPS [LT20b] signature schemes.

The main idea is the following: the public key consists of a random matrix $\boldsymbol{H}$ and a matrix $\boldsymbol{T} = \boldsymbol{HS}$ where $\boldsymbol{S}$ is a secret matrix of low weight syndromes. To prove the knowledge of $\boldsymbol{S}$, the signer outputs a signature consisting of a challenge $\boldsymbol{c}$ and a vector $\boldsymbol{z} = \boldsymbol{y} + \boldsymbol{cS}$. The idea is that $\boldsymbol{y}$ acts as a mask that hides the secret value $\boldsymbol{cS}$.

While the Durandal scheme reuses the same secret matrix $\boldsymbol{S}$ across all signatures and checks that the techniques used in the decoding algorithm of the LRPC codes do not leak information, the RPS scheme uses ephemeral keys in order to randomize this matrix for each signature.

3.5.1 Presentation of the scheme

Let us describe the RPS scheme from [LT20b].

In what follows, $\mathcal{S}_n^w$ denotes the set of vectors of length n and weight w over $\mathbb{F}_{q^m}$. For the sake of clarity, we omit the dot $\cdot$ for the product between vectors of length k, see Section 2.1.3 for its definition.

Keygen:

- Sample $\boldsymbol{x} \xleftarrow{\$} \mathcal{S}_k^{r_x}$ and $\boldsymbol{y} \xleftarrow{\$} \mathcal{S}_k^{r_y}$.
- Let $\boldsymbol{h} = \boldsymbol{x}^{-1}\boldsymbol{y}$.
- Output $(pk = \boldsymbol{h}, sk = (\boldsymbol{x}, \boldsymbol{y}))$.

Sign(μ, pk, sk): for $1 \leq i \leq l$, sample:

- $\boldsymbol{e}_i \xleftarrow{\$} \mathcal{S}_k^{r_e}$
- $\boldsymbol{f}_i \xleftarrow{\$} \mathcal{S}_k^{r_f}$
- $\boldsymbol{u}_i \xleftarrow{\$} \mathcal{S}_k^{r_u}$
- $\boldsymbol{v}_i \xleftarrow{\$} \mathcal{S}_k^{r_v}$

Let $\mathcal{H}$ be a hash function which outputs values in $\mathcal{S}_k^1$ and $\boldsymbol{H}$ be the parity-check matrix generated by $(\boldsymbol{h}, \boldsymbol{h}^{-1})$. Compute:

- $\boldsymbol{s}_i = (\boldsymbol{e}_i\boldsymbol{x}, \boldsymbol{f}_i\boldsymbol{y})\boldsymbol{H}^\intercal$
- $\boldsymbol{w}_i = (\boldsymbol{u}_i\boldsymbol{x}, \boldsymbol{v}_i\boldsymbol{y})\boldsymbol{H}^\intercal$
- $\boldsymbol{c}_i = \mathcal{H}(\{\boldsymbol{w}_i, \boldsymbol{s}_i\}, \mu, pk)$
- $\boldsymbol{a}_i = (\boldsymbol{u}_i + \boldsymbol{c}_i\boldsymbol{e}_i)\boldsymbol{x}$
- $\boldsymbol{b}_i = (\boldsymbol{v}_i + \boldsymbol{c}_i\boldsymbol{f}_i)\boldsymbol{y}$

Then output $\sigma = (\{\boldsymbol{c}_i, \boldsymbol{a}_i, \boldsymbol{b}_i, \boldsymbol{s}_i\}_{1 \leq i \leq l})$.

Verify(σ, μ, pk): for $1 \leq i \leq l$, we check the rank of the following values:

- $\|\boldsymbol{a}_i\| = (r_u + r_e)r_x$
- $\|\boldsymbol{a}_i\boldsymbol{h}\| = (r_u + r_e)r_y$
- $\|\boldsymbol{b}_i\| = (r_v + r_f)r_y$
- $\|\boldsymbol{b}_i\boldsymbol{h}^{-1}\| = (r_v + r_f)r_x$
- $\|\boldsymbol{s}_i\| = r_e r_y + r_f r_x$
- $\|\boldsymbol{s}_i\boldsymbol{h}\| \geq min(m-1, k)$
- $\|\boldsymbol{s}_i\boldsymbol{h}^{-1}\| \geq min(m-1, k)$

If one of them is not valid, reject the signature. Otherwise, compute $\boldsymbol{w}_i$ as $(\boldsymbol{a}_i, \boldsymbol{b}_i)\boldsymbol{H}^\intercal - \boldsymbol{c}_i\boldsymbol{s}_i$. Accept the signature if $\boldsymbol{c}_i = \mathcal{H}(\{\boldsymbol{w}_i, \boldsymbol{s}_i\}, \mu, pk)$ for $1 \leq i \leq l$.

In [LT20b], the authors proposed the following parameter sets, where $r_x = r_y$, $r_e = r_v$ and $r_u = r_f$:

Parameter set	$(l, q, m, k, r_x, r_e, r_u)$	Security
RPS-C1	(3, 2, 61, 59, 5, 6, 5)	128 (classical)
RPS-C2	(4, 2, 67, 59, 5, 6, 5)	192 (classical)
RPS-P1	(3, 2, 89, 83, 7, 6, 5)	128 (post-quantum)
RPS-P2	(3, 2, 89, 107, 11, 4, 3)	192 (post-quantum)

Table 3.5: Parameters of the RPS signature scheme.

3.5.2 Some useful propositions

We will need several propositions about vector spaces of $\mathbb{F}_{q^m}$ or vector spaces in general, there are given in this section.

Proposition 8 (Dimension of random vectors, [AGH+19]). *Let X be a vector space of $\mathbb{F}_{q^m}$ of dimension w_r and let $\boldsymbol{x}$ be a random vector in X^n.*

The probability that $\|\boldsymbol{x}\| = w_r - i$ can be approximated by:

$$q^{-i(max(n,w_r)-min(n,w_r)+i)}$$

Lemma 2 (Intersection of 2 vector spaces)**.** *Let A and B be two vector spaces of $\mathbb{F}_{q^m}$, their dimensions fulfill the following inequality:*

$$\dim(A \cap B) \geq \dim(A) + \dim(B) - m.$$

Proof. Straightforward using Grassmann's formula: $\dim(A \cap B) = \dim(A) + \dim(B) - \dim(A + B)$. □

Lemma 2 can easily be generalized to the intersection of 3 subspaces of $\mathbb{F}_{q^m}$. The proof is straightforward and only requires to use lemma 2.

Lemma 3 (Intersection of 3 vector spaces)**.** *Let A, B, and C, be 3 vector spaces of $\mathbb{F}_{q^m}$, their dimensions fulfill the following inequality:*

$$\dim(A \cap B \cap C) \geq \dim(A) + \dim(B) + \dim(C) - 2m.$$

Lemma 4 (Random vector in vector space)**.** *Let $X \subset Y$ be two vector spaces of $\mathbb{F}_{q^m}$ of dimensions x and y, respectively. If one picks at random a vector $v \in Y$, it will belong to X with probability q^{x-y}.*

Proof. Straightforward by computing the ratio of favorable vectors over all possible vectors. □

Proposition 9. *Let $X \subset Y$ be two vector spaces of $\mathbb{F}_{q^m}$ of dimensions x and y, respectively. If one picks at random x vectors $v_1, \ldots, v_x$ in Y, $\{v_1, \ldots, v_x\}$ will be a basis of X with probability*

$$\prod_{j=0}^{x-1} q^{-y}(q^x - q^j).$$

Proof. When one picks x vectors in the vector space Y, the probability that these vectors form a basis of a subvector space of dimension x, i.e. that they are linearly independent, is

$$\prod_{j=0}^{x-1} (1 - q^{j-y}) \;=\; \prod_{j=0}^{x-1} q^{-y}(q^y - q^j). \tag{3.49}$$

There are

$$\begin{bmatrix} y \\ x \end{bmatrix}_q := \prod_{j=0}^{x-1} \frac{q^y - q^j}{q^x - q^j}$$

distinct vector spaces of dimension x in the vector space Y. Thus, each time one picks a random vector space in Y, it will be X with probability

$$\prod_{j=0}^{x-1} \frac{q^x - q^j}{q^y - q^j}. \tag{3.50}$$

One ends the proof by multiplying the two probabilities given by (3.49) and (3.50). □

3.5.3 Our attacks

3.5.3.1 Information leakage attack

In this section, we present our first attack, namely the information leakage attack. It shows that one can exploit information leakage from the signatures when m is high enough in order to recover the ephemeral keys $\boldsymbol{e}_i\boldsymbol{x}, \boldsymbol{u}_i\boldsymbol{x}, \boldsymbol{v}_i\boldsymbol{y}$ and $\boldsymbol{f}_i\boldsymbol{y}$.

Given a valid signature $\sigma = (\{\boldsymbol{c}_i, \boldsymbol{a}_i, \boldsymbol{b}_i, \boldsymbol{s}_i\}_{1\leq i\leq l})$ and using the public key $\boldsymbol{h}$, one can easily compute the three following vectors:

$$\begin{aligned} \boldsymbol{a}_i\boldsymbol{h} &= \boldsymbol{u}_i\boldsymbol{y} + \boldsymbol{c}_i\boldsymbol{e}_i\boldsymbol{y}, \\ \boldsymbol{w}_i &= \boldsymbol{u}_i\boldsymbol{y} + \boldsymbol{v}_i\boldsymbol{x}, \\ \boldsymbol{w}_i + \boldsymbol{b}_i\boldsymbol{h}^{-1} &= \boldsymbol{u}_i\boldsymbol{y} + \boldsymbol{c}_i\boldsymbol{f}_i\boldsymbol{x}. \end{aligned}$$

The weights of those vectors are respectively bounded from above by the following values: $(r_u+r_e)r_y$, $r_ur_y+r_vr_x$, $r_ur_y+r_fr_x$; however, in practice, with high probability, those inequalities are equalities. Moreover, those 3 vectors have length k and their weights are bounded from above by values that are always strictly smaller than k for the given sets of parameters. Thus, using Proposition 8, we can consider that, with high probability:

$$\|\boldsymbol{a}_i\boldsymbol{h}\| = (r_u + r_e)r_y, \quad \|\boldsymbol{w}_i\| = r_ur_y + r_vr_x, \quad \|\boldsymbol{w}_i + \boldsymbol{b}_i\boldsymbol{h}^{-1}\| = r_ur_y + r_fr_x.$$

Thus, one can easily compute the intersection of the supports of those vectors and using Lemma 3, one gets the following bound:

$$\dim\left(\text{Support}(\boldsymbol{a}_i\boldsymbol{h}) \cap \text{Support}(\boldsymbol{w}_i) \cap \text{Support}(\boldsymbol{w}_i + \boldsymbol{b}_i\boldsymbol{h}^{-1})\right) \geq 2(r_x(r_e + 2r_u) - m). \tag{3.51}$$

Note that, for the sake of clarity, we considered, like the authors of [LT20b], that $r_x = r_y$, $r_v = r_e$, and $r_f = r_u$. One can get an equality from Equation (3.51) by replacing m by

$$\dim(\text{Support}(\boldsymbol{a}_i\boldsymbol{h}) + \text{Support}(\boldsymbol{w}_i) + \text{Support}(\boldsymbol{w}_i + \boldsymbol{b}_i\boldsymbol{h}^{-1}))$$

which is exactly m with high probability.

So far, one has computed an intersection of 3 vector spaces, denoted Z in the following, whose dimension is $2(r_x(r_e + 2r_u) - m)$ and which contains $\text{Support}(\boldsymbol{u}_i\boldsymbol{y})$ of dimension $r_u r_y = r_u r_x$.

Finding $\boldsymbol{u}_i\boldsymbol{y}_i$. Then, we use the knowledge of Z to recover the ephemeral key $\boldsymbol{u}_i\boldsymbol{y}_i$ using the following procedure:

- Sample a random subspace T of Z of dimension $r_u r_x$.
- Let $A = \text{Support}(\boldsymbol{a}_i\boldsymbol{h})$. Find a vector space T' such that $A = T + T'$ where T and T' are in direct sum.
- Write $\boldsymbol{a}_i\boldsymbol{h}$ as $\boldsymbol{t} + \boldsymbol{t}'$ where $\boldsymbol{t} \in T^k$ and $\boldsymbol{t}' \in T'^k$. $\boldsymbol{t}$ is a candidate for $\boldsymbol{u}_i\boldsymbol{y}_i$.
- If $\|\boldsymbol{t}\boldsymbol{h}^{-1}\| = \|\boldsymbol{t}\|$, then with overwhelming probability $\boldsymbol{t} = \boldsymbol{u}_i\boldsymbol{y}_i$. Otherwise, start over by sampling an other vector space T.

Finishing the attack. Once we recover $\boldsymbol{u}_i\boldsymbol{y}_i$ for $1 \leq i \leq l$, we can recover the other ephemeral keys by computing:

- $\boldsymbol{u}_i\boldsymbol{x} = \boldsymbol{h}^{-1}\boldsymbol{u}_i\boldsymbol{y}_i$
- $\boldsymbol{v}_i\boldsymbol{y} = \boldsymbol{h}(\boldsymbol{w}_i - \boldsymbol{u}_i\boldsymbol{y})$
- $\boldsymbol{e}_i\boldsymbol{x} = \boldsymbol{c}_i^{-1}(\boldsymbol{a}_i - \boldsymbol{u}_i\boldsymbol{x})$
- $\boldsymbol{f}_i\boldsymbol{y} = \boldsymbol{c}_i^{-1}(\boldsymbol{b}_i - \boldsymbol{v}_i\boldsymbol{y})$

Using these keys and a valid signature $\sigma = (\{\boldsymbol{c}_i, \boldsymbol{a}_i, \boldsymbol{b}_i, \boldsymbol{s}_i\}_{1\leq i\leq l})$ on a message μ, one can forge a valid signature $\sigma' = (\{\boldsymbol{c}'_i, \boldsymbol{a}'_i, \boldsymbol{b}'_i, \boldsymbol{s}'_i\}_{1\leq i\leq l})$ for a message μ' using the following procedure for $1 \leq i \leq l$:

- Compute $\boldsymbol{w}_i = (\boldsymbol{a}_i, \boldsymbol{b}_i)\boldsymbol{H}^\intercal - \boldsymbol{c}_i\boldsymbol{s}_i$

- Set $\boldsymbol{s}'_i = \boldsymbol{s}_i$ and $\boldsymbol{w}'_i = \boldsymbol{w}_i$
- Compute $\boldsymbol{c}'_i = \mathcal{H}(\{\boldsymbol{w}_i, \boldsymbol{s}_i\}, \mu', pk)$
- Set $\boldsymbol{a}'_i = \boldsymbol{u}_i\boldsymbol{x} + \boldsymbol{c}'_i\boldsymbol{e}_i\boldsymbol{x}$
- Set $\boldsymbol{b}'_i = \boldsymbol{v}_i\boldsymbol{y} + \boldsymbol{c}'_i\boldsymbol{f}_i\boldsymbol{y}$

Finally, we derive the complexity of this attack:

Theorem 8. *The information leakage attack forges a valid signature using:*

$$l \times m \times k \times \min(m,k) \times \prod_{j=0}^{r_u r_x - 1} q^z(q^{r_u r_x} - q^j)$$

operations over $\mathbb{F}_q$, where z is dimension of the recovered vector space Z.

Proof. Using Proposition 9, we know that the probability of finding the correct support of $\boldsymbol{u}_i\boldsymbol{y}$ is $\prod_{j=0}^{r_u r_x - 1} q^{-z}(q^{r_u r_x} - q^j)$.

The most costly step in verifying that the sampled support T is the correct one is checking the rank of $\boldsymbol{t}\boldsymbol{h}^{-1}$, which can be done using a Gaussian elimination in $m \times k \times \min(m,k)$ operations over $\mathbb{F}_q$.

This process needs to be repeated l times to forge a complete signature, hence the result. □

Table 3.6 gives the complexity of our *information leakage* attack on the RPS parameters given in [LT20b].

Parameter set	Claimed security	Success probability	Complexity
RPS-C1	128	2^{-327}	2^{347}
RPS-C2	192	2^{-27}	2^{47}

Table 3.6: Complexities of our *information leakage* attack on the RPS parameters given in [LT20b], and comparison with their claimed security.

3.5.3.2 Random low rank vectors attack

In this section, we present our second attack, namely the random low rank vectors attack.

In order to do so, we describe a procedure that can be used to forge valid signatures when the parameters m and k are too close to the weight of the vectors $\boldsymbol{a}_i$ and $\boldsymbol{b}_i$.

On input (pk, μ), our goal is to find a signature σ such that Verify(σ, μ, pk) accepts it. For each $1 \leq i \leq l$, we start by computing $\boldsymbol{s}_i$, $\boldsymbol{w}_i$ and $\boldsymbol{c}_i$ as follows:

- Sample $\boldsymbol{s}_i$ as a random vector of weight $r_e r_y + r_f r_x$
- Sample $\boldsymbol{w}_i$ as a random vector of weight $r_u r_y + r_v r_x$
- Let $\boldsymbol{c}_i = \mathcal{H}(\{\boldsymbol{w}_i, \boldsymbol{s}_i\}, \mu, pk)$

Since $\boldsymbol{s}_i$ is chosen randomly, the following conditions are fulfilled:

- $\|\boldsymbol{s}_i \boldsymbol{h}\| \geq \min(m-1, k)$
- $\|\boldsymbol{s}_i \boldsymbol{h}^{-1}\| \geq \min(m-1, k)$

Now we need to find $\boldsymbol{a}_i$ and $\boldsymbol{b}_i$ such that:

$$\begin{aligned}(\boldsymbol{a}_i, \boldsymbol{b}_i)\boldsymbol{H}^\intercal - \boldsymbol{c}_i \boldsymbol{s}_i = \boldsymbol{w}_i &\iff (\boldsymbol{a}_i, \boldsymbol{b}_i)\boldsymbol{H}^\intercal = \boldsymbol{w}_i + \boldsymbol{c}_i \boldsymbol{s}_i \\ &\iff \boldsymbol{a}_i \boldsymbol{h} + \boldsymbol{b}_i \boldsymbol{h}^{-1} = \boldsymbol{w}_i + \boldsymbol{c}_i \boldsymbol{s}_i.\end{aligned}$$

Let $\boldsymbol{z} = \boldsymbol{w}_i + \boldsymbol{c}_i \boldsymbol{s}_i$. We start by splitting $\boldsymbol{z}$ into a sum of two vectors $\boldsymbol{z}_1$ and $\boldsymbol{z}_2$ such that $\|\boldsymbol{z}_1\| = (r_u + r_e) r_y$ and $\|\boldsymbol{z}_2\| = (r_v + r_f) r_x$.

In order to do so, one needs to write the vector $\boldsymbol{z}$ as matrix $\boldsymbol{Z}$ in $\mathbb{F}_q^{m \times k}$ using a basis of $\mathbb{F}_{q^m}$ seen as an $\mathbb{F}_q$-vector space. Then, one picks 2 matrices $\boldsymbol{Z}_1$ and $\boldsymbol{Z}_2$, of rank respectively $(r_u + r_e) r_y$ and $(r_v + r_f) r_x$, at random in $\mathbb{F}_q^{m \times k}$ and solves the following linear system:

$$\boldsymbol{V} \times (\boldsymbol{Z}_1 | \boldsymbol{Z}_2)^\intercal \ = \ \boldsymbol{Z}$$

where $\boldsymbol{V}$ is a $m \times 2m$ matrix containing unknowns.

Since $\|\boldsymbol{z}_1\|$ and $\|\boldsymbol{z}_2\|$ have the same order of magnitude as k, this system has far more unknowns than equations, thus, it has a lot of solutions.

Those solutions are of the form $\boldsymbol{V}_0 + \boldsymbol{K}$ where $\boldsymbol{K}$ is a matrix whose rows belong to the left kernel of $(\boldsymbol{Z}_1|\boldsymbol{Z}_2)^\intercal$. Each one of these solutions yields different matrices $\boldsymbol{Z}'_1$ and $\boldsymbol{Z}'_2$ which can easily be turned into vectors $\boldsymbol{z}_1$ and $\boldsymbol{z}_2$ in $\mathbb{F}_{q^m}^k$.

The two vectors $\boldsymbol{z}_1$ and $\boldsymbol{z}_2$ will have correct weights (respectively $(r_u + r_e)r_y$ and $(r_v + r_f)r_x$) with a probability bounded from below by $0.28^2 \approx 8\%$. This is due to the fact that one is looking for a solution matrix $\boldsymbol{V}_0 + \boldsymbol{K}$ whose two first blocks of size $m \times m$ are non-singular. This happens with probability asymptotically close to 1 when q grows but bounded from below by 0.288, see for instance Lemma 7 in [BGL03].

In order to verify this assumption in an RPS context, we computed 10.000 solutions for a system with parameters $q = 2, m = 100, k = 90, r_1 = r_2 = 80$, and it appeared that 8.1% of the associated $\boldsymbol{z}_1$ and $\boldsymbol{z}_2$ had the expected weights.

The cost of computing this decomposition is basically the one of performing linear algebra (solving a system, computing rank, multiplying two matrices, ...) on matrices of size $m^2 \times k$, $m \times m$ and $m \times k$.

Most importantly, the number of different decompositions of the form $\boldsymbol{z} = \boldsymbol{z}_1 + \boldsymbol{z}_2$ (with correct weights) that can be found is huge, roughly

$$\begin{bmatrix} m \\ (r_u + r_e)r_y \end{bmatrix}_q \times \begin{bmatrix} m \\ (r_v + r_f)r_x \end{bmatrix}_q \times 0.28^2 \times 2^{m(2m-k)}. \tag{3.52}$$

The two first terms concern the choice of the two supports of $\boldsymbol{z}_1$ and $\boldsymbol{z}_2$, the third one the aforementioned probability and the last one the number of possible choices for the matrix $\boldsymbol{K}$. This is only a very rough estimation since it does not consider the probability that the two vector spaces have a non-trivial intersection nor the probability that the rank of $(\boldsymbol{Z}_1|\boldsymbol{Z}_2)^\intercal$ is maximal.

However, this rough estimation enables one to grasp the order of magnitude of the number of distinct solutions.

Then, we compute $\boldsymbol{a}_i = \boldsymbol{z}_1\boldsymbol{h}^{-1}$ and $\boldsymbol{b}_i = \boldsymbol{z}_2\boldsymbol{h}$. By construction, the following conditions are verified:

- $(\boldsymbol{a}_i, \boldsymbol{b}_i)\boldsymbol{H}^\intercal - \boldsymbol{c}_i\boldsymbol{s}_i = \boldsymbol{w}_i$
- $\|\boldsymbol{a}_i\boldsymbol{h}\| = (r_u + r_e)r_y$
- $\|\boldsymbol{b}_i\boldsymbol{h}\| = (r_v + r_f)r_x$

Applying Proposition 8 with $X = \mathbb{F}_{q^m}$, and $\boldsymbol{a}_i \in \mathbb{F}_{q^m}^k$, one gets the following result:

Proposition 10. *Assuming $\boldsymbol{a}_i$ and $\boldsymbol{b}_i$ behave like independent vectors chosen uniformly at random, the probability that this procedure outputs $\boldsymbol{a}_i$ and $\boldsymbol{b}_i$ such that*

$$\|\boldsymbol{a}_i\| = (r_u + r_e)r_y, \text{ and } \|\boldsymbol{b}_i\| = (r_v + r_f)r_x$$

is:

$$q^{-i_1(max(m,k)-\min(m,k)+i_1)} \times q^{-i_2(max(m,k)-\min(m,k)+i_2)}$$

where $i_1 = \min(m,k) - (r_u + r_e)r_x$, and $i_2 = \min(m,k) - (r_v + r_f)r_y$.

Note that the assumption in Proposition 10 above is supported by experiments.

Finally, we derive the complexity of this attack:

Theorem 9. *The random low rank vectors attack forges a valid signature using:*

$$l \times 2(m \times k \times min(m,k)) \times \frac{1}{p}$$

operations over $\mathbb{F}_q$, where p is the probability of success given in Proposition 10.

Proof. Checking the rank of $\boldsymbol{a}_i$ and $\boldsymbol{b}_i$ is done by performing two Gaussian eliminations on $m \times k$ matrices over $\mathbb{F}_q$, and each Gaussian elimination costs $m \times k \times min(m,k)$ operations over $\mathbb{F}_q$.

This process needs to be repeated $\frac{1}{p}$ times on average in order to find a valid pair of vectors.

Finally, this process needs to be repeated for l times, hence the result. □

Table 3.7 gives the complexity of our *random low rank vectors* attack on the RPS parameters given in [LT20b].

Parameter set	Claimed security	Success probability	Attack complexity
RPS-C1	128	2^{-48}	2^{68}
RPS-C2	192	2^{-96}	2^{117}

Table 3.7: Complexities of our *random low rank vectors* attack on the RPS parameters given in [LT20b], and comparison with their claimed security.

3.5.3.3 Quantum speedup

In [GHT16], the authors described how solving the rank decoding problem using a quantum computer gives a quadratic speed-up when considering combinatorial algorithms.

In this section, we use similar techniques to show that we obtain a quadratic speedup for both attacks described in Sections 3.5.3.1 and 3.5.3.2.

Theorem 10 ([GHT16], Theorem 1.)**.** *Let f be a Boolean function $f : \{0,1\}^b \to \{0,1\}$ that is computable by a NAND circuit of size S. Let p be the proportion of roots of the Boolean function:*

$$p \stackrel{def}{=} \frac{\#\{x \in \{0,1\}^b : f(x) = 0\}}{2^b}.$$

Then there is a quantum algorithm based on iterating a quantum circuit $\mathcal{O}\left(\frac{1}{\sqrt{p}}\right)$ many times that outputs with probability at least $\frac{1}{2}$ one of the roots of the Boolean function. The size of this circuit is $\mathcal{O}(S)$.

Information leakage attack. For this attack, we want to speed up the process of finding the correct vector space $\text{Support}(\boldsymbol{u}_i\boldsymbol{y})$ from the vector space leaked from the signatures. From Proposition 9, we have:

$$p = \prod_{j=0}^{x-1} q^{-y}(q^x - q^j).$$

As explained in Section 3.5.3.1, checking whether the vector space is the right one can be done by solving a linear system with $r_u r_y$ unknowns, performing a multiplication in $\mathbb{F}_{q^m}[X]/P$, and checking the rank of the resulting vector, which is the most costly operation. Hence, there exists a NAND classical circuit that performs this verification using $\mathcal{O}\left(max(m,k)^3\right)$ gates.

Random low rank vectors attack. For this attack, we want to speed up the search of vectors $\boldsymbol{a}_i = \boldsymbol{z}_1\boldsymbol{h}^{-1}$ and $\boldsymbol{b}_i = \boldsymbol{z}_2\boldsymbol{h}$ with correct weights. Proposition 10 gives the value for p:

$$p = q^{-i_1(\max(m,k)-\min(m,k)+i_1)} \times q^{-i_2(\max(m,k)-\min(m,k)+i_2)}$$

where $i_1 = \min(m,k) - (r_u + r_e)r_x$ and $i_2 = \min(m,k) - (r_v + r_f)r_y$.

As for the previous attack, checking whether the two resulting vectors have the desired weight can be performed by a NAND classical circuit using $\mathcal{O}\left(max(m,k)^3\right)$ gates.

Resulting complexities. These results show that, for both of our attacks, the search for, respectively, the correct vector space and vectors with right weight, can be performed in $\mathcal{O}\left(\frac{1}{\sqrt{p}}\right)$ iterations of the circuit when using a quantum computer. The cost of evaluating this circuit remains unchanged.

Parameter set	Claimed security	Quantum attack complexity
RPS-P1	128	2^{94}
RPS-P2	192	2^{170}

Table 3.8: Complexities of our attacks with a quantum speedup on the RPS parameters given in [LT20b], and comparison with their claimed security.

This yields the complexities given in Table 3.8 where they are compared with the RPS parameters targeting quantum security.

Note that the attack against RPS-P1 uses the *random low rank vectors* attack, whereas the attack against RPS-P2 uses the *information leakage* attack.

3.5.4 Conclusion

In this section, we have described attacks against the RPS signature scheme which break all sets of parameters proposed in [LT20b].

More precisely, our attacks enable us to forge valid signatures in 2^{68} and 2^{47} operations for sets of parameters whose claimed securities are, respectively, 128 and 192 bits.

In addition to this, we give a quantum adaptation of our attack; this yields an attack on the last two sets of parameters given in [LT20b] which target quantum security.

Overall, our attacks highlight weaknesses of the RPS scheme and give new constraints when designing new parameter sets.

Last but not least, we performed simulations for both of our attacks in order to support our theoretical claims. More details about these experimental results together with the source code are given in [ABG21].

Chapter 4

Improvements on Rank-based cryptosystems

Contents

4.1 Rank Quasi-Cyclic (RQC) Encryption scheme

Rank Quasi-Cyclic (RQC) is a code-based encryption scheme using the rank metric, it has been introduced in [AMBD+18].

The main advantage of RQC is that its security relies on random ideal codes, thus it does not require any masking process like in classical McEliece-like cryptosystems such as ROLLO.

In this section, we recall its classical version, before introducing three improvements: Augmented Gabidulin codes, the multi-syndromes, and the non-homogeneous error approaches. Finally, using all this improvements together, we propose two new encryption schemes, namely Multi-RQC-AG and Multi-UR-AG in Sections 4.2.4 and 4.2.5.

4.1.1 Classic RQC

On Figure 4.1, we briefly recall the classic RQC scheme as described in [AAB+20].

In order to do so, we need to introduce the following notation:

$$\begin{aligned}\mathcal{S}_w^n(\mathbb{F}_{q^m}) &= \{\boldsymbol{x} \in \mathbb{F}_{q^m}^n : \|\boldsymbol{x}\| = w\},\\ \mathcal{S}_{w,1}^n(\mathbb{F}_{q^m}) &= \{\boldsymbol{x} \in \mathbb{F}_{q^m}^n : \|\boldsymbol{x}\| = w, 1 \in \mathrm{Support}(\boldsymbol{x})\}.\end{aligned}$$

Moreover, as indicated by its name, RQC relies on ideal codes, see Definition 12 and Section 2.1.3, which are a generalization of quasi-cyclic codes.

In this section, when we mention the name of a coding theory problem with an upper case "I" in it, it stands for its version relying on ideal codes; see the list of acronyms on page 16.

For instance, **IRD** stands for the Ideal Rank Decoding problem.

4.1.1.1 Correctness.

Let us have a look at the correctness of the scheme, that is to say, check if the decryption process works. For the sake of clarity, the "mod P" operators are omitted below, but all the computations are done modulo P.

$$\begin{aligned}\boldsymbol{v} - \boldsymbol{y}\cdot\boldsymbol{u} &= \boldsymbol{m}\boldsymbol{G} + \boldsymbol{s}\cdot\boldsymbol{r}_2 + \boldsymbol{e} - \boldsymbol{y}\cdot(\boldsymbol{r}_1 + \boldsymbol{h}\cdot\boldsymbol{r}_2)\\ &= \boldsymbol{m}\boldsymbol{G} + (\boldsymbol{x} + \boldsymbol{h}\cdot\boldsymbol{y})\cdot\boldsymbol{r}_2 + \boldsymbol{e} - \boldsymbol{y}\cdot(\boldsymbol{r}_1 + \boldsymbol{h}\cdot\boldsymbol{r}_2)\\ &= \boldsymbol{m}\boldsymbol{G} + \boldsymbol{x}\cdot\boldsymbol{r}_2 - \boldsymbol{y}\cdot\boldsymbol{r}_1 + \boldsymbol{e}\end{aligned}$$

$\underline{\mathsf{Setup}(1^\lambda)}$: Generates and outputs $\mathsf{param} = (n, k, \delta, w, w_1, P)$ where $P \in \mathbb{F}_q[X]$ is an irreducible polynomial of degree n.

$\underline{\mathsf{KeyGen}(\mathsf{param})}$: Samples $\boldsymbol{h} \xleftarrow{\$} \mathbb{F}_{q^m}^n$, $\boldsymbol{g} \xleftarrow{\$} \mathcal{S}_n^n(\mathbb{F}_{q^m})$, and $(\boldsymbol{x}, \boldsymbol{y}) \xleftarrow{\$} \mathcal{S}_{w,1}^{2n}(\mathbb{F}_{q^m})$. Computes the generator matrix $\boldsymbol{G} \in \mathbb{F}_{q^m}^{k \times n}$ of a code $\mathcal{C}$ which corrects up to δ errors, sets $\mathsf{pk} = (\boldsymbol{g}, \boldsymbol{h}, \boldsymbol{s} = \boldsymbol{x} + \boldsymbol{h} \cdot \boldsymbol{y} \mod P)$, and $\mathsf{sk} = (\boldsymbol{x}, \boldsymbol{y})$, returns $(\mathsf{pk}, \mathsf{sk})$.

$\underline{\mathsf{Encrypt}(\mathsf{pk}, \boldsymbol{m}, \theta)}$: Uses randomness θ to generate $(\boldsymbol{r}_1, \boldsymbol{e}, \boldsymbol{r}_2) \xleftarrow{\$} \mathcal{S}_{w_1}^{3n}(\mathbb{F}_{q^m})$, sets $\boldsymbol{u} = \boldsymbol{r}_1 + \boldsymbol{h} \cdot \boldsymbol{r}_2 \mod P$ and $\boldsymbol{v} = \boldsymbol{m}\boldsymbol{G} + \boldsymbol{s} \cdot \boldsymbol{r}_2 + \boldsymbol{e} \mod P$, returns $\boldsymbol{c} = (\boldsymbol{u}, \boldsymbol{v})$.

$\underline{\mathsf{Decrypt}(\mathsf{sk}, \mathbf{c})}$: Returns $\mathcal{C}.\mathsf{Decode}(\boldsymbol{v} - \mathbf{u} \cdot \mathbf{y} \mod P)$.

Figure 4.1: Description of the RQC encryption scheme.

Since

$$\boldsymbol{v} - \boldsymbol{y} \cdot \boldsymbol{u} = \boldsymbol{m}\boldsymbol{G} + (\boldsymbol{x} \cdot \boldsymbol{r}_2 - \boldsymbol{y} \cdot \boldsymbol{r}_1 + \boldsymbol{e}) \in \mathbb{F}_{q^m}^n,$$

it means that, as long as

$$\|\boldsymbol{x} \cdot \boldsymbol{r}_2 - \boldsymbol{y} \cdot \boldsymbol{r}_1 + \boldsymbol{e}\| \leq \delta,$$

one recovers $\boldsymbol{m}$ by decoding $\boldsymbol{v} - \boldsymbol{y} \cdot \boldsymbol{u}$; in other words:

$$\mathcal{C}.\mathsf{Decode}(\boldsymbol{v} - \mathbf{u} \cdot \mathbf{y} \mod P) = \boldsymbol{m}.$$

4.1.1.2 Security.

Theorem 11. *The RQC scheme depicted in Figure 4.1 is* **IND-CPA** *under the* **DIRD** *assumption.*

Proof. For the proof of Theorem 11, the reader may refer to [AMBD+18, AAB+20]. □

Despite the fact that the aforementioned proof is not given in this document, we would like to recall the two instances an adversary could attack:

$$\begin{pmatrix} \mathbf{I}_n & \mathcal{IM}(\boldsymbol{h}) \end{pmatrix} \times \begin{pmatrix} \boldsymbol{x}^\intercal \\ \boldsymbol{y}^\intercal \end{pmatrix} = \begin{pmatrix} \boldsymbol{s}^\intercal \end{pmatrix}, \tag{4.1}$$

$$\begin{pmatrix} \mathbf{I}_n & \mathbf{0} & \mathcal{IM}(\boldsymbol{h}) \\ \mathbf{0} & \mathbf{I}_n & \mathcal{IM}(\boldsymbol{s}) \end{pmatrix} \times \begin{pmatrix} \boldsymbol{r}_1^\intercal \\ \boldsymbol{e}^\intercal \\ \boldsymbol{r}_2^\intercal \end{pmatrix} = \begin{pmatrix} \boldsymbol{u}^\intercal \\ (\boldsymbol{v} - \boldsymbol{m}\boldsymbol{G})^\intercal \end{pmatrix}. \tag{4.2}$$

On the one hand, the decoding instance given by Equation (4.1) corresponds to attacking a code with parameters $[m, 2n, n, w]$, this is a key recovery attack.

On the other hand, the decoding instance given by Equation (4.2) corresponds to attacking a code with parameters $[m, 3n, n, w_1]$, this is an attack on the security of the message.

4.2 Our Multi-RQC-AG and Multi-UR-AG encryption schemes

4.2.1 Augmented Gabidulin codes

In this section, we introduce a new family of efficiently decodable codes, namely Augmented Gabidulin codes.

The main idea behind these codes is to add a sequence of zeros at the end of a Gabidulin codes, see Definition 15; by doing this, one directly gets elements of the support of the error, which correspond to *support erasure* in a rank metric context. Decoding these codes corresponds to decoding classical Gabidulin codes to which support erasures are added.

In practice, this approach permits to decrease m, i.e. the degree of the extension field $\mathbb{F}_{q^m}$, at the cost of having a probabilistic decoding.

This approach is particularly suitable when many errors have to be corrected, which is exactly the case in code-based cryptography where the code to be decoded has a very low rate.

In what follows, we give a definition of augmented Gabidulin codes, and for didactic purpose, in Proposition 12, we recall a simple and natural way to decode Gabidulin code with support erasures. For others or more efficient approaches, the reader may refer to [ALR18, CB22, GP08].

Remark 21. *Notice that this type of approach (adding zeros) is not relevant in Hamming metric since the errors are independent in a classical noisy channel, whereas in rank metric, errors located on different coordinates are linked since they share the same support.*

Definition 41 (Augmented Gabidulin codes)**.** *Let $(k, n, n', m) \in \mathbb{N}^4$ such that $k \leq n' \leq m < n$. Let $\boldsymbol{g} = (g_1, \ldots, g_{n'})$ be an $\mathbb{F}_q$-linearly independent family of n' elements of $\mathbb{F}_{q^m}$, and let $\overline{\boldsymbol{g}}$ be the vector of length n which is equal to $\boldsymbol{g}$ padded with $n - n'$ extra zeros on the right.*

The Augmented Gabidulin code $\mathcal{G}^+_{\overline{\boldsymbol{g}}}(n, n', k, m)$ *is the* $[n, k]_{q^m}$*-code defined by*

$$\mathcal{G}^+_{\overline{\boldsymbol{g}}}(n, n', k, m) := \left\{ P(\overline{\boldsymbol{g}}),\ \deg_q(P) < k \right\},$$

where $P(\overline{\boldsymbol{g}}) := (P(g_1), \ldots, P(g_{n'}), 0, \ldots, 0)$.

A generator matrix for $\mathcal{G}_{\mathbf{g}}$ is given by:

$$\boldsymbol{G} = \begin{pmatrix} g_1 & \ldots & g_{n'} & 0 & \ldots & 0 \\ g_1^q & \ldots & g_{n'}^q & 0 & \ldots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ g_1^{q^{k-1}} & \ldots & g_{n'}^{q^{k-1}} & 0 & \ldots & 0 \end{pmatrix} \in \mathbb{F}_{q^m}^{k \times n}.$$

Proposition 11 (Decoding capacity of Augmented Gabidulin codes)**.**
Let $\mathcal{G}^+_{\overline{\boldsymbol{g}}}(n, n', k, m)$ *be an augmented Gabidulin code, and let*

$$\varepsilon \in \{1, 2, \ldots, \min(n - n', n' - k)\}$$

be the dimension of the vector space generated by the support erasures.
Then, $\mathcal{G}^+_{\overline{\boldsymbol{g}}}(n, n', k, m)$ *can uniquely decode an error of rank weight up to*

$$t := \left\lfloor \frac{n' - k + \varepsilon}{2} \right\rfloor.$$

Proof. The minimal distance of $\mathcal{G}^+_{\overline{\boldsymbol{g}}}(n, n', k, m)$ is clearly $d = n' - k + 1$ since it is made of a Gabidulin code augmented with zeros.

Let $x = c_1 + e_1$ be a noisy codeword where $c_1 \in \mathcal{G}^+_{\overline{\boldsymbol{g}}}$, and $\|e_1\| \leq t$.

Let us assume that x is not uniquely decodable to find a contradiction.

If x is not uniquely decodable, it means that there exists c_2 in $\mathcal{G}^+_{\overline{\boldsymbol{g}}}$ such that $c_2 \neq c_1$ and $x = c_2 + e_2$ where $\|e_2\| \leq t$.

Recall that we assume that one knows support erasures which span a vector space of dimension ε. These support erasures come from the $n - n'$ last coordinates of the code $\mathcal{G}^+_{\overline{\boldsymbol{g}}}$, thus these support elements are common to e_1 and e_2. Since Support(e_1) and Support(e_2) share ε elements, one has that

$$\mathrm{d}(e_1, e_2) \leq 2(t - \varepsilon) + \varepsilon = 2t - \varepsilon \leq n' - k.$$

Since $x = c_1 + e_1 = c_2 + e_2$, one clearly has that $\mathrm{d}(c_1, c_2) = \mathrm{d}(e_1, e_2)$, thus $\mathrm{d}(c_1, c_2) \leq n' - k$, which is a contradiction.

Thus, $\mathcal{G}^+_{\overline{\boldsymbol{g}}}$ can uniquely decode errors of rank weight up to $t := \left\lfloor \frac{n'-k+\varepsilon}{2} \right\rfloor$.

Finally, the condition $1 \leq \varepsilon \leq \min(n - n', n' - k)$ comes from the fact that the dimension of the vector space spanned by support erasures cannot exceed the maximum rank weight of the error nor the number of zero coordinates of the augmented Gabidulin code; in other words, on one hand ε is clearly smaller than $n - n'$, and on the other hand

$$\begin{aligned} \varepsilon \leq \left\lfloor \frac{n' - k + \varepsilon}{2} \right\rfloor &\implies 2\varepsilon \leq n' - k + \varepsilon \\ &\implies \varepsilon \leq n' - k. \end{aligned}$$

□

Proposition 12 (Decoding Algorithm for Augmented Gabidulin codes)**.**
Let $\mathcal{G}^{+}_{\boldsymbol{g}}(n, n', k, m)$ be an augmented Gabidulin code, and let

$$\varepsilon \in \{1, 2, \ldots, \min(n - n', n' - k)\}$$

be the dimension of the vector space generated by the support erasures.

This code benefits from an efficient decoding algorithm correcting errors of rank weight up to $\delta := \left\lfloor \frac{n'-k+\varepsilon}{2} \right\rfloor$ with a decoding failure rate (DFR) of

$$1 - q^{\delta(n'-n)} \sum_{i=\varepsilon}^{\delta} \prod_{j=0}^{\varepsilon-1} \frac{(q^{\delta} - q^{j})(q^{n-n'} - q^{j})}{q^{i} - q^{j}}. \tag{4.3}$$

Proof. The proof gives the decoding algorithm. One is given a noisy encoded word $\overline{\boldsymbol{y}} = \overline{\boldsymbol{c}} + \overline{\boldsymbol{e}} \in \mathbb{F}_{q^m}^{n}$ where $\overline{\boldsymbol{c}} := \boldsymbol{x}\boldsymbol{G}$ belongs to $\mathcal{G}^{+}_{\boldsymbol{g}}(n, n', k, m)$ and $\|\boldsymbol{e}\| \leq \delta$.

Step 1: recovering a part of the error support. By construction we have $\overline{\boldsymbol{c}} = (*|0 \ldots 0)$, so that the last $n-n'$ coordinates of $\boldsymbol{y}$ are exactly the last coefficients of $\overline{\boldsymbol{e}}$. Thus, one may use these coefficients to recover ε elements in $E := \mathrm{Support}(\overline{\boldsymbol{e}})$. This will be doable as long as these $n-n'$ coefficients contain at least ε linearly independent ones. The converse probability is the probability that a random $\delta \times (n - n')$ matrix with coefficients in $\mathbb{F}_q$ has rank less than ε. This yields the probability given by Equation (4.3).

Step 2: recovering $\overline{\mathbf{c}}$. Assume now that ε elements in the support of $\overline{\boldsymbol{e}}$ are known and let E_2 be the vector space spanned by these elements. In what follows, we focus on the first n' coordinates of $\overline{\boldsymbol{y}}, \overline{\boldsymbol{c}}$, and $\overline{\boldsymbol{e}}$ which are denoted by $\boldsymbol{y}, \boldsymbol{c}$ and $\boldsymbol{e}$ respectively.

By definition of $\mathcal{G}^{+}_{\boldsymbol{g}}(n, n', k, m)$, there exists a q-polynomial P of q-degree at most $k-1$ such that for $1 \leq i \leq n'$:

$$y_i = P(g_i) + e_i. \tag{4.4}$$

Let also V and V_2 be the unique monic q-polynomials of q-degree δ and ε which vanish on the vector spaces E and E_2 respectively. The ring of q-polynomials being left Euclidean, there exists a unique monic q-polynomial W of degree $\delta - \varepsilon$ such that $V = W \circ V_2$. As E_2 is known, one can easily build the q-polynomial V_2, for instance using the iterative process described in [Ore33, Loi07]. Evaluating V at both sides of Equation (4.4), one gets $V(y_i) = (V \circ P)(g_i) + V(e_i) = V \circ P(g_i)$. This secret polynomial can be written symbolically using $\delta - \varepsilon$ unknowns in $\mathbb{F}_{q^m}$, and similarly we view $R := V \circ P$ as a q-polynomial of q-degree $k-1+\delta$ with unknown coefficients. Thus, we can derive a linear equation containing $k + 2\delta - \varepsilon$ unknowns in $\mathbb{F}_{q^m}$ from

$$V(y_i) = R(g_i), \tag{4.5}$$

and the same goes for any $i \in \{1, 2, \dots n'\}$. Overall, this gives a linear system with n' equations in $k + 2\delta - \varepsilon$ variables. This linear system has more equations than unknowns as long as $\delta \leq \left\lfloor \frac{n'-k+\varepsilon}{2} \right\rfloor$, which is the case by assumption. Moreover, this system has a unique solution by Proposition 11. This means that exactly $k + 2\delta - \varepsilon$ equations are linearly independent, thus one can solve the system to recover V and R, so one finally gets P. □

4.2.2 Multiple syndromes approach

The multiple syndrome approach has recently been used in [AAD+22] for LRPC-based cryptosystems.

Roughly, it relies on the use of multiple **RD** instances to significantly decrease the value of n, i.e. the length of the code.

In order to understand the core of the multiple syndromes approach, let us use figures.

On Figure 4.2, one is able to notice that the length of the code $\mathcal{C}$ used in RQC, i.e. n, sets the lengths of all the other vectors of the scheme: $\boldsymbol{x}$, $\boldsymbol{y}$, $\boldsymbol{s}$, etc.

Since this length depends on the rank weight of the error one needs to correct, it cannot be decreased, however, one could decrease the length of the secret and public key.

More precisely, Figure 4.3 shows that, using 3 times a shorter public key $\boldsymbol{s}$ of length $n_2 = \frac{n}{3}$ still enables one to encrypt a message.

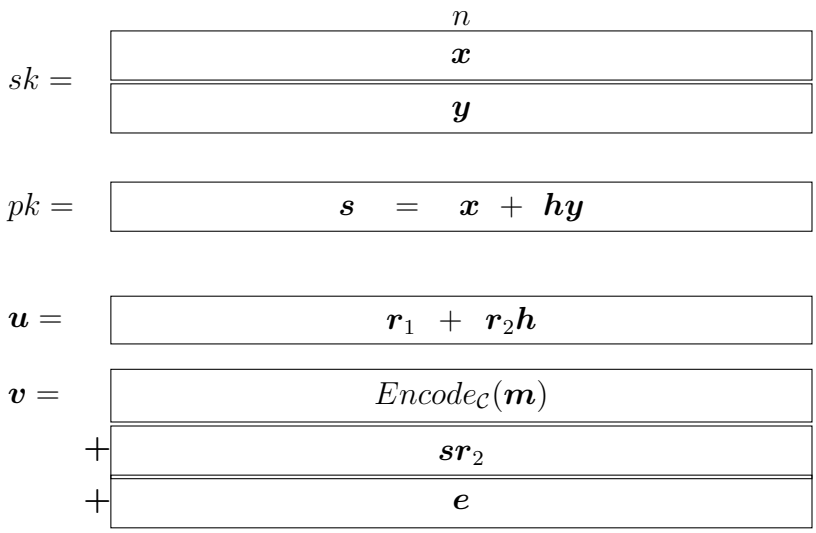


Figure 4.2: Classic RQC.

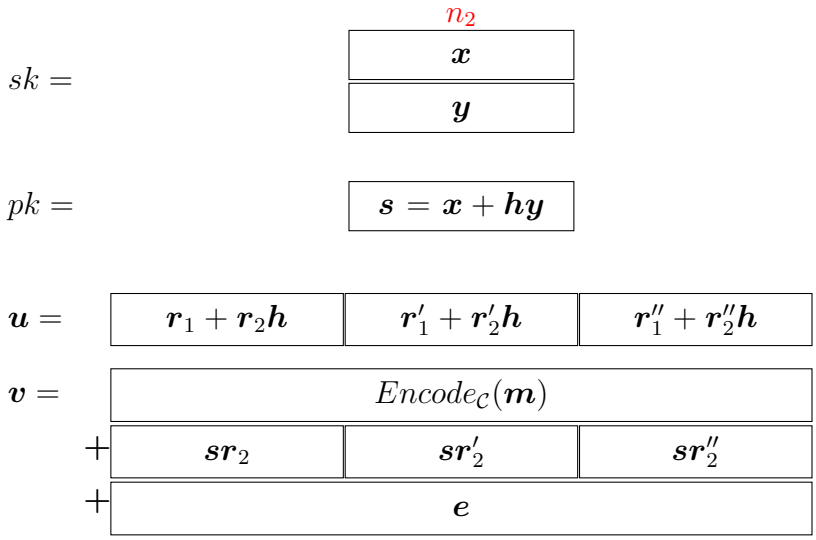


Figure 4.3: Multiple syndromes approach.

The decryption process is straightforward; still referring to Figure 4.3: in order to recover $\boldsymbol{m}$, one simply has to decode

$$\boldsymbol{v} - \Big(\boldsymbol{u}_1 \cdot \boldsymbol{y} \mid \boldsymbol{u}_2 \cdot \boldsymbol{y} \mid \boldsymbol{u}_3 \cdot \boldsymbol{y}\Big)$$

where the $\boldsymbol{u}_i$'s are the three parts, each of length n_2, of $\boldsymbol{u}$.

Overall, this approach does not change the size of the ciphertext, but it enables one to significantly decrease the size of the public key.

Most importantly, the vectors $\boldsymbol{r}_1, \boldsymbol{r}_2, \boldsymbol{r}'_1, \boldsymbol{r}'_2, \boldsymbol{r}''_1, \boldsymbol{r}''_2$ have to share the *same support*, otherwise the rank weight of the error to be decoded would be far too high.

Naturally, this approach, described as an example on Figure 4.3, is readily generalized to any multiple of n_2, as we will see in Sections 4.2.4 and 4.2.5.

Last but not least, this approach enables one to switch the regime of the error to an area closer to the rank Gilbert-Varshamov bound. More precisely, in RQC, the rank weight of the error to be decoded is in $\mathcal{O}(\sqrt{n})$ where n is the length of the code.

For instance, with the multiple syndromes approach, the public key security of the scheme relies on a decoding instance with parameters $[m, 2n_2, n_2]$ instead of $[m, 2n, n]$, without modifying the rank weight of the error; this mechanically leads to an **RD** instance with a target rank closer to the Gilbert-Varshamov bound since $n_2 < n$.

This is of interest in a cryptographic context since this area is the one where the decoding instances are the hardest to solve.

4.2.3 Non-homogeneous error approach

Recall that the security of the RQC encryption scheme relies on two decoding instances with parameters $[m, 2n, n, w]$ and $[m, 3n, n, w_1]$, see Equations (4.1) and (4.2).

For a given rank weight, it is far easier to attack a $[3n, n]_{\mathbb{F}_{q^m}}$-code than a $[2n, n]_{\mathbb{F}_{q^m}}$-code, see Section 3.2.

Thus, one has to take $w_1 \gg w$ in order to reach a certain level of security. This has a strong impact on the parameters of RQC since it increases the total rank weight of the error to be decoded, i.e. ww_1.

Our idea to mitigate the impact of the $[3n, n]_{\mathbb{F}_{q^m}}$ attack is to increase only a part of the error. More precisely, we increase only the rank weight of a third of the error vector, namely the vector $\boldsymbol{e}$.

Formally, the error vector $(\boldsymbol{r}_1, \boldsymbol{e}, \boldsymbol{r}_2) \in \mathbb{F}_{q^m}^{3n}$ will be picked such that

$$\|(\boldsymbol{r}_1, \boldsymbol{r}_2)\| = w_1, \|\boldsymbol{e}\| = w_1 + w_2, \quad \text{Support}(\boldsymbol{r}_1, \boldsymbol{r}_2) \subset \text{Support}(\boldsymbol{e}).$$

Doing so, the decoding instance given by Equation (4.2) is now a decoding instance with a *non-homogeneous* error; and the total rank weight to be decoded in RQC becomes $ww_1 + w_2$.

We introduced this new approach in the Second Round (of NIST PQC Standardization Process) update of RQC [AAB+20].

4.2.4 Our Multi-RQC-AG encryption scheme

4.2.4.1 Notation and procedures

Before introducing Multi-RQC-AG, let us define a few notation and procedures. Let $\mathcal{S}^{2n}_{w,1}(\mathbb{F}_{q^m})$ and $\mathcal{S}^{n_2\times 3n_1}_{(w_1,w_2)}(\mathbb{F}_{q^m})$ be defined as:

$$\begin{aligned}\mathcal{S}^{2n}_{w,1}(\mathbb{F}_{q^m}) &= \{\boldsymbol{x} = (\boldsymbol{x}_1, \boldsymbol{x}_2) \in \mathbb{F}^{2n}_{q^m} \;\big|\; \|\boldsymbol{x}\| = w,\ 1 \in \mathrm{Support}(\boldsymbol{x})\},\\ \mathcal{S}^{n_2\times 3n_1}_{(w_1,w_2)}(\mathbb{F}_{q^m}) &= \{\mathbf{X} = (\mathbf{X}_1, \mathbf{X}_2, \mathbf{X}_3) \in \mathbb{F}^{n_2\times 3n_1}_{q^m} \;\big|\; \|(\mathbf{X}_1, \mathbf{X}_3)\| = w_1,\\ &\qquad \|\mathbf{X}_2\| = w_1 + w_2,\ \mathrm{Support}(\mathbf{X}_1, \mathbf{X}_3) \subset \mathrm{Support}(\mathbf{X}_2)\}.\end{aligned}$$

Let n_1, n_2 be positive integers such that $n = n_1 \times n_2$, for a vector $\mathbf{v} \in \mathbb{F}^{n_2}_{q^m}$ and a matrix $\mathbf{M} \in \mathbb{F}^{n_2\times n_1}_{q^m}$ whose columns are labeled $\mathbf{M}_1, \ldots, \mathbf{M}_{n_1}$, we extend the dot product (defined for ideal codes in Section 2.1.3) such that:

$$\mathbf{v} \cdot \mathbf{M} = \left((\mathbf{v}\cdot\mathbf{M}_1^\intercal \mod P)^\intercal, \ldots, (\mathbf{v}\cdot\mathbf{M}_{n_1}^\intercal \mod P)^\intercal\right) \in \mathbb{F}^{n_2\times n_1}_{q^m},$$

Let $\mathbf{v} = (\mathbf{v}_1, \ldots, \mathbf{v}_{n_1}) \in \mathbb{F}^{n}_{q^m}$ with $\mathbf{v}_i \in \mathbb{F}^{n_2}_{q^m}\ \forall i \in \{1, \ldots, n_1\}$, the Fold () procedure turns the vector $\boldsymbol{v}$ into a $n_2 \times n_1$ matrix

$$\mathrm{Fold}\,(\mathbf{v}) = (\mathbf{v}_1^\intercal, \ldots, \mathbf{v}_{n_1}^\intercal) \in \mathbb{F}^{n_2\times n_1}_{q^m}.$$

The procedure Unfold () is naturally defined as the converse of Fold ().

4.2.4.2 Multi-RQC-AG

By using all the aforementioned improvements together, we were able to design a new scheme, namely Multi-RQC-AG, which stands for Multiple Syndromes RQC with Augmented Gabidulin-code. This scheme is described on Figure 4.4.

It can be instantiated with or without the use of non-homogeneous error, this is why we sometimes call it Multi-RQC-AG-NH; however, in general and when there is no ambiguity, we just call it Multi-RQC-AG.

Multi-RQC-AG relies on two codes, an augmented Gabidulin code $\mathcal{G}^{+}_{\boldsymbol{g}}(n, n', k, m)$ that can correct up to $\delta := \left\lfloor \frac{n'-k+\varepsilon}{2} \right\rfloor$ errors using the efficient decoding algorithm

$\underline{\mathsf{Setup}(1^\lambda)}$

Generate and output the parameters

$$\mathsf{param} = (n', n_1, n_2, k, \epsilon, \delta, w, w_1, w_2, P)$$

where $P \in \mathbb{F}_q[X]$ is an irreducible polynomial of degree n_2.

$\underline{\mathsf{KeyGen}(\mathsf{param})}$:

Sample $\mathbf{g} \xleftarrow{\$} \mathcal{S}_{n'}^{n'}(\mathbb{F}_{q^m})$, $\mathbf{h} \xleftarrow{\$} \mathbb{F}_{q^m}^{n_2}$ and $(\mathbf{x}, \mathbf{y}) \xleftarrow{\$} \mathcal{S}_{w,1}^{2n_2}(\mathbb{F}_{q^m})$

Compute $\mathbf{s} = \mathbf{x} + \mathbf{h} \cdot \mathbf{y} \mod P$

Output $\mathsf{pk} = (\mathbf{g}, \mathbf{h}, \mathbf{s})$ and $\mathsf{sk} = (\mathbf{x}, \mathbf{y})$

$\underline{\mathsf{Encrypt}(\mathsf{pk}, \mathbf{m}, \theta)}$:

Compute $\overline{\mathbf{g}} = (\mathbf{g} \,|\, 0 \dots 0) \in \mathbb{F}_{q^m}^{n_1 n_2}$

Compute the generator matrix $\mathbf{G} \in \mathbb{F}_{q^m}^{k \times (n_1 n_2)}$ of $\mathcal{G}_{\overline{\boldsymbol{g}}}^{+}(n_1 n_2, n', k, m)$

Sample $(\mathbf{R}_1, \mathbf{E}, \mathbf{R}_2) \xleftarrow{\$} \mathcal{S}_{w_1, w_2}^{n_2 \times 3n_1}(\mathbb{F}_{q^m})$ using randomness θ

Compute $\mathbf{U} = \mathbf{R}_1 + \mathbf{h} \cdot \mathbf{R}_2$ and $\mathbf{V} = \mathrm{Fold}\,(\mathbf{mG}) + \mathbf{s} \cdot \mathbf{R}_2 + \mathbf{E}$

Output $\mathbf{C} = (\mathbf{U}, \mathbf{V})$

$\underline{\mathsf{Decrypt}(\mathsf{pk}, \mathsf{sk}, \mathbf{C})}$:

Output $\mathbf{m} = \mathcal{G}_{\overline{\boldsymbol{g}}}^{+}.\mathsf{Decode}\,(\mathrm{Unfold}\,(\mathbf{V} - \mathbf{y} \cdot \mathbf{U}))$

Figure 4.4: Multi-RQC-AG encryption scheme

$\mathcal{G}_{\overline{\boldsymbol{g}}}^{+}.\mathsf{Decode}(.)$ as well as a random ideal $[2n_2, n_2]_{\mathbb{F}_{q^m}}$-code with parity check matrix $(\mathbf{I} \;\; \mathcal{IM}(\mathbf{h}))$. The correctness of the protocol follows from:

$$\begin{aligned}\mathbf{V} - \mathbf{y} \cdot \mathbf{U} &= \mathrm{Fold}\,(\mathbf{mG}) + (\mathbf{x} + \mathbf{h} \cdot \mathbf{y}) \cdot \mathbf{R}_2 + \mathbf{E} - \mathbf{y} \cdot (\mathbf{R}_1 + \mathbf{h} \cdot \mathbf{R}_2) \\ &= \mathrm{Fold}\,(\mathbf{mG}) + \mathbf{x} \cdot \mathbf{R}_2 - \mathbf{y} \cdot \mathbf{R}_1 + \mathbf{E}.\end{aligned}$$

As a consequence,

$$\mathrm{Unfold}\,(\mathbf{V} - \mathbf{y} \cdot \mathbf{U}) = \mathbf{mG} + \mathrm{Unfold}\,(\mathbf{x} \cdot \mathbf{R}_2 - \mathbf{y} \cdot \mathbf{R}_1 + \mathbf{E}) \in \mathbb{F}_{q^m}^{n}$$

which means that $\mathcal{G}_{\boldsymbol{g}}.\mathsf{Decode}\,(\mathrm{Unfold}\,(\mathbf{V} - \mathbf{y} \cdot \mathbf{U})) = \mathbf{m}$ as long as:

$$\| \mathrm{Unfold}\,(\mathbf{x} \cdot \mathbf{R}_2 - \mathbf{y} \cdot \mathbf{R}_1 + \mathbf{E}) \| \leq \delta.$$

4.2.5 Our Multi-UR-AG encryption scheme

4.2.5.1 Notation and procedures

To define Multi-UR-AG, we use the same procedures defined in the previous section, namely Fold () and Unfold (). In addition to this, we just need the following notation:

$$\begin{aligned}\mathcal{S}_{w,1}^{n\times 2n_1}(\mathbb{F}_{q^m}) &= \{\mathbf{X}=(\mathbf{X}_1,\mathbf{X}_2)\in\mathbb{F}_{q^m}^{n\times 2n_1} \;\big|\; \|\mathbf{X}\|=w,\ 1\in\mathrm{Support}(\mathbf{X})\},\\ \mathcal{S}_{(w_1,w_2)}^{n_2\times(n+n_1+n)}(\mathbb{F}_{q^m}) &= \{\mathbf{X}=(\mathbf{X}_1,\mathbf{X}_2,\mathbf{X}_3)\in\mathbb{F}_{q^m}^{n_2\times(n+n_1+n)} \;\big|\; \|(\mathbf{X}_1,\mathbf{X}_3)\|=w_1,\\ &\qquad \|\mathbf{X}_2\|=w_1+w_2,\quad \mathrm{Support}(\mathbf{X}_1,\mathbf{X}_3)\subset\mathrm{Support}(\mathbf{X}_2)\}.\end{aligned}$$

4.2.5.2 Multi-UR-AG

For more security, one could want to rely on unstructured decoding instances, that is to say, without ideal structure.

With the classical RQC encryption scheme, removing the ideal structure would drastically increase the size the public key and the ciphertext.

However, with all the improvements that we introduced, it is now possible to remove the ideal structure and yet still have competitive sizes, see Table 4.2.

We call this new scheme Multi-UR-AG(-NH), which stands for Multiple Syndromes Unstructured Rank with Augmented Gabidulin-code, still with or without the use of non-homogeneous error. Multi-UR-AG is described on Figure 4.5.

It relies on two codes: an augmented Gabidulin code $\mathcal{G}_{\boldsymbol{g}}^{+}(n,n',k,m)$ that can can correct up to $\delta := \left\lfloor \frac{n'-k+\varepsilon}{2} \right\rfloor$ errors using the efficient decoding algorithm $\mathcal{G}_{\boldsymbol{g}}^{+}.\mathsf{Decode}(.)$ as well as a random $[2n,n]_{\mathbb{F}_{q^m}}$-code with parity check matrix $(\mathbf{I}\ \mathbf{H})$. The correctness of the protocol follows from:

$$\begin{aligned}\mathbf{V}-\mathbf{UY} &= \mathrm{Fold}\,(\mathbf{mG})+\mathbf{R}_2(\mathbf{X}+\mathbf{HY})+\mathbf{E}-(\mathbf{R}_1+\mathbf{R}_2\mathbf{H})\mathbf{Y}\\ &= \mathrm{Fold}\,(\mathbf{mG})+\mathbf{R}_2\mathbf{X}-\mathbf{R}_1\mathbf{Y}+\mathbf{E}.\end{aligned}$$

As a consequence,

$$\mathrm{Unfold}\,(\mathbf{V}-\mathbf{YU}) = \mathbf{mG}+\mathrm{Unfold}\,(\mathbf{XR}_2-\mathbf{YR}_1+\mathbf{E})\in\mathbb{F}_{q^m}^{n},$$

which means that $\mathcal{G}_{\boldsymbol{g}}.\mathsf{Decode}\,(\mathrm{Unfold}\,(\mathbf{V}-\mathbf{YU}))=\mathbf{m}$ as long as

$$\|\ \mathrm{Unfold}\,(\mathbf{XR}_2-\mathbf{YR}_1+\mathbf{E})\ \| \le \delta.$$

$\underline{\mathsf{Setup}(1^\lambda)}$

Generate and output

$$\mathsf{param} = (n, n', n_1, n_2, k, \epsilon, \delta, w, w_1, w_2).$$

$\underline{\mathsf{KeyGen}(\mathsf{param})}$:

Sample $\mathbf{g} \xleftarrow{\$} \mathcal{S}_{n'}^{n'}(\mathbb{F}_{q^m})$, $\mathbf{H} \xleftarrow{\$} \mathbb{F}_{q^m}^{n \times n}$ and $(\mathbf{X}, \mathbf{Y}) \xleftarrow{\$} \mathcal{S}_{w,1}^{n \times 2n_1}(\mathbb{F}_{q^m})$

Compute $\mathbf{S} = \mathbf{X} + \mathbf{H}\mathbf{Y}$

Output $\mathsf{pk} = (\mathbf{g}, \mathbf{H}, \mathbf{S})$ and $\mathsf{sk} = (\mathbf{X}, \mathbf{Y})$

$\underline{\mathsf{Encrypt}(\mathsf{pk}, \mathbf{m}, \theta)}$:

Compute $\overline{\mathbf{g}} = (\mathbf{g} \,|\, 0 \dots 0) \in \mathbb{F}_{q^m}^{n_1 n_2}$

Compute the generator matrix $\mathbf{G} \in \mathbb{F}_{q^m}^{k \times (n_1 n_2)}$ of $\mathcal{G}_{\overline{\boldsymbol{g}}}^{+}(n_1 n_2, n', k, m)$

Sample $(\mathbf{R}_1, \mathbf{E}, \mathbf{R}_2) \xleftarrow{\$} \mathcal{S}_{w_1,w_2}^{n_2 \times (n+n_1+n)}(\mathbb{F}_{q^m})$ using randomness θ

Compute $\mathbf{U} = \mathbf{R}_1 + \mathbf{R}_2\mathbf{H}$ and $\mathbf{V} = \mathrm{Fold}\,(\mathbf{m}\mathbf{G}) + \mathbf{R}_2\mathbf{S} + \mathbf{E}$

Output $\mathbf{C} = (\mathbf{U}, \mathbf{V})$

$\underline{\mathsf{Decrypt}(\mathsf{pk}, \mathsf{sk}, \mathbf{C})}$:

Output $\mathbf{m} = \mathcal{G}_{\overline{\boldsymbol{g}}}^{+}.\mathsf{Decode}\,(\mathrm{Unfold}\,(\mathbf{V} - \mathbf{U}\mathbf{Y}))$

Figure 4.5: Multi-UR-AG encryption scheme

4.2.6 Conclusion and Security

Let us sum up the 3 improvements described in Sections 4.2.1 to 4.2.3:

- The Augmented Gabidulin codes enables one to decrease the value of m, i.e. the degree of the extension field $\mathbb{F}_{q^m}$.
- The Multiple Syndromes approach enables one to decrease the value of n, i.e. the length of the random code. Moreover, this improvement leads to **RD** instances for which the target rank is closer to the rank Gilbert-Varshamov bound compared to classical RQC.
- The non-homogeneous error approach enables one to mitigate the impact of attacks against the $[3n, n]_{\mathbb{F}_{q^m}}$-code, thus one can take $w = w_1$ (and no longer

$w_1 \gg w$) with an extra rank weight w_2 which will not have a strong impact on the total weight to be decoded.

Since everything comes at a price, each of the aforementioned improvements has consequences. More specifically, the three following sections, namely Sections 4.2.6.1 to 4.2.6.3, give details about the implications of each improvements.

4.2.6.1 Augmented Gabidulin codes.

The use of Gabidulin codes introduces a decryption failure rate whereas the decoding algorithm for classical RQC is deterministic.

However, the probability of decoding failure can be easily controlled by sacrificing only a few support erasures or by slightly increasing the number of extra zeros; indeed, the decoding failure probability decreases exponentially fast, with a quadratic exponent, see Equation (4.3).

4.2.6.2 Multiple Syndromes.

The use of multiple syndromes changes the problem an adversary can attack. Let us come back to the toy example from Figures 4.2 and 4.3; in the case of classical RQC, recall that the adversary can attack the following decoding instance (in order to attack the ciphertext):

$$\begin{pmatrix} \mathbf{I}_n & \mathcal{IM}(\boldsymbol{h}) \end{pmatrix} \times \begin{pmatrix} \boldsymbol{r}_1^\intercal \\ \boldsymbol{r}_2^\intercal \end{pmatrix} = \begin{pmatrix} \boldsymbol{u}^\intercal \end{pmatrix},$$

When using multiple syndromes like on Figure 4.3, an adversary could attack the following decoding instances:

$$\begin{pmatrix} \mathbf{I}_n & \mathcal{IM}(\boldsymbol{h}) \end{pmatrix} \times \begin{pmatrix} \boldsymbol{r}_1^\intercal & \boldsymbol{r}_1'^\intercal & \boldsymbol{r}_1''^\intercal \\ \boldsymbol{r}_2^\intercal & \boldsymbol{r}_2'^\intercal & \boldsymbol{r}_2''^\intercal \end{pmatrix} = \begin{pmatrix} \boldsymbol{u}_1^\intercal & \boldsymbol{u}_2^\intercal & \boldsymbol{u}_3^\intercal \end{pmatrix}, \tag{4.6}$$

It is clear that Equation (4.6) corresponds to an **RSL** instance instead of an **RD** instance in the case of classic RQC.

This is, in a sense, the drawback of using multiple syndromes: one does not rely on the classical decoding problem but on its generalization, namely the Rank Support Learning (**RSL**) problem, see Definition 18.

4.2.6.3 Non-Homogeneous Error.

Recall than in classic RQC, the adversary can attack the following $[3n, n]_{\mathbb{F}_{q^m}}$-code:

$$\begin{pmatrix} \mathbf{I}_n & \mathbf{0} & \mathcal{IM}(\boldsymbol{h}) \\ \mathbf{0} & \mathbf{I}_n & \mathcal{IM}(\boldsymbol{s}) \end{pmatrix} \times \begin{pmatrix} \boldsymbol{r}_1^\intercal \\ \boldsymbol{e}^\intercal \\ \boldsymbol{r}_2^\intercal \end{pmatrix} = \begin{pmatrix} \boldsymbol{u}^\intercal \\ (\boldsymbol{v} - \boldsymbol{m}\boldsymbol{G})^\intercal \end{pmatrix}. \tag{4.7}$$

If one uses non-homogeneous error, the instance given by Equation (4.7) becomes an **NHRD** instance.

Similarly to what happened with the multiple syndrome approach, the use of non-homogeneous error changes the problem which appears in the security proof.

4.2.6.4 Security.

The security of NH-Multi-RQC-AG relies on the Decisional Ideal Rank Decoding problem (**DIRD**), and on the Decisional Ideal Non-Homogeneous Rank Support Learning problem (**DNHIRSL**), see Theorem 12.

Recall that, so far, there is no known attack to solve the decisional versions of these problems without solving the associated search instances.

In addition to this, there is currently no attack that takes advantage of the ideal structure; thus, studying the security of NH-Multi-RQC-AG comes down to evaluating the complexity of **RD** and **NHRSL**.

Unlike NH-Multi-RQC-AG, our new scheme NH-Multi-UR-AG does not use ideal structure. Despite the aforementioned absence of attack that exploits ideal structure, it might induce a weakness in a scheme.

This is why NH-Multi-UR-AG, which does not use any structure like its name suggests it, is more secure.

To study its security, according to Theorem 13, one has to evaluate the complexity of **RSL** and **NHRSL**.

However, for an even better security, one could use Multi-UR-AG, i.e. with homogeneous weight, making its security relying solely on **RSL**, see for instance the parameters sets Multi-UR-AG-128 and Multi-UR-AG-192 in Section 4.2.7.

Theorem 12. *The NH-Multi-RQC-AG scheme depicted in Figure 4.4 is* **IND-CPA** *under the* **DIRD** *and the* **DNHIRSL** *assumptions.*

Proof. The proof of Theorem 12 is a straightforward adaptation of the proof given in [AAB+20]. One simply has to use the following instances: an $\mathbf{IRD}(m, 2n_2, n_2, w)$ instance defined from a $[2n_2, n_2]$ code given by Equation (4.8), and an

$$\mathbf{NHIRSL}(m, n_2, n_2, w_1, w_2, n_1)$$

instance defined from a $[3n_2, n_2]$ code given by Equation (4.9).

$$\begin{pmatrix} \mathbf{I}_{n_2} & \mathcal{IM}(\mathbf{h}) \end{pmatrix} \times (\mathbf{x}, \mathbf{y})^\intercal = \mathbf{s}^\intercal, \tag{4.8}$$

$$\begin{pmatrix} \mathbf{I}_{n_2} & \mathbf{0} & \mathcal{IM}(\mathbf{h}) \\ \mathbf{0} & \mathbf{I}_{n_2} & \mathcal{IM}(\mathbf{s}) \end{pmatrix} \times (\mathbf{R}_1, \mathbf{E}, \mathbf{R}_2)^\intercal = (\mathbf{U}, \mathbf{V} - \mathrm{Fold}(\mathbf{mG}))\,. \tag{4.9}$$

□

Theorem 13. *The NH-Multi-UR-AG scheme depicted in Figure 4.5 is* **IND-CPA** *under the* **DRSL** *and the* **DNHRSL** *assumptions.*

Proof. Similarly to the previous proof, the proof of Theorem 13 is a straightforward adaptation of the proof given in [AAB+20]. One simply has to use the following instances: an $\mathbf{RSL}(m, 2n, n, w, n_1)$ instance defined from a $[2n, n]$ code given by Equation (4.10), and an $\mathbf{NHRSL}(m, n, n_1, w_1, w_2, n_2)$ instance defined from a $[2n + n_1, n]$ code given by Equation (4.11).

$$\begin{pmatrix} \mathbf{I}_n & \mathbf{H} \end{pmatrix} \times (\mathbf{X}, \mathbf{Y})^\intercal = \mathbf{S}, \tag{4.10}$$

$$(\mathbf{R}_1, \mathbf{E}, \mathbf{R}_2) \times \begin{pmatrix} \mathbf{I}_n & \mathbf{0} & \mathbf{H} \\ \mathbf{0} & \mathbf{I}_{n_1} & \mathbf{S} \end{pmatrix}^\intercal = (\mathbf{U}, \mathbf{V} - \mathrm{Fold}(\mathbf{mG}))\,. \tag{4.11}$$

□

4.2.7 Parameters and comparisons

In this section, we propose parameters for our new encryption schemes, see Table 4.1; all parameters are chosen to resist to attacks described in Chapter 3.

Among the different codes that can be attacked for each of our schemes (see proofs of Theorems 12 and 13), there is not a weaker one which enables us to fix all of our parameters.

For instance, sometimes attacking the public key, *i.e.*, a code $[2n, n]$, gives the lowest complexity, but for another set of parameters, it will be the $[2n+n_1, n, w_1, w_2]$-code instead.

However, there seems to be an invariant: no matter the length n of the code or the dimension m of the extension, it looks like the closer to RGV bound the target rank weight r is, the better the combinatorial attacks are, and the worse are the algebraic attacks.

In other words, for a given $[m, n, k]$-code, there seems to always be a value of r such that all the combinatorial attacks will beat the algebraic ones. This seems to be the case both for homogeneous and non-homogeneous versions of the aforementioned problems, and with or without multiple syndromes.

Instance	Struct.	m	n'	n	n_1	n_2	k	ε	w	w_1	w_2	DFR	Sizes in **KB**		
													pk	**ct**	Total
Loong-128 [Wan19]	Random	191	182	35	13	14	6	0	8	11	0	0	10.9	16.0	**26.9**
Multi-RQC-AG-128	Ideal	83	82	-	5	38	2	74	7	11	0	-138	0.4	3.9	**4.4**
NH-Multi-RQC-AG-128	Ideal	61	60	-	3	50	3	51	7	7	5	-158	0.4	2.3	**2.7**
Multi-RQC-AG-192	Ideal	113	112	-	4	60	2	98	8	13	0	-215	0.9	6.8	**7.7**
NH-Multi-RQC-AG-192	Ideal	79	78	-	2	95	5	65	8	8	5	-238	0.9	3.8	**4.7**
Multi-UR-AG-128	Random	97	96	24	14	15	3	83	8	11	0	-190	4.1	6.9	**11.0**
NH-Multi-UR-AG-128	Random	73	72	22	13	14	2	66	8	8	4	-133	2.7	4.5	**7.1**
Multi-UR-AG-192	Random	127	126	35	15	16	3	93	9	12	0	-350	8.4	12.7	**21.1**
NH-Multi-UR-AG-192	Random	97	96	30	14	14	3	77	9	9	4	-214	5.1	7.5	**12.6**

Table 4.1: Parameters for our scheme with $q = 2$.

Similarly to [AAB+20], we use the fact that $1 \in \text{Support}(\boldsymbol{x}, \boldsymbol{y})$ to set $\delta := ww_1$ in the homogeneous case and $\delta := ww_1 + w_2$ in the non-homogeneous case. Recall that this quantity corresponds to the weight of the error decoded by the public Augmented Gabidulin code. For all our protocols (where "NH" denote non-homogeneous errors), both 128 and 192 bits security level are considered.

As a comparison, we also updated the parameters of the code-based KEM Loong [Wan19]. Note that this scheme does not use augmented Gabidulin codes nor non-homogeneous error but it does use multiple syndromes.

The parameter sets given in Table 4.1 come with the sizes of the associated public key **pk** and ciphertext **ct** expressed in kilo-bytes (KB).

Instance	128 bits	192 bits
NH-Multi-UR-AG	**7,122**	**12,602**
LRPC-MS [AAD+22]	7,205	14,270
Multi-UR-AG	**11,026**	**21,075**
FrodoKEM [ABD+21]	19,336	31,376
Loong-128 [Wan19]	26,948	-
Loidreau [Pha21]	36,300	-
Classic McEliece [BCL+17]	261,248	524,348

Instance	128 bits	192 bits
CRYSTALS-KYBER [BDK+18]	1,568	2,272
NH-Multi-RQC-AG	**2,710**	**4,732**
ILRPC-MS [AAD+22]	2,439	4,851
BIKE [AAB+21a]	3,113	6,197
Multi-RQC-AG	**4,378**	**7,668**
HQC [AAB+21b]	6,730	13,548

Table 4.2: Comparison of sizes for unstructured (random), and structured (ideal) KEMs. The sizes represent the sum of the public key and the ciphertext, expressed in bytes.

For Multi-RQC-AG:

$$|\mathbf{pk}| = 40 + \left\lceil \frac{n_2 m}{8} \right\rceil, \text{ and } \quad |\mathbf{ct}| = \left\lceil \frac{2n_1 n_2 m}{8} \right\rceil.$$

For Multi-UR-AG:

$$|\mathbf{pk}| = 40 + \left\lceil \frac{n n_1 m}{8} \right\rceil, \text{ and } \quad |\mathbf{ct}| = \left\lceil \frac{m(n n_2 + n_1 n_2)}{8} \right\rceil.$$

The term 40 represents the length of a seed used to generate $(\boldsymbol{g}, \boldsymbol{h})$, recall that the public key consists in $(\boldsymbol{g}, \boldsymbol{h}, \boldsymbol{s})$ and the ciphertext in the couple $(\boldsymbol{u}, \boldsymbol{v})$. Note that

the size of the secret key is not relevant since it is only a seed, thus it always has size 40 bytes.

Our most competitive parameter set in terms of sizes is NH-Multi-RQC-AG-128 which relies on non-homogeneous errors together with an ideal structure.

Otherwise, our best non-structured parameter set whose security solely depends on **RSL** is Multi-UR-AG-128.

Overall, Table 4.2 enables one to compare all our sizes to the ones of other KEMs based on ideal or non-structured matrices. For structured lattice-based schemes, we chose to focus on CRYSTALS-KYBER [BDK+18] which will be considered for standardization by NIST.

Last but not least, the vertical green line at $N = 150$ on Figure 3.1 shows the number of syndromes available for an adversary trying to attack a ciphertext of our scheme NH-Multi-RQC-AG-128. It is worth noticing that, even though the blue squares are below the black line (complexity of the plain **RD** attack), they are still way above the security level of 128 bits, and even given 150 syndromes, an attacker could not break our scheme. Note that it is far away from the area where the complexities of the different **RSL** attacks start to drop. More generally, we picked all our parameters that way, not only to resist to these attacks, but to be sure not to be targeted by any minor improvements.

Chapter 5

A new problem: the Square Space Factorization Problem

Contents

5.1 Presentation of the problem

Recall that in this document, $\beta = \{1, \alpha, \ldots, \alpha^{m-1}\}$ is a basis of $\mathbb{F}_{q^m}$ seen as an $\mathbb{F}_q$-vector space of dimension m.

Definition 42 (Product of Vector Spaces in $\mathbb{F}_{q^m}$)**.** *Let A, B be $\mathbb{F}_q$-vector spaces in $\mathbb{F}_{q^m}$. We define the product AB as the $\mathbb{F}_q$-vector space in $\mathbb{F}_{q^m}$ spanned by the following elements:*

$$\{ab \mid a \in A,\ b \in B\}.$$

Definition 43 (**SquareSpace** problem (search))**.** *The* **SquareSpace** *problem with parameters (q, m, r) is given by:*
Input : *an integer $r \in \mathbb{N}$, and a vector space $U \subset \mathbb{F}_{q^m}$ of dimension*

$$t := \frac{(r+1)r}{2}.$$

Output : *$E \subset \mathbb{F}_{q^m}$ such that $\dim(E) = r$, and $U = E^2$.*

Remark 22 (Parameters range)**.** *In this document, we will mainly focus on* **SquareSpace** *instances for which the parameters (q, m, r) fulfill the following conditions:*

$$q > 2, \quad r > 2, \quad m \text{ odd}, m > t^2.$$

There are several reasons for these conditions choice, one of the most important being that this range is suitable for cryptography, see Section 5.4 and the challenges in Section 5.3.

In what follows, before studying the attacks against the **SquareSpace** problem, we will discuss two points: if one picks a vector space E uniformly at random in $\mathbb{F}_{q^m}$, what will be the dimension of $U = E^2$? Then, will there be other solutions than E to the equation $U = E^2$?

5.1.1 Discussion about the dimension

Let m, r be integers such that

$$t := \frac{(r+1)r}{2} \ < \ m.$$

For any vector space E of dimension r in $\mathbb{F}_{q^m}$, the dimension of $U = E^2$ cannot be larger than t. In fact, there are t products of two elements belonging to basis of E, and U is by definition generated by these products, see Definition 42.

Moreover, if E is picked uniformly at random in $\mathbb{F}_{q^m}$, it looks like, with overwhelming probability, the dimension of U will be exactly t. This is the purpose of Proposition 13.

It is worth noticing that this latter result is similar to what happens with the dimension of square codes studied in [CCMZ15], even if these mathematical objects are different.

Proposition 13. *Let E be a vector space spanned by $r \geq 1$ elements picked uniformly at random in $\mathbb{F}_{q^m}$. The probability that the vector space $U = E^2$ has dimension smaller than $t := (r+1)r/2$ is bounded from above by*

$$\frac{4}{q^{m-1}} \sum_{i=0}^{r-1} q^{\frac{(i+2)(i+1)}{2}} \quad \leq \quad 4rq^{t-m+1}$$

Proof. Let f be an element in $\mathbb{F}_{q^m}$, let $\dim(E) = r$, we set $F := \langle E, f \rangle$, and we want to study the dimension of F^2.

It is readily seen that

$$\dim(F^2) \leq \dim(E^2) + r + 1,$$

moreover $\dim(F^2) < \dim(E^2) + r + 1$ if, and only if, there exists a linear combination of elements in F^2 which is zero.

This is equivalent to saying that $\exists b \in \{0,1\},\ e \in E, h \in E^2$ such that f is a root of $bX^2 + eX + h \in \mathbb{F}_{q^m}[X]$. Since this univariate polynomial has degree at most 2, for fixed b, e, h, if f is picked uniformly at random in $\mathbb{F}_{q^m}$, the probability that f is a root is smaller than $2/q^m$.

Considering all the possible values for b, e, h, we get that:

$$\mathbb{P}(\dim(F^2) < \dim(E^2) + r + 1) \leq \underbrace{2}_{\#\ \{0,1\}} \times \underbrace{q^r}_{\#\ E} \times \underbrace{q^t}_{\#\ E^2 \leq q^t} \times 2/q^m.$$

Using this upper bound, one can construct E^2 in an incremental way by adding elements e_i's picked uniformly at random in $\mathbb{F}_{q^m}$ one by one. More precisely one has

that:

$$\begin{aligned}
\mathbb{P}(\dim(\langle e_1\rangle^2) < 1 = 0+0+1) &\leq 4/q^m \\
\mathbb{P}(\dim(\langle e_1, e_2\rangle^2) < 3 = 1+1+1) &\leq 4q^2/q^m \\
\mathbb{P}(\dim(\langle e_1, e_2, e_3\rangle^2) < 6 = 2+3+1) &\leq 4q^5/q^m \\
&\vdots \\
\mathbb{P}(\dim(\langle e_1, e_2, \ldots, e_i\rangle^2) < i(i-1)/2 + (i-1) + 1) &\leq 4q^{\frac{(i+2)(i+1)}{2}-1}/q^m
\end{aligned}$$

The event "$\langle e_1, e_2, \ldots, e_i\rangle^2$ has dimension less than t" is the union of all the above events, thus one can bound the probability we are looking for from above, namely:

$$\mathbb{P}(\dim(\langle e_1, e_2, \ldots, e_r\rangle^2) < (r+1)r/2 = t) \leq \frac{4}{q^{m-1}} \sum_{i=0}^{r-1} q^{\frac{(i+2)(i+1)}{2}}.$$

□

The upper bound given by Proposition 13 is not very tight, especially for small parameters (q, m, r), however it is clear that it tends to zero when m grows, which proves the aforementioned assumption: with overwhelming probability, the dimension of $U = E^2$ will be t.

Another way to look at the dimension of E^2 when E is picked uniformly at random in $\mathbb{F}_{q^m}$, is to consider that the generating elements, *i.e.*, the products $e_i e_j$'s, behave like if they were "random".

More precisely, if these products were independent and identically distributed among vectors in $\mathbb{F}_{q^m}$, then the probability that U has rank t would be given by Equation (5.1), that is to say, the probability for an $m \times t$ matrix in $\mathbb{F}_q$ to be of full rank.

$$q^{-mt} \prod_{i=0}^{t-1} (q^m - q^i) \tag{5.1}$$

Experimentally, the percentage of vector spaces E such that $\dim(E^2) = t$ is very close to the probability given by Equation (5.1), see Table 5.1. One notices that the error decreases when m grows.

Parameters (q, m, r)	Equation (5.1)	Experimental ratio	Error
$(3, 4, 2)$	0.84540	0.92147	9.00%
$(3, 5, 2)$	0.94716	1	5.58%
$(3, 6, 2)$	0.98224	0.99198	0.99%
$(3, 7, 2)$	0.99406	1	0.60%
$(3, 9, 2)$	0.99934	1	0.07%
$(3, 7, 3)$	0.84038	0.89172	6.11%
$(3, 8, 3)$	0.94528	0.95335	0.85%
$(3, 9, 3)$	0.98159	0.98859	0.71%
$(3, 10, 3)$	0.99385	0.9522	0.14%
$(5, 7, 3)$	0.95042	0.96032	1.04%
$(5, 8, 3)$	0.99002	0.99023	0.02%
$(5, 9, 3)$	0.99800	0.99857	0.06%
$(5, 10, 3)$	0.99960	0.99961	0.00%
$(3, 11, 4)$	0.84019	0.84494	0.57%
$(3, 12, 4)$	0.94521	0.94598	0.08%
$(3, 13, 4)$	0.98157	0.98249	0.09%

Table 5.1: Experimental percentage of spaces E of dimension r s.t. $\dim(U) = t$ for different parameters, each time $100,000$ attempts were made; and comparison with the formula given by Equation (5.1).

Remark 23. *This is similar to what happens with the product of two vector spaces $A, B \subset \mathbb{F}_{q^m}$, a lower bound on the probability that $\dim(AB)$ is maximal is given in [AGH+19]; however, if one simply makes the same assumption as above, i.e., that every generating pair is independent from the others, then one finds a probability very close to the lower bound proved in [AGH+19].*

5.1.2 Discussion about the uniqueness

The uniqueness of the solution to a **SquareSpace** instance is a very important point in a cryptographic context, see Section 5.4; this is the topic of this section.

First, one notices that, if $m = t$, then any instance of the **SquareSpace** problem

with parameters (q, m, r) is trivial to solve.

In fact, for almost all spaces $E \subset \mathbb{F}_{q^m}$ of dimension r, E^2 will span the whole space, thus E will be a solution. This degenerate case shows that if m is too small, instances of the **SquareSpace** problem have too many solutions, and the problem is trivial.

When $m \gg t$, we think that there is a unique solution to a **SquareSpace** instance given by $(U = E^2, m, r)$ for a space E of dimension r in $\mathbb{F}_{q^m}$.

That is to say that, with overwhelming probability, there will not be any other space $\widetilde{E} \neq E$ of dimension r such that $U = \widetilde{E}^2$. When such a space $\widetilde{E}$ exists, one often speaks about *spurious* solution, or spurious key in a cryptographic context.

Similarly to the dimension of the square space discussed in Section 5.1.1, we do not have a proof of this result, but we do have experimental results.

Parameters (q, m, r)	N	Percentage of **SquareSpace** instances such that $\exists \widetilde{E} \neq E$ which is solution
$(3, 9, 3)$	10	70.0% (alg.)
$(3, 11, 3)$	1000	4.2%
$(3, 13, 3)$	1000	0.4%
$(3, 15, 3)$	1000	0.1%
$(3, 17, 3)$	10000	0.02%
$(3, 21, 3)$	10000	0%
$(5, 9, 3)$	10	50% (alg.)
$(5, 11, 3)$	100	1.0%
$(5, 13, 3)$	1000	0.1%
$(5, 15, 3)$	10000	0.01%

Table 5.2: For each set of parameters (q, m, r), N **SquareSpace** instances where generated, and the right column indicates the percentage of instances for which we found spurious solutions using the intersection or the algebraic attack "(alg.)".

One notices, on Table 5.2, that when m grows, the percentage of **SquareSpace** instances having a non-unique solution rapidly decreases down to zero.

Most values in Table 5.2 were computed using the intersection attack described in Section 5.2.1.2, however, when "(alg.)" is written by a percentage, it means that the algebraic attack, described in Section 5.2.2, was used instead.

5.2 Attacks and Complexities

Recall that all along this document,

$$t := \frac{(r+1)r}{2}.$$

In this section, let us consider a **SquareSpace** instance $(U = E^2, q, m, r)$ where $E \subset \mathbb{F}_{q^m}$ such that $\dim(E) = r$, and $\dim(U) = t$.

5.2.1 Combinatorial attacks

The plainest combinatorial attack is the bruteforce attack where one simply picks a vector space $E' \subset \mathbb{F}_{q^m}$ of dimension r and checks whether $E'^2 = U$; if yes, one returns E', otherwise she picks another space, and so on.

There are $\begin{bmatrix} m \\ r \end{bmatrix}_q$ vector spaces of dimension r in $\mathbb{F}_{q^m}$.

Thus, the complexity of the brute force attack, on average, in terms of the number of spaces one has to pick before finding a solution, is upper bounded by

$$\begin{bmatrix} m \\ r \end{bmatrix}_q.$$

In the two following sections, we propose two combinatorial attacks that are far more efficient than the bruteforce attack.

5.2.1.1 First combinatorial attack

Let $\{e_1, e_2, \ldots, e_r\} \subset \mathbb{F}_{q^m}$ be a basis of the secret space E.

A less naive combinatorial attack than brute force could take advantage of the fact that

$$\{e_1^2, e_2^2, \ldots, e_r^2\} \subset U. \tag{5.2}$$

Thus, by picking elements at random in U, one eventually finds an element of the form e_i^2, and having r such elements gives a candidate for E', see Algorithm 1.

There are

$$\begin{bmatrix} t \\ r \end{bmatrix}_q$$

Algorithm 1: A first combinatorial attack against **SquareSpace**

Input: A **SquareSpace** instance $(U = E^2, q, m, r)$ where $\dim(E) = r$, and $\dim(U) = t$.
Output: E' such that $\dim(E') = r$, and $U = E'^2$

```
found ← 0;
while found = 0 do
    ∀i ∈ {1..r}, u_i ←$ U;
    if IsSquare(u_1) and IsSquare(u_2) and ... and IsSquare(u_r) then
        ∀i ∈ {1..r}, x_i ← √u_i;
        E' ← ⟨x_1, x_2, ..., x_r⟩;
        if dim(E') = r and E'^2 = U then
            found ← 1;
        end
    end
end
return E';
```

vector spaces of dimension r in U, and at least one of them, namely $\langle e_1^2, e_2^2, \ldots, e_r^2 \rangle$ yields the secret space E and breaks the loop.

Thus, the success probability of finding a solution space E' is bounded from below by

$$\begin{bmatrix} t \\ r \end{bmatrix}_q^{-1}.$$

The complexity of the algorithm follows from the expectancy of a geometric distribution, in term of iterations of its **while** loop.

On a **SquareSpace** instance

$$(U = E^2, q, m, r)$$

where $\dim(E) = r$, and $\dim(U) = t$, Algorithm 1 finds a solution space E' with, on average, at most

$$\mathcal{O}\left(\begin{bmatrix} t \\ r \end{bmatrix}_q\right)$$

iterations of its **while** loop.

5.2.1.2 Intersection combinatorial attack

In this section, we give another combinatorial attack against the **SquareSpace** problem, namely the intersection attack. It is more efficient than the one described in Section 5.2.1.1.

Concerning this section, I would like to thank Jean-Pierre Tillich for very fruitful discussions.

The intersection attack is based on a simple fact given by Lemma 5.

Lemma 5. *Let $\{e_1, e_2, \ldots, e_r\} \subset \mathbb{F}_{q^m}$ be a basis of the secret space E, then for two integers i, j in $\{1..r\}$, one has that:*

$$E \subset \left(e_i^{-1}U\right) \cap \left(e_j^{-1}U\right). \tag{5.3}$$

Proof. Recall that by definition:

$$U := \langle e_i e_j \rangle_{1 \leq i,j \leq r}.$$

Then the proof is straightforward since for any $i \in \{1..r\}$:

$$\begin{aligned} e_i^{-1}U &= \langle \ldots, e_i^{-1}e_1e_i, e_i^{-1}e_2e_i, \ldots, e_i^{-1}e_{r-1}e_i, e_i^{-1}e_re_i, \ldots \rangle \\ &= \langle \ldots, e_1, e_2, \ldots, e_{r-1}, e_r, \ldots \rangle, \end{aligned}$$

thus $E \subset e_i^{-1}U$, for all $i \in \{1..r\}$. □

Remark 24. *Using the same notation as in the proof of Lemma 5, let us call A and B two complement vector spaces of E in $e_i^{-1}U$ and $e_j^{-1}U$, in other words, $E \oplus A = e_i^{-1}U$, and $E \oplus B = e_j^{-1}U$.*

In what follows, we assume that A and B are vector spaces uniformly distributed among the vector spaces of dimension $t - r$ in $\mathbb{F}_{q^m}$. This assumption is very light since it is similar to the one used in [AGH+19], and it is supported by experiments.

Lemma 6. *Let A, B be two vector spaces in $\mathbb{F}_{q^m}$, of dimension, respectively, d_1 and d_2, and such that $d_1 + d_2 \leq m$. If A, B are picked uniformly at random in $\mathbb{F}_{q^m}$, then the probability that*

$$A \cap B = \{0\}$$

is equal to:

$$\prod_{i=0}^{d_2-1} \frac{q^m - q^{d_1+i}}{q^m - q^i}. \tag{5.4}$$

Algorithm 2: The intersection attack against **SquareSpace**

Input: A **SquareSpace** instance $(U = E^2, q, m, r)$ where $\dim(E) = r$, and $\dim(U) = t$.
Output: E' such that $\dim(E) = r$, and $U = E'^2$

```
found ← 0;
while found = 0 do
    u_1 ←$ U;
    u_2 ←$ U;
    if IsSquare(u_1) and IsSquare(u_2) then
        E' ← (√u_1^{-1} U) ∩ (√u_2^{-1} U);
        if dim(E') = r and E'^2 = U then
            found ← 1;
        end
    end
end
return E';
```

Proof. Let $\boldsymbol{A} \in \mathbb{F}_q^{m \times d_1}$ (resp. $\boldsymbol{B} \in \mathbb{F}_q^{m \times d_2}$), be a basis matrix of A (resp. B) with respect to a basis β of $\mathbb{F}_{q^m}$ over $\mathbb{F}_q$. It is readily seen that the rank of the matrix $(\boldsymbol{A}|\boldsymbol{B})$ will be $d_1 + d_2$ if, and only if, if $A \cap B = \{0\}$.

Thus, the probability we are looking for is the probability for a matrix made of two full rank matrices (of size $m \times d_1$ and $m \times d_2$) to be of full rank. □

Using Lemma 5, Lemma 6, and using the same assumption as in Remark 24, one has that, as long as $m \gg t$, with overwhelming probability, the intersection in Algorithm 1 will be exactly E if u_1 and u_2 are *linearly independent* square elements of the form e_i^2.

There are $\begin{bmatrix} t \\ 2 \end{bmatrix}_q$ pairs of linearly independent elements in U, and we assume that $\begin{bmatrix} r \\ 2 \end{bmatrix}_q$ of them are squares of elements in E.

Thus, we expect the complexity of the intersection attack depicted in Algorithm 2

to be given by

$$\mathcal{O}\left(\begin{bmatrix} t \\ 2 \end{bmatrix}_q \times \begin{bmatrix} r \\ 2 \end{bmatrix}_q^{-1}\right) \tag{5.5}$$

in terms of iterations of the main loop of the algorithm.

Our experiments support the complexity formula given by Equation (5.5), see Table 5.3.

Remark 25. *If m is too close to t, Algorithm 2 can very well go on forever. In fact, in that case, the dimension of the intersection would never be r.*

A possible way to fix this issue is to look at subspaces of dimension r in the intersection; however we do not focus on that case since the parameters of interest are all such that $m \gg t$.

5.2.1.3 Experimental results and conclusion

Overall, the combinatorial attack given by Algorithm 1 is far more efficient than brute force; and the intersection combinatorial attack, given by Algorithm 2, is even more efficient, this is confirmed by our experiments depicted in Table 5.3.

It is worth noticing that the complexities of our combinatorial attacks (Algorithm 1 and Algorithm 2) are both independent from the value of m, as long as it is greater than t.

This is totally different from the brute force attack or the algebraic attacks described in Section 5.2.2.

5.2.2 Algebraic Attacks

When one is given a **SquareSpace** instance $U = E^2$ with parameters (q, m, r), the secret space E is unknown, so it is natural to consider elements in a basis of E to be the variables.

More precisely, let $\boldsymbol{E}$ be an $m \times r$ matrix of variables with an identity block on

Parameters (q, m, r)	Bruteforce	Algorithm 1	Algorithm 2
$[3, 11, 3]$	38.8	18.1	11.8
$[3, 13, 3]$	48.4	18.2	11.8
$[3, 15, 3]$	57.9	18.4	11.7
$[3, 17, 3]$	67.4	18.0	11.8
$[3, 21, 3]$	86.4	18.2	11.8
$[5, 11, 3]$	56.1	24.7^*	16.0
$[5, 13, 3]$	70.1	23.8^*	16.0
$[5, 15, 3]$	84.0	24.3^*	15.9
$[3, 21, 4]$	108.6	–	21.1^*
$[3, 23, 4]$	121.3	–	21.0^*

Table 5.3: Experimental complexities (in term of main loop iterations) of combinatorial attacks against the **SquareSpace** problem and comparison with brute force. All values are binary logarithms, and they are average values over 100 experiments (only 10 for the values with a star symbol "$*$"). The "$-$" symbol denotes experiments out of reach for a personal computer in a reasonable time.

its first $r \times r$ block:

$$\boldsymbol{E} := \left(\begin{array}{ccc} & \boldsymbol{I}_r & \\ \hline e_{1,r+1} & \cdots & e_{r,r+1} \\ e_{1,r+2} & \cdots & e_{r,r+2} \\ \vdots & \cdots & \vdots \\ e_{1,m} & \cdots & e_{r,m} \end{array}\right). \tag{5.6}$$

In order to match a basis matrix of E with the matrix $\boldsymbol{E}$, one simply requires the first maximal minor of a basis matrix E to be non-singular. Since E is picked at random, this happens with overwhelming probability if q is large, and it is lower bounded by 0.288, see for instance Lemma 7 in [BGL03]. If the following algebraic attack does not find any solution, then one simply picks another spot to put the identity matrix, and it eventually works since E has dimension r by definition.

This systematic form for $\boldsymbol{E}$ helps with the 3 following points:

- it enables one to reduce the number of variables from mr down to $(m-r)r$, which is an important point when dealing with algebraic attacks,
- it prevents one from getting singular solutions, for instance solution spaces with dimension smaller than r or even 0,
- last but not least, if E is the only solution, then our algebraic modelings will have a single solution as well whereas it would have a lot of solutions, namely $\#\,\mathrm{GL}_r(\mathbb{F}_q)$, without this specialization.

This last point is similar to what happens in Section 2.3.3 when one does not use Plücker coordinates.

5.2.2.1 The symbolic basis $\boldsymbol{U}_{\mathbf{symb}}$

Once one has built the matrix $\boldsymbol{E}$ containing variables, that is to say, the symbolic basis of the secret space E, she can derive a symbolic basis of U involving the very same variables; this is done by considering all the pairwise products of columns in $\boldsymbol{E}$ seen as elements in $\mathbb{F}_{q^m}$, see the toy example below.

We denote by $\boldsymbol{U}_{\mathrm{symb}}$ this $m \times t$ matrix containing *quadratic* polynomials in

$$\mathbb{F}_q[e_{r+1,1}, e_{r+2,1}, \ldots, e_{m-1,r}, e_{m,r}].$$

Here is a toy example corresponding to a **SquareSpace** instance with parameters $(q,m,r) = (3,4,2)$. Let $\{1, \alpha, \alpha^2, \alpha^3\}$ be an $\mathbb{F}_3$-basis of $\mathbb{F}_{3^4}$ where $\alpha^4 = \alpha^3 + 1$.

In that case, the matrix $\boldsymbol{E}$ would be

$$\boldsymbol{E} := \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ e_{1,3} & e_{2,3} \\ e_{1,4} & e_{2,4} \end{pmatrix} \in \mathbb{F}_3[e_{1,3}, e_{1,4}, e_{2,3}, e_{2,4}]^{4\times 2}.$$

In order to compute the entries of the first column of $\boldsymbol{U}_{\mathrm{symb}}$, one computes

$$\begin{aligned}(1 + e_{1,3}\alpha^2 + e_{1,4}\alpha^3)^2 \;=\; & (e_{1,3}^2 + 2e_{1,3}e_{1,4} + e_{1,4}^2 + 1) \;+\; (2e_{1,3}e_{1,4} + e_{1,4}^2)\alpha \\ & +\; (e_{1,4}^2 + 2e_{1,3})\alpha^2 \;+\; (e_{1,3}^2 + 2e_{1,3}e_{1,4} + e_{1,4}^2 + 2e_{1,4})\alpha^3\end{aligned}$$

where the 4 entries are in blue. More precisely, the first column of $\boldsymbol{U}_{\text{symb}}$ is given by

$$\begin{pmatrix} e_{1,3}^2 + 2e_{1,3}e_{1,4} + e_{1,4}^2 + 1 \\ 2e_{1,3}e_{1,4} + e_{1,4}^2 \\ e_{1,4}^2 + 2e_{1,3} \\ e_{1,3}^2 + 2e_{1,3}e_{1,4} + e_{1,4}^2 + 2e_{1,4} \end{pmatrix}$$

Then by computing

$$(1 + e_{1,3}\alpha^2 + e_{1,4}\alpha^3)(\alpha + e_{2,3}\alpha^2 + e_{1,4}\alpha^3),$$

and

$$(\alpha + e_{2,3}\alpha^2 + e_{1,4}\alpha^3)^2,$$

one gets the 2 remaining columns.

Overall:

$$\boldsymbol{U}_{\text{symb}} = \left(E_1^2 \;\middle|\; E_1E_2 \;\middle|\; E_2^2\right) \in \mathbb{F}_3[e_{1,3}, e_{1,4}, e_{2,3}, e_{2,4}]^{4\times 3}$$

where

$$E_1^2 := \begin{pmatrix} e_{1,3}^2 + 2e_{1,3}e_{1,4} + e_{1,4}^2 + 1 \\ 2e_{1,3}e_{1,4} + e_{1,4}^2 \\ e_{1,4}^2 + 2e_{1,3} \\ e_{1,3}^2 + 2e_{1,3}e_{1,4} + e_{1,4}^2 + 2e_{1,4} \end{pmatrix},$$

$$E_1E_2 := \begin{pmatrix} e_{1,3}e_{2,3} + e_{1,4}e_{2,3} + e_{1,3}e_{2,4} + e_{1,4}e_{2,4} + e_{1,4} \\ e_{1,4}e_{2,3} + e_{1,3}e_{2,4} + e_{1,4}e_{2,4} + 1 \\ e_{1,4}e_{2,4} + e_{2,3} \\ e_{1,3}e_{2,3} + e_{1,4}e_{2,3} + e_{1,3}e_{2,4} + e_{1,4}e_{2,4} + e_{1,3} + e_{1,4} + e_{2,4} \end{pmatrix},$$

$$E_2^2 := \begin{pmatrix} e_{2,3}^2 + 2e_{2,3}e_{2,4} + e_{2,4}^2 + 2e_{2,4} \\ 2e_{2,3}e_{2,4} + e_{2,4}^2 \\ e_{2,4}^2 + 1 \\ e_{2,3}^2 + 2e_{2,3}e_{2,4} + e_{2,4}^2 + 2e_{2,3} + 2e_{2,4} \end{pmatrix}.$$

5.2.2.2 The minors modeling

Using the aforementioned matrix $\boldsymbol{U}_{\text{symb}}$, we can derive an algebraic modeling for the **SquareSpace** problem, namely the minors modeling, see Modeling 3.

Modeling 3 (**SquareSpace** minors modeling)**.** *Let $U = E^2$ be a* **SquareSpace** *instance of parameters (q, m, r). Let $\boldsymbol{U} \in \mathbb{F}_q^{m \times t}$ be a basis matrix of the space U, and let $\boldsymbol{U}_{symb}$ be the $m \times t$ matrix of polynomials previously defined. For all i in $\{1..t\}$, we denote by $\boldsymbol{U}_{symb}^{[i]}$ the i^{th} column of the matrix $\boldsymbol{U}_{symb}$.*

Then, the minors modeling is given by

$$\left\{ f = 0 \;\middle|\; f = \left| \left(\boldsymbol{U} \middle| \boldsymbol{U}_{symb}^{[i]} \right) \right|_{T,*}, \ \forall T \subset \{1..t+1\}, \ \forall i \in \{1..t\}, \right\} \tag{5.7}$$

The minors modeling has the following properties:

- *it contains $(m-r)r$ variables over $\mathbb{F}_q$,*
- *it contains $t\binom{m}{t+1}$ polynomial equations with coefficients in $\mathbb{F}_q$,*
- *its polynomials have degree at most 2, this is readily seen by looking at the Laplace expansion of the maximal minors with respect to their last column.*

The fact that the coefficients of the basis matrix of E in systematic form are solution to the minors modeling is straightforward. The converse is also true by construction and thanks to the identity block which guarantees that any solution leads to a space E' of dimension precisely r.

In what follows, we will see that among the equations of Modeling 3, a lot are linear combinations of the others that we do not need. Then, we will give a way to only generate the equations of interest, *i.e.*, the set of linearly independent equations.

For instance, with parameters $(q, m, r) = (3, 15, 3)$, one has 38610 equations given by the maximal minors but only $54 = (m-t)t$ of them are linearly independent, see the experimental results in Table 5.4.

We were able to prove this result, see Proposition 14.

Proposition 14. *Assuming that the matrix $\boldsymbol{U}$ behaves like a matrix picked at random in $\mathbb{F}_q^{m\times t}$, among the $t\binom{m}{t+1}$ equations given by the minors modeling, see Modeling 3, generically, at most*

$$(m-t)t$$

are linearly independent.

Proof. The idea of the proof is to write each maximal minor as a vector-matrix product between a vector of maximal minors of $\boldsymbol{U}$ and a column of $\boldsymbol{U}_{\text{symb}}$.

Then, using the following theorem, one notices that this vector of maximal minors is an element belonging to the left-kernel of $\boldsymbol{U}$. Theorem 14 has been proved in [Onn94] as mentioned in [Spa12], however the formulation we use is taken from [BBB+20].

Theorem 14 ([Onn94])**.** *Let $\boldsymbol{M}$ be an $M\times t$ matrix in $\mathbb{F}_q$.*

If $t<M$, then generically the left kernel of $\boldsymbol{M}$ is generated by vectors whose coefficients are maximal minors of $\boldsymbol{M}$, specifically vectors of the form

$$\boldsymbol{V}_J = (\ldots, \underbrace{0}_{j\notin J}, \ldots, \underbrace{(-1)^{l+1}\,|\boldsymbol{M}|_{J\setminus\{j\},*}}_{j\in J,\ j=j_l}, \ldots)_{1\le j\le M}$$

where $J=\{j_1<j_2<\ldots<j_{t+1}\}\subset\{1,2,\ldots,M\}$, $\#J=t+1$.

For $T=\{T_1<T_2<\ldots<T_{t+1}\}\subset\{1..t+1\}$ such that $\#T=t+1$, and $k\in\{1..t\}$, by definition

$$\left|\left(\boldsymbol{U}\middle|\boldsymbol{U}^{[k]}_{\text{symb}}\right)\right|_{T,*}=0$$

gives an equation in the minor modeling. In what follows, we denote by $\boldsymbol{U}^{[l,k]}_{\text{symb}}$ the (l,k)-entry in the matrix $\boldsymbol{U}_{\text{symb}}$.

One has that

$$\left|\left(\boldsymbol{U}\middle|\boldsymbol{U}^{[k]}_{\text{symb}}\right)\right|_{T,*}=\sum_{i\in T,\ i=T_j}(-1)^{j+1}\boldsymbol{U}^{[i,k]}_{\text{symb}}\,|\boldsymbol{U}|_{T\setminus\{i\},*}$$

which can be rewritten as a vector-matrix product:

$$(\ldots,\underbrace{0}_{i\notin T},\ldots,\underbrace{(-1)^{j+1}\,|\boldsymbol{U}|_{T\setminus\{i\},*}}_{i\in T,\ i=T_j},\ldots)_{1\le j\le M}\ \times\ \boldsymbol{U}^{[k]}_{\text{symb}}$$

Using Theorem 14, this means that the equations

$$\left|\left(\boldsymbol{U}|\boldsymbol{U}_{\text{symb}}^{[k]}\right)\right|_{T,*} = 0$$

can be written as $\boldsymbol{v}\boldsymbol{U}_{\text{symb}}^{[k]} = 0$ where $\boldsymbol{v}$ is a vector in the left kernel of $\boldsymbol{U}$.

Since $\boldsymbol{U}$ has rank t by definition, its kernel has rank $m-t$, and there are t columns in $\boldsymbol{U}_{\text{symb}}$, this means that there cannot be more than $(m-t)t$ linearly independent equations in the minors modeling. □

Note that the assumption about the matrix $\boldsymbol{U}$ in Proposition 14 is supported by our experiments.

5.2.2.3 The kernel modeling

From Proposition 14, one can easily derive a new algebraic modeling for the **SquareSpace** problem by computing the left-kernel of $\boldsymbol{U}$ and then multiply $\boldsymbol{U}_{\text{symb}}$ by it.

This can be seen as a "nice" modeling, since one will not have to compute $t\binom{m}{t+1}$ maximal minors, but she will instead directly get the $(m-t)t$ equations.

We call this modeling of the **SquareSpace** problem the *kernel modeling*, see Modeling 4.

Modeling 4 (**SquareSpace** kernel modeling)**.** *Let $U = E^2$ be a* **SquareSpace** *instance of parameters (q, m, r). Let $\boldsymbol{U} \in \mathbb{F}_q^{m\times t}$ be a basis matrix of the space U, and let $\boldsymbol{U}_{symb}$ be the $m \times t$ matrix of polynomials previously defined.*

Then, the minors modeling is given by

$$\text{Ker}_L(\boldsymbol{U})\boldsymbol{U}_{symb} = \boldsymbol{0}^{(m-t)\times t} \tag{5.8}$$

The minors modeling has the following properties:

- *it contains $(m-r)r$ variables over $\mathbb{F}_q$,*
- *it contains at most $(m-t)t$* linearly independent *polynomials equations with coefficients in $\mathbb{F}_q$,*
- *its polynomials have degree at most 2.*

Last but not least, in the previous section and in Modeling 4, we mentioned that at most $(m-t)t$ equations could be linearly independent; in what follows, we show that this bound is generically reached, see Proposition 15.

Proposition 15. *The kernel modeling, see Modeling 4, when used without any specialization, i.e., with mr variables $e_{i,j}$'s, yields a system of $(m-t)t$ quadratic polynomial equations, and they are all* linearly independent *over $\mathbb{F}_q$.*

In other words, the rank of the Macaulay matrix of the kernel system at degree 2 is always of full rank.

Proof. Recall that α is a primitive element and thus $\beta := (1, \alpha, \alpha^2, \ldots, \alpha^{m-1})$ is a basis of $\mathbb{F}_{q^m}$ over $\mathbb{F}_q$. Let us start by considering a product $e_i e_j$ where all the coefficients over $\mathbb{F}_q$ are variables, i.e. without the aforementioned specialization.

This product can be written as follow:

$$(e_{i,1} + e_{i,2}\alpha + e_{i,3}\alpha^2 + \ldots + e_{i,m}\alpha^{m-1})(e_{j,1} + e_{j,2}\alpha + e_{j,3}\alpha^2 + \ldots + e_{j,m}\alpha^{m-1})$$

Writing $e_i e_j$ in the basis β, one notices that the monomials $e_{i,1}e_{j,1}$ will only appear in the coefficient of 1.

Similarly, the monomial $e_{i,1}e_{j,2}$ will only appear in the coefficient of α, and so on.

An equation in the kernel modeling corresponds to a vector-matrix product between a row of the $(m-t) \times m$ kernel matrix and a column of the symbolic basis $\boldsymbol{U}_{\text{symb}}$.

It is readily seen that for a given kernel row, the t corresponding equations will be linearly independent since they involve different pairwise products $e_i e_j$'s.

For a given column $e_i e_j$, assuming without loss of generality that the kernel matrix is in systematic form, and using the aforementioned fact, it is straightforward that every polynomial will contain a monomial that none of the other polynomials contains.

Thus all the $(m-t)t$ equations are linearly independent. □

Proposition 15 is a bit frustrating to us in the sense that it gives the linear independence only without the $r \times r$ identity block in $\boldsymbol{E}$; however, we are convinced by extensive experiments that this result generalizes when using $\boldsymbol{E}$ in systematic form as described in Modeling 4.

See for instance the experimental results in Table 5.4, for which we always used the specialized kernel modeling.

(q, m, r)	# var.	# eq. Mod. 3	# indep.	$(m-t)t$	Time GB (s)	d
$[3, 9, 3]$	18	216	18	18	≈ 2	5
$[3, 11, 3]$	24	1980	30	30	≈ 100	6
$[3, 13, 3]$	30	10296	42	42	≈ 200	5
$[3, 15, 3]$	36	38610	54	54	≈ 1500	5
$[5, 9, 3]$	18	216	18	18	≈ 300	6
$[5, 11, 3]$	24	1980	30	30	≈ 13000	6
$[5, 13, 3]$	30	10296	42	42	`crash`	-
$[5, 15, 3]$	36	38610	54	54	`crash`	-

Table 5.4: Experimental results of our algebraic attacks against **SquareSpace**. For a given set of parameters, the "var" column gives the number of variables which is the same for both modelings, then there is the number of equations in Modeling 3, and the number of linearly independent among them. The column "Time GB" indicates the time needed (in seconds) to compute the Gröbner basis (with the field equations) using the F4 implementation by Allan Steel in `Magma V2.27-1` on a personal computer with 16GB of RAM. When it is written "`crash`", it means that the RAM saturated during the computation, causing the process to crash. Last but not least, the "d" column gives the greatest degree reached during this computation, see Section 2.3.2.

5.3 SquareSpace Challenges

According to our complexity estimations of the **SquareSpace** problem, see Definition 43, we propose 4 challenges in Table 5.5.

	q	m	r	Security
SquareSpace I	79	251	5	128
SquareSpace II	1451	131	4	128
SquareSpace III	65521	131	4	128
SquareSpace IV	2097169	167	3	128

Table 5.5: **SquareSpace** challenges parameters.

Based on the attacks described in this document, we think that they would require

	Irreducible Polynomial in $\mathbb{F}_q[X]$
SquareSpace I	$X^{251} + X^2 + X + 4$
SquareSpace II	$X^{131} + 2X + 60$
SquareSpace III	$X^{131} + X + 9$
SquareSpace IV	$X^{167} + X + 29$

Table 5.6: Irreducible Polynomials in $\mathbb{F}_q[X]$ used to define $\mathbb{F}_{q^m}$ for the **SquareSpace** challenges given in Table 5.5.

2^{128} elementary bit operations to be solved. This correspond to the first level of security in NIST Post-Quantum Standardization process.

Challenge instances can be generated using our program, see Section 5.4.3, available at

https://github.com/MaximeBros/SquareSpace_signature.

5.4 Application to Cryptography

A natural way to use the **SquareSpace** problem in cryptography is to look at cryptosystems which rely on the use of a "similar" problem. This is for instance the case of the Fiat-Shamir authentication protocol, see Section 2.4.2.

5.4.1 Our authentication scheme

The authentication scheme described in Figure 5.1 is our adaptation of the aforementioned Fiat-Shamir authentication protocol using the **SquareSpace** problem, see Definition 43.

It is readily seen that recovering the private key from the public key is exactly an instance of the **SquareSpace** problem.

The rest of this section is dedicated to prove the correctness, and the soundness of this protocol, as defined in Section 2.4.2.

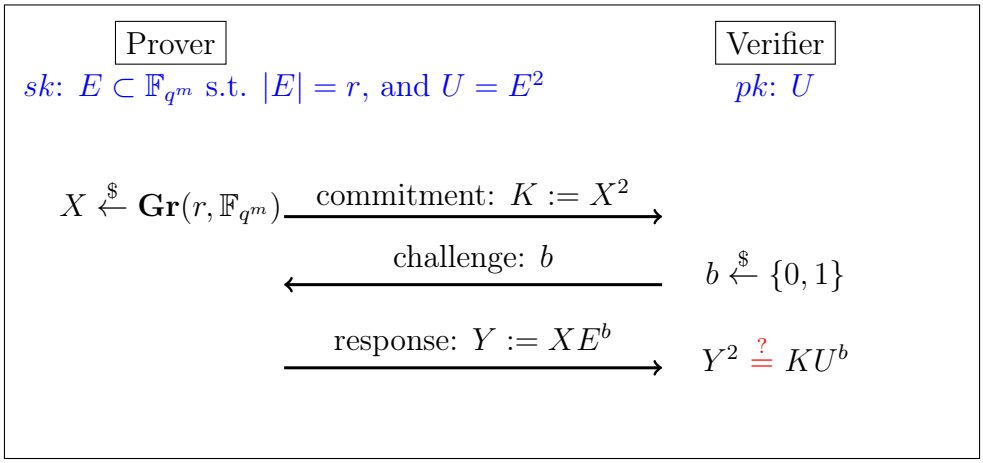


Figure 5.1: Our authentication scheme based on the **SquareSpace** problem.

The correctness is straightforward, similarly to the Fiat-Shamir authentication protocol, one has that $(XE)^2 = KU$ by construction and using the commutativity of the multiplication of vector spaces in $\mathbb{F}_{q^m}$.

The soundness probability is $1/2$ as well, however the tools used to prove it are a bit trickier than for the Fiat-Shamir authentication scheme.

First of all, it is clear that a cheater can predict the challenge $b = 0$ very easily, thus, for the same reasons mentioned in Section 2.4.2, it means that an adversary can cheat with probability at least $1/2$.

To prove that the soundness probability is exactly $1/2$, we will proceed as we did in the previous section and consider an adversary that could predict both challenges; this is the point of Proposition 16.

Proposition 16. *Let us assume the uniqueness for solution of the* **SquareSpace** *problem, see Section 5.1.2.*

If an adversary can predict both challenges in the authentication scheme depicted in Figure 5.1, then she can solve the **SquareSpace** *problem with overwhelming probability.*

Proof. Let us assume that an adversary can predict both challenges, thus she knows Y_1 and Y_2 such that :

$$\begin{cases} Y_1^2 = KU \\ Y_2^2 = K \end{cases}.$$

By assumption, the solutions to **SquareSpace** instances are unique, thus

$$\begin{cases} Y_1 = XE \\ Y_2 = X \end{cases}.$$

Let $x_1, x_2, \ldots, x_r \in \mathbb{F}_{q^m}$ be a basis of the secret space X. In order to recover E from Y_1 and Y_2, the adversary can compute the following intersections:

$$\bigcap_{i=1}^{r} x_1^{-1} Y_1. \tag{5.9}$$

Using results from LRPC decoding algorithm, see [AGH+19], the intersection in Equation (5.9) will be exactly E with overwhelming probability. □

5.4.2 Our signature scheme

Using the Fiat-Shamir heuristic, see Section 2.4.3, we can turn our authentication scheme into a signature scheme whose security relies on the **SquareSpace** problem.

However, to claim that this signature scheme is **EUF-CMA** in the ROM, one needs the zero-knowledge property, see Section 2.4.3. Indeed, with this property, our authentication scheme would become an interactive zero-knowledge proof of knowledge, and the results from Pointcheval and Stern would apply.

Even if this last property is not proved yet, as mentioned in Chapter 6, we propose parameters for our **SquareSpace** signature scheme together with an implementation in C, this is the topic of the last sections of this document.

5.4.2.1 Parameters

Table 5.7 gives our proposed parameters for the signature scheme based on the **SquareSpace** problem. They are matching the challenges given in Section 5.3. Since these challenges target the security level of 128 bits, we took $\lambda = 128$ for the number of iterations in the Fiat-Shamir heuristic.

Table 5.7 also gives the sizes of the public key pk and of the signature $|\sigma|$ for each set of parameters.

	Parameters (q, m, r)	$\lvert\mathsf{pk}\rvert$	$\lvert\sigma\rvert$
SquareSpace Signature I	$(79, 251, 5)$	**3.1** KB	**1.0** MB
SquareSpace Signature II	$(1451, 131, 4)$	**1.7** KB	**0.5** MB
SquareSpace Signature III	$(65521, 131, 4)$	**2.4** KB	**0.8** MB
SquareSpace Signature IV	$(2097169, 167, 3)$	**2.7** KB	**0.8** MB

Table 5.7: Parameters for our **SquareSpace** signature scheme for security 128 bits. The sizes are expressed in kilo-bytes (KB) and mega-bytes (MB).

These sizes, in bytes, where computed using the following formulas:

$$|\mathbf{pk}| = \left\lceil \frac{(m-t) \times t \times \lceil \log_2(q) \rceil}{8} \right\rceil, \tag{5.10}$$

$$|\sigma| = \frac{128}{8} \times \left((m-t) \times t \times \lceil \log_2(q) \rceil + (m-r^2) \times r^2 \times \lceil \log_2(q) \rceil \right) \tag{5.11}$$

where $t := (r+1)r/2$.

Note that the size of the secret key **sk** is not relevant since it is only a seed, thus it always has size 40 bytes.

Last but not least, we store all $M \times N$ matrices, where $M > N$, in systematic form, this is why it only requires $(M-N)N$ entries for each matrix, the rest being an identity block.

Remark 26. *It is worth noticing that the signature sizes given in Table 5.7 using Equation (5.11) are smaller in practice. Indeed, about half of the time, when the challenge bit is 0, only $Y = X$ is sent, and it has dimension r, whereas $Y = XE$ has dimension r^2 when the challenge bit is 1.*

Thus, the signature sizes given in Table 5.7 correspond to the maximum size a signature could have in the worst case.

5.4.3 Implementation

5.4.3.1 Presentation

We propose a pure C implementation of the signature scheme described in Section 5.4.2. This implementation is available at

https://github.com/MaximeBros/SquareSpace_signature.

This is a joint work with Frédéric Canaud.

Our code uses the following libraries:

- FLINT (Fast Library for Number Theory) [Har11] (which depends on [FHL+07, Gra96]) for the operations in $\mathbb{F}_{q^m}$, and the linear algebra operations on matrices with entries in $\mathbb{F}_q$.
- `arc4random` pseudo-random number generator by David Mazieres and Damien Miller,
- SHA2 implementation by D. J. Bernstein for the hash function SHA256.

5.4.3.2 Execution times

Table 5.8 gives the times we obtained when executing our C code for the three signature algorithms: KeyGen, Sign, Verif.

These benchmarks have been performed on a personal computer that has 8 gigabytes of memory, and an Intel® CoreTM i5-8300H 2.30GHz CPU. Note that the Hyper-Threading, the Turbo Boost, and the SpeedStep features were disabled, and the `gcc` compiler version was `11.2.0`.

	Key generation	Signature	Verification
SquareSpace I	0.9 ms	178 ms	1292 ms
SquareSpace II	0.4 ms	63 ms	256 ms
SquareSpace III	0.4 ms	66 ms	270 ms
SquareSpace IV	0.3 ms	53 ms	138 ms

Table 5.8: Average times (in milliseconds) over 100 executions of each algorithm of our **SquareSpace** signature scheme implementation, for each set of parameters.

Chapter 6

Perspectives

Attacks against RD and its variants

Rank-metric complexity estimator. There are a lot of different attacks for each problem in rank-based cryptography, such as **RD**, **RSL**, **NHRD**, etc.

Even if the complexities of each attack are available in research papers, we want to develop a useful software that everybody could use to compute the complexities of all state-of-the-art attacks for a given parameter set.

This would be similar to what exists for multivariate-based cryptography with the tool developed by Bellini, Makarim, Sanna, and Verbel [BMSV22].

MaxMinors and SupportMinors implementations. As we performed a lot of experiments to study the MaxMinors and SupportMinors attacks, we do have `Magma` implementations for both.

However, we would like to propose a more efficient C/C++ implementation of these attacks, for instance taking advantage of the state-of-the-art libraries for dense linear algebra (NTL, Flint, fflas-ffpack, m4ri/m4rie), and for sparse linear algebra (polycephaly, linbox).

Formulas for the complexity of MaxMinors, SupportMinors, and bilinear systems in general. In a few words, to compute the complexity of the MaxMinors attack against an **RD** (q, m, n, k, r) instance, one has to find the smallest integer a such that

$$m\binom{n-k-1}{r} \geq \binom{n-a}{r} - 1, \tag{6.1}$$

and then the complexity is given by

$$q^{ar} m \binom{n-k-1}{r} \binom{n-a}{r}^{\omega-1}.$$

This formula is very different from the classical complexity formulas, usually written as a power of q, where the exponent is a rather simple function depending on the parameters.

We recently managed to find a closed formula of this form for the MaxMinors attack. However, we need to go deeper in its analysis to both make it more accurate and study its error thoroughly.

In addition to this, we would like to extend this analysis to the complexity of the SupportMinors attack, and even to the complexity of bilinear systems in general, as these complexity formulas all have the same form.

Algebraic attacks against RSL. The **RSL** problem, see Definition 18 and Section 3.3.2, is at the core of some cryptosystems [Wan19, ABG+19, AAD+22], and it will very likely be important in the new NIST Call for Post-Quantum Signatures where some candidates might rely on it.

However, as described in [BB21] and in Section 3.3.2.2, there is a threshold below which these algebraic attacks are not well understood. More precisely, the aforementioned algebraic attacks have complexity formulas when $N > n-k-r$. Nonetheless, they could work for smaller values of N, as we experimentally checked it.

We would like to study this specific area and derive a complexity formula valid for any value of N.

Algebraic attacks against NHRSL. In Section 3.3.3, we described the first attack against the **NHRSL** problem, recall that it is of combinatorial nature.

Although we feel that the adaptation of the algebraic attacks against **NHRD** and **RSL** to non-homogeneous error support learning will not be straightforward, this is an interesting task that we look forward to studying.

Algebraic attack against RD. In the most recent attack against the **RD** problem [BBB+22], one part remains to be solved: what is the complexity when the equations are projected from $\mathbb{F}_{q^m}$ to $\mathbb{F}_q$? Despite conducting many experiments, the precise number of linearly independent equations has not yet been found.

Continuing this work would be interesting, as it is the last promising lead we have concerning the algebraic attacks against **RD**.

The PSSI problem.

The **PSSI** problem is very important for the security of the Durandal signature scheme [ABG+19].

Even if the reduction from **PSSI** to **MinRank** described in Section 3.4 enabled us to benefit from all the attacks against **MinRank**, both combinatorial and algebraic, the complexities we got from them are still far too high.

We would like to take advantage of the strong structure of these **MinRank** instances in order to improve these attacks against **PSSI**.

Last but not least, it would be very interesting to have algebraic attacks against **PSSI** that take advantage of the several instances one is given, like the current combinatorial attacks do.

Miscellaneous

Improve the decoding of LRPC codes. The decoding of Low Rank Parity-Check codes (LRPC) [AGH+19] relies on computing several intersections, which is done by performing linear algebra on block matrices.

We would like to improve this process by solving a slightly different linear system for which we would try to take advantage of its structure.

It is not clear if this new decoding algorithm would be more efficient for every LRPC code; therefore we would have to look for the specific parameter areas where our new approach would beat the classic one, if they exist.

MQ-cyclic. The Multivariate Quadratic problem (MQ) asks one to find a solution to a system of multivariate polynomial equations of degree 2 over $\mathbb{F}_q$. The decisional version of this problem is NP-complete and it is clearly connected to the algebraic attacks studied in this document; indeed, an algebraic attack is nothing but an instance of MQ, usually with a strong structure.

Some signature schemes rely on MQ, see for instance [CHR+16, Beu20, BG22]. Quasi-cyclic or ideal codes, see Section 2.1.3, were introduced in code-based cryptography to reduce key sizes; in the same spirit, an adapted notion of cyclicity can be used for MQ.

We would like to study the complexity of this problem, that is to say, check if this cyclicity structure can be exploited.

SquareSpace encryption and more.

The **SquareSpace** problem has been introduced in this document and we described a few attacks against it.

As for the algebraic attacks, we need to further investigate their solving degree in order to derive more precise complexity formulas, as we currently have rough estimations.

We would also like to study the zero-knowledge part of our authentication scheme based on the **SquareSpace** problem, see Section 5.4. That is to say, make sure that when one is given a lot of **SquareSpace** instances relying on the same secret space, it does not leak secret information that an attacker could retrieve using less computations than the level of security allows.

In addition to this, we would like to study a more general version of this problem, namely the factorization of vector spaces over $\mathbb{F}_{q^m}$. More precisely, let r, d be positive integers, one is given a vector space $X \subset \mathbb{F}_{q^m}$ of dimension rd and tries to find whether there exist spaces A and B of dimensions r and d, respectively, such that $X = AB$. This more general problem is mentioned in the MURAVE rank-based signature scheme [LT20a].

Last but not least, we have a promising idea that could lead to the construction of a **SquareSpace**-based encryption scheme. As it would be exciting to have both a signature scheme and an encryption scheme that rely on the **SquareSpace** problem, we look forward to studying this.

List of Algorithms

List of Figures

List of Tables

Bibliography

[AAAS+19] Gorjan Alagic, Gorjan Alagic, Jacob Alperin-Sheriff, Daniel Apon, David Cooper, Quynh Dang, Yi-Kai Liu, Carl Miller, Dustin Moody, Rene Peralta, et al. Status Report on the First Round of the NIST Post-Quantum Cryptography Standardization Process. Technical report, National Institute of Standards and Technology Gaithersburg, MD, 2019.

[AAB+20] Carlos Aguilar Melchor, Nicolas Aragon, Slim Bettaieb, Loïc Bidoux, Olivier Blazy, Maxime Bros, Alain Couvreur, Jean-Christophe Deneuville, Philippe Gaborit, Gilles Zémor, and Adrien Hauteville. Rank Quasi-Cyclic (RQC). Second Round submission to NIST Post-Quantum Cryptography Call, 2020.

[AAB+21a] Carlos Aguilar Melchor, Nicolas Aragon, Paulo Barreto, Slim Bettaieb, Loïc Bidoux, Olivier Blazy, Jean-Christophe Deneuville, Philippe Gaborit, Shay Gueron, Tim Güneysu, Rafael Misoczki, Edoardo Persichetti, Nicolas Sendrier, Jean-Pierre Tillich, and Gilles Zémor. BIKE. Round 3 Submission to the NIST Post-Quantum Cryptography Call, 2021.

[AAB+21b] Carlos Aguilar Melchor, Nicolas Aragon, Slim Bettaieb, Loïc Bidoux, Olivier Blazy, Jean-Christophe Deneuville, Philippe Gaborit, Edoardo Persichetti, Gilles Zémor, and Jurjen Bos. HQC. Round 3 Submission to the NIST Post-Quantum Cryptography Call, 2021.

[AAC+22] Gorjan Alagic, Daniel Apon, David Cooper, Quynh Dang, Thinh Dang, John Kelsey, Jacob Lichtinger, Yi-Kai Liu, Carl Miller, Dustin Moody, Rene Peralta, Ray Perlner, Angela Robinson, and Daniel Smith-Tone. Status Report on the Third Round of the NIST Post-Quantum Cryptography Standardization Process. Technical report, National Institute of Standards and Technology Gaithersburg, MD, 2022.

[AAD+22] Carlos Aguilar Melchor, Nicolas Aragon, Victor Dyseryn, Philippe Gaborit, and Gilles Zémor. LRPC codes with multiple syndromes: near ideal-size KEMs without ideals. In *PQCrypto*, 2022.

[ABD+19] Nicolas Aragon, Olivier Blazy, Jean-Christophe Deneuville, Philippe Gaborit, Adrien Hauteville, Olivier Ruatta, Jean-Pierre Tillich, Gilles Zémor, Carlos Aguilar Melchor, Slim Bettaieb, Loïc Bidoux, Bardet Magali, and Ayoub Otmani. ROLLO (merger of Rank-Ouroboros, LAKE and LOCKER). Second round submission to the NIST Post-Quantum Cryptography Call, 2019.

[ABD+21] Erdem Alkim, Joppe W Bos, Léo Ducas, Patrick Longa, and Ilya Mironov. Frodokem. 3rd Round submission to the NIST. 2021.

[ABG+19] Nicolas Aragon, Olivier Blazy, Philippe Gaborit, Adrien Hauteville, and Gilles Zémor. Durandal: a rank metric based signature scheme. In *Eurocrypt*, 2019.

[ABG21] Nicolas Aragon, Maxime Bros, and Philippe Gaborit. Cryptanalysis of the Rank Preserving Signature. In *IMA International Conference on Cryptography and Coding (IMACC)*, 2021.

[AGH+19] Nicolas Aragon, Philippe Gaborit, Adrien Hauteville, Olivier Ruatta, and Gilles Zémor. Low Rank Parity Check Codes: New Decoding Algorithms and Applications to Cryptography. *IEEE Transactions on Information Theory (IT)*, 2019.

[AGHT18] Nicolas Aragon, Philippe Gaborit, Adrien Hauteville, and Jean-Pierre Tillich. A new algorithm for solving the rank syndrome decoding problem. In *2018 IEEE International Symposium on Information Theory (ISIT)*, 2018.

[Ajt96] Miklós Ajtai. Generating hard instances of lattice problems. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, 1996.

[Ale03] Alekhnovich, Michael. More on Average Case vs Approximation Complexity. In *44th Symposium on Foundations of Computer Science (FOCS 2003)*, 2003.

[ALR18] Daniel Augot, Pierre Loidreau, and Gwezheneg Robert. Generalized Gabidulin codes over fields of any characteristic. *Designs, Codes and Cryptography (DCC)*, 2018.

[AMBD+18] Carlos Aguilar Melchor, Olivier Blazy, Jean-Christophe Deneuville, Philippe Gaborit, and Gilles Zémor. Efficient Encryption From Random Quasi-Cyclic Codes. *IEEE Transactions on Information Theory (IT)*, 2018.

[Bar04] Magali Bardet. *Étude des systèmes algébriques surdéterminés. Applications aux codes correcteurs et à la cryptographie*. PhD thesis, Université Paris VI, France, 2004.

[Bar09] Gregory Bard. *Algebraic cryptanalysis*. Springer Science & Business Media, 2009.

[BB21] Magali Bardet and Pierre Briaud. An algebraic approach to the Rank Support Learning problem. In *PQCrypto*, 2021.

[BBB+20] Magali Bardet, Pierre Briaud, Maxime Bros, Philippe Gaborit, Vincent Neiger, Olivier Ruatta, and Jean-Pierre Tillich. An Algebraic Attack on Rank Metric Code-Based Cryptosystems. In *Eurocrypt*, 2020.

[BBB+22] Magali Bardet, Pierre Briaud, Maxime Bros, Philippe Gaborit, and Jean-Pierre Tillich. Revisiting Algebraic Attacks on MinRank and on the Rank Decoding Problem. 2022. URL: https://arxiv.org/abs/2208.05471.

[BBBG22] Loïc Bidoux, Pierre Briaud, Maxime Bros, and Philippe Gaborit. RQC revisited and more cryptanalysis for Rank-based Cryptography. 2022. URL: https://arxiv.org/abs/2207.01410.

[BBC+20] Magali Bardet, Maxime Bros, Daniel Cabarcas, Philippe Gaborit, Ray Perlner, Daniel Smith-Tone, Jean-Pierre Tillich, and Javier Verbel. Improvements of Algebraic Attacks for solving the Rank Decoding and MinRank problems. In *Asiacrypt*, 2020.

[BBC+22] John Baena, Pierre Briaud, Daniel Cabarcas, Ray Perlner, Daniel Smith-Tone, and Javier Verbel. Improving Support-Minors rank attacks: applications to G*e*MSS and Rainbow. *Crypto*, 2022.

[BCL+17] Daniel J Bernstein, Tung Chou, Tanja Lange, Ingo von Maurich, Rafael Misoczki, Ruben Niederhagen, Edoardo Persichetti, Christiane Peters, Peter Schwabe, Nicolas Sendrier, et al. Classic McEliece. 2017.

[BDK+18] Joppe Bos, Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, John M Schanck, Peter Schwabe, Gregor Seiler, and Damien Stehlé. CRYSTALS-Kyber: a CCA-secure module-lattice-based KEM. In *2018 IEEE European Symposium on Security and Privacy (EuroS&P)*, 2018.

[BESV22] Emanuele Bellini, Andre Esser, Carlo Sanna, and Javier Verbel. MR-DSS – Smaller MinRank-based (Ring-)Signatures. In *PQCrypto*, 2022.

[Beu20] Ward Beullens. Sigma protocols for MQ, PKP and SIS, and fishy signature schemes. In *Eurocrypt*, 2020.

[Beu22] Ward Beullens. Breaking Rainbow takes a weekend on a laptop. *Crypto*, 2022.

[BFS99] Jonathan F Buss, Gudmund S Frandsen, and Jeffrey O Shallit. The computational complexity of some problems of linear algebra. *Journal of Computer and System Sciences*, 1999.

[BG22] Loïc Bidoux and Philippe Gaborit. Shorter Signatures from Proofs of Knowledge for the SD, MQ, PKP and RSD Problems. 2022. URL: https://arxiv.org/abs/2204.02915.

[BGL03] Richard P Brent, Shuhong Gao, and Alan GB Lauder. Random Krylov spaces over finite fields. *SIAM Journal on Discrete Mathematics*, 2003.

[BH74] James R Bunch and John E Hopcroft. *Triangular factorization and inversion by fast matrix multiplication*. PhD thesis, 1974.

[BMSV22] Emanuele Bellini, Rusydi H Makarim, Carlo Sanna, and Javier Verbel. An Estimator for the Hardness of the MQ Problem. *Cryptology ePrint Archive*, 2022.

[BMvT78] Elwyn Berlekamp, Robert McEliece, and Henk van Tilborg. On the inherent intractability of certain coding problems (corresp.). *IEEE Transactions on Information Theory (IT)*, 1978.

[BR93] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *ACM CCS*, 1993.

[Buc65] Bruno Buchberger. Ein Algorithmus zum Auffinden der Basiselemente des Restklassenringes nach einem nulldimensionalen Polynomideal. *PhD thesis, Universitat Insbruck*, 1965.

[BV06] Winfried Bruns and Udo Vetter. *Determinantal rings*, volume 1327. Springer, 2006.

[C+16] Lily Chen et al. NISTIR 8105, "Report on Post-Quantum Cryptography". Technical report, National Institute of Standards and Technology Gaithersburg, MD, 2016.

[CB22] Alain Couvreur and Maxime Bombar. Right-hand side decoding of Gabidulin codes and applications. In *WCC 2022*, 2022.

[CCMZ15] Ignacio Cascudo, Ronald Cramer, Diego Mirandola, and Gilles Zémor. Squares of random linear codes. *IEEE Transactions on Information Theory (IT)*, 2015.

[CD22] Wouter Castryck and Thomas Decru. An efficient key recovery attack on SIDH (preliminary version). *Cryptology ePrint Archive*, 2022. URL: https://eprint.iacr.org/2022/975.

[CFMR+19] Antoine Casanova, Jean-Charles Faugère, Gilles Macario-Rat, Jacques Patarin, Ludovic Perret, and Jocelyn Ryckeghem. GeMSS: A Great Multivariate Short Signature. Second Round Submission to the NIST Post-Quantum Cryptography Call, 2019.

[CG23] Alessio Caminata and Elisa Gorla. Solving degree, last fall degree, and related invariants. *Journal of Symbolic Computation (JSC)*, 2023.

[CGH04] Ran Canetti, Oded Goldreich, and Shai Halevi. The random oracle methodology, revisited. *Journal of the ACM (JACM)*, 2004.

[CHR+16] Ming-Shing Chen, Andreas Hülsing, Joost Rijneveld, Simona Samardjiska, and Peter Schwabe. From 5-pass MQ-based identification to MQ-based signatures. 2016.

[CKPS00] Nicolas Courtois, Alexander Klimov, Jacques Patarin, and Adi Shamir. Efficient algorithms for solving overdefined systems of multivariate polynomial equations. In *Eurocrypt*, 2000.

[CLO92] David Cox, John Little, and Donal O'Shea. *Ideals, varieties, and algorithms*. Springer, 1992.

[Cop94] Don Coppersmith. Solving homogeneous linear equations over GF(2) via block Wiedemann algorithm. *Mathematics of Computation*, 1994.

[Cou01a] Nicolas Courtois. Efficient zero-knowledge authentication based on a linear algebra problem MinRank. In *Asiacrypt*, 2001.

[Cou01b] Nicolas Courtois. *La sécurité des primitives cryptographiques basées sur les problemes algébriques multivariables MQ, IP, MinRank, et HFE*. PhD thesis, Université Paris VI, France, 2001.

[Cou06] Jean-Marc Couveignes. Hard homogeneous spaces. *Cryptology ePrint Archive*, 2006.

[Cou19] Alain Couvreur. Introduction to Coding Theory, 2019.

[DCP+19] Jintai Ding, Ming-Shing Chen, Albrecht Petzoldt, Dieter Schmidt, and Bo-Yin Yang. Rainbow. Second Round Submission to the NIST Post-Quantum Cryptography Call, 2019.

[Del78] Philippe Delsarte. Bilinear Forms over a Finite Field, with Applications to Coding Theory. *J. Comb. Theory, Ser. A*, 1978.

[DH76] Whitfield Diffie and Martin E Hellman. New Directions in cryptography. In *IEEE Transactions on Information Theory (IT)*. 1976.

[DT18] Thomas Debris-Alazard and Jean-Pierre Tillich. Two attacks on rank metric code-based schemes: RankSign and an Identity-Based-Encryption scheme. In *Asiacrypt*, 2018.

[Fau99] Jean-Charles Faugère. A new efficient algorithm for computing Gröbner bases (F4). *Journal of pure and applied algebra*, 1999.

[Fau02] Jean Charles Faugère. A new efficient algorithm for computing Gröbner bases without reduction to zero (F5) . In *ISSAC*, 2002.

[FHL+07] Laurent Fousse, Guillaume Hanrot, Vincent Lefèvre, Patrick Pélissier, and Paul Zimmermann. MPFR: A multiple-precision binary floating-point library with correct rounding. *ACM Transactions on Mathematical Software (TOMS)*, 2007.

[FLdVP08] Jean-Charles Faugère, Françoise Levy-dit Vehel, and Ludovic Perret. Cryptanalysis of MinRank. In *Crypto*, 2008.

[FS87] Amos Fiat and Adi Shamir. How to Prove Yourself: Practical Solutions to Identification and Signature Problems. In *Crypto*, 1987.

[FSEDS11] Jean-Charles Faugère, Mohab Safey El Din, and Pierre-Jean Spaenlehauer. Gröbner bases of bihomogeneous ideals generated by polynomials of bidegree (1, 1): Algorithms and complexity. *Journal of Symbolic Computation (JSC)*, 2011.

[FSEDS13] Jean-Charles Faugère, Mohab Safey El Din, and Pierre-Jean Spaenlehauer. On the complexity of the generalized MinRank problem. *Journal of Symbolic Computation (JSC)*, 2013.

[Gab85] Ernest M. Gabidulin. Theory of codes with maximum rank distance. *Problemy peredachi informatsii*, 1985.

[GGH+20] Philippe Gaborit, Lucky Galvez, Adrien Hauteville, Jon-Lark Kim, Myeong Jae Kim, and Young-Sik Kim. Dual-Ouroboros: an improvement of the McNie scheme. *Advances in Mathematics of Communications (ACM)*, 2020.

[GHPT17] Philippe Gaborit, Adrien Hauteville, Duong Hieu Phan, and Jean-Pierre Tillich. Identity-based Encryption from Rank Metric. In *Crypto*, 2017.

[GHT16] Philippe Gaborit, Adrien Hauteville, and Jean-Pierre Tillich. RankSynd a PRNG Based on Rank Metric. In *PQCrypto*, 2016.

[GM82] Shafi Goldwasser and Silvio Micali. Probabilistic encryption and how to play mental poker keeping secret all partial information. In *ACM STOC*, 1982.

[GMR85] Shafi Goldwasser, Silvio Micali, and Chales Rackoff. The knowledge complexity of interactive proof-systems (extended abstract). In *ACM STOC*, 1985.

[GMRZ13] Philippe Gaborit, Gaétan Murat, Olivier Ruatta, and Gilles Zémor. Low Rank Parity Check codes and their application to cryptography. In *WCC*, 2013.

[GP08] Ernst M. Gabidulin and Nina I Pilipchuk. Error and erasure correcting algorithms for rank codes. *Designs, Codes and Cryptography (DCC)*, 2008.

[GPT91] Ernst M. Gabidulin, A. V. Paramonov, and O. V. Tretjakov. Ideals over a non-commutative ring and their applications to cryptography. In *Eurocrypt*, 1991.

[Gra96] Torbjörn Granlund. GNU MP. *The GNU Multiple Precision Arithmetic Library*, 1996.

[GRS16] Philippe Gaborit, Olivier Ruatta, and Julien Schrek. On the Complexity of the Rank Syndrome Decoding Problem. *IEEE Transactions on Information Theory (IT)*, 2016.

[GRSZ14a] Philippe Gaborit, Olivier Ruatta, Julien Schrek, and Gilles Zémor. New Results for Rank-Based Cryptography. In *Africacrypt*, 2014.

[GRSZ14b] Philippe Gaborit, Olivier Ruatta, Julien Schrek, and Gilles Zémor. RankSign: An Efficient Signature Algorithm Based on the Rank Metric. In *PQCrypto*, 2014.

[GZ16] Philippe Gaborit and Gilles Zémor. On the hardness of the decoding and the minimum distance problems for rank codes. *IEEE Transactions on Information Theory (IT)*, 2016.

[Har11] William B Hart. Flint: Fast library for number theory. *Computeralgebra-Rundbrief: Vol. 49*, 2011.

[HPS98] Jeffrey Hoffstein, Jill Pipher, and Joseph H. Silverman. NTRU: A Ring-Based Public Key Cryptosystem. In *Algorithmic Number Theory (ANTS)*, 1998.

[JF11] David Jao and Luca De Feo. Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. In *International Workshop on Post-Quantum Cryptography*, 2011.

[Jou09] Antoine Joux. *Algorithmic cryptanalysis*. Chapman and Hall/CRC, 2009.

[Kal95] Erich Kaltofen. Analysis of Coppersmith's block Wiedemann algorithm for the parallel solution of sparse linear systems. *Mathematics of Computation*, 1995.

[KL20] Jonathan Katz and Yehuda Lindell. *Introduction to modern cryptography*. CRC press, 2020.

[KS99] Aviad Kipnis and Adi Shamir. Cryptanalysis of the HFE Public Key Cryptosystem by Relinearization. In *Crypto*, 1999.

[Lam79] Leslie Lamport. Constructing digital signatures from a one way function. 1979.

[Laz83] Daniel Lazard. Gröbner bases, Gaussian elimination and resolution of systems of algebraic equations. In *European Conference on Computer Algebra*, 1983.

[Loi06] Pierre Loidreau. Properties of codes in rank metric, 2006. URL: https://arxiv.org/abs/cs/0610057.

[Loi07] Pierre Loidreau. *Metrique rang et cryptographie*. Habilitation à Diriger des Recherches, Université Pierre et Marie Curie - Paris VI, 2007.

[Loi17] Pierre Loidreau. A new rank metric codes based encryption scheme. In *PQCrypto*, 2017.

[LT20a] Terry Shue Chien Lau and Chik How Tan. MURAVE: A New Rank Code-Based Signature with MUltiple RAnk VErification. In *Code-Based Cryptography Workshop*, 2020.

[LT20b] Terry Shue Chien Lau and Chik How Tan. Rank Preserving Code-based Signature. In *2020 IEEE International Symposium on Information Theory (ISIT)*, 2020.

[Lyu09] Vadim Lyubashevsky. Fiat-Shamir with aborts: Applications to lattice and factoring-based signatures. In *Asiacrypt*, 2009.

[McE78] Robert J McEliece. A Public-Key Cryptosystem Based On Algebraic Coding Theory. *Deep Space Network Progress Report*, 1978.

[Mer89] Ralph C Merkle. A certified digital signature. In *Crypto*, 1989.

[MI88] Tsutomu Matsumoto and Hideki Imai. Public quadratic polynomial-tuples for efficient signature-verification and message-encryption. In *Eurocrypt*, 1988.

[MM22] Luciano Maino and Chloe Martindale. An attack on SIDH with arbitrary starting curve. *Cryptology ePrint Archive*, 2022. URL: https://eprint.iacr.org/2022/1026.

[MTSB13] Rafael Misoczki, Jean-Pierre Tillich, Nicolas Sendrier, and Paulo S. L. M. Barreto. MDPC-McEliece: New McEliece variants from Moderate Density Parity-Check codes. In *2013 IEEE International Symposium on Information Theory (ISIT)*, 2013.

[Onn94] Shmuel Onn. Hilbert series of group representations and Gröbner bases for generic modules. *Journal of Algebraic Combinatorics*, 1994.

[Ore33] Oystein Ore. On a special class of polynomials. *Trans. Amer. Math. Soc.*, 1933.

[OTKN18] Ayoub Otmani, Hervé Talé-Kalachi, and Sélestin Ndjeya. Improved cryptanalysis of rank metric schemes based on Gabidulin codes. *Designs, Codes and Cryptography (DCC)*, 2018.

[Ove05] Raphael Overbeck. A New Structural Attack for GPT and Variants. In *Mycrypt*, 2005.

[Pat95] Jacques Patarin. Cryptanalysis of the Matsumoto and Imai public key scheme of Eurocrypt'88. In *Crypto*, 1995.

[Pat96] Jacques Patarin. Hidden fields equations (HFE) and isomorphisms of polynomials (IP): Two new families of asymmetric algorithms. In *Eurocrypt*, 1996.

[Pha21] Ba Duc Pham. *Étude et conception de nouvelles primitives de chiffrement fondées sur les codes correcteurs d'erreurs en métrique rang*. PhD thesis, Rennes 1, 2021.

[Ple98] Vera Pless. *Introduction to the theory of error-correcting codes*. John Wiley & Sons, 1998.

[PS96] David Pointcheval and Jacques Stern. Security proofs for signature schemes. In *Eurocrypt*, 1996.

[Reg05] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In *ACM STOC*, 2005.

[RS60] Irving S Reed and Gustave Solomon. Polynomial codes over certain finite fields. *Journal of the Society for Industrial and Applied Mathematics (SIAM)*, 1960.

[RS06] Alexander Rostovtsev and Anton Stolbunov. Public-key cryptosystem based on isogenies. *Cryptology ePrint Archive*, 2006.

[RSA78] Ronald L Rivest, Adi Shamir, and Leonard Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 1978.

[Sch91] Claus-Peter Schnorr. Efficient signature generation by smart cards. *Journal of Cryptology*, 1991.

[Sho99] Peter W Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM review*, 1999.

[Spa12] Pierre-Jean Spaenlehauer. *Solving multi-homogeneous and determinantal systems: algorithms, complexity, applications.* PhD thesis, Université Pierre et Marie Curie, 2012.

[Sto00] Arne Storjohann. *Algorithms for matrix canonical forms.* PhD thesis, ETH Zurich, 2000.

[VBC+19] Javier Verbel, John Baena, Daniel Cabarcas, Ray Perlner, and Daniel Smith-Tone. On the Complexity of "Superdetermined" Minrank Instances. In *PQCrypto*, 2019.

[Vil97] Gilles Villard. *A study of Coppersmith's block Wiedemann algorithm using matrix polynomials.* Citeseer, 1997.

[vzGG13] Joachim von zur Gathen and Jürgen Gerhard. *Modern computer algebra*. Cambridge university press, 2013.

[Wan19] Li-Ping Wang. Loong: a new IND-CCA-secure code-based KEM. In *2019 IEEE International Symposium on Information Theory (ISIT)*, 2019.

[Wie86] Douglas Wiedemann. Solving sparse linear equations over finite fields. *IEEE Transactions on Information Theory (IT)*, 1986.