"Ss. Cyril and Methodius" University in Skopje
FACULTY OF COMPUTER
SCIENCE AND ENGINEERING

Bachelor's Thesis

Error-Correcting Codes in the Rank Metric

With Applications to Cryptography

Dario Gjorgjevski
gjorgjevski.dario@students.finki.ukim.mk

January 24, 2018

Contents

List of Figures

List of Tables

List of Algorithms

List of Listings

Abstract

Public-key cryptography was invented by Diffie and Hellman in order to remove the need for trusted couriers when exchanging secret keys that will facilitate secure communication. The Diffie–Hellman key exchange (DHKE) and the Rivest–Shamir–Adleman (RSA) cryptosystem are two of the first public-key constructions that remain largely unbroken and in use to this day. However, the publication of Shor's algorithm in 1994 meant that these algorithms are rendered completely insecure given a quantum computer. Even though quantum computing is still far beyond reach, cryptographers are pushing for so-called post-quantum algorithms, i.e., algorithms which are believed to be secure even against attacks carried on quantum computers. Error-correcting codes are a popular way of building such algorithms and have been studied since roughly the same time as RSA. The McEliece cryptosystem is one such construction. It is based on the hardness of decoding random linear codes in the Hamming metric, using a scrambled Goppa code that only the holder of some secret information can decode efficiently. Unfortunately, the size of this "secret information" can get quite high in practice. Trying to alleviate this, Gabidulin, Paramonov, and Tretjakov proposed an analogous cryptosystem that utilizes error-correcting codes in the *rank metric*: the GPT cryptosystem. The goal of this thesis is twofold: first, to provide a survey of the GPT cryptosystem, Overbeck's structural attack on it, and the most recent reparation by Loidreau [23]; and second, to investigate the applicability of information set decoding in the rank metric. Information set decoding (ISD) is a family of algorithms that has been the most successful way to attack McEliece-like cryptosystems, but has not been explored in the context of the rank metric.

1. Introduction

In its advent, cryptography was used with the intention of making the communication between two parties secure in an ad-hoc manner. During the late 20th century, the picture of cryptography radically changed. A rich and rigorous theory was developed allowing for formal arguments to be made regarding the security of certain constructions. Furthermore, the field of cryptography now encompasses many well-defined objectives such as:

confidentiality – keeping information secret from all but those who are authorized to see it;

data integrity – ensuring information has not been altered by unauthorized or unknown means;

authentication – corroboration of the identity of an entity (e.g., a person, a computer terminal, a credit card, etc.); and

non-repudiation – preventing the denial of previous commitments or actions.

Secure communication can be established in two settings: symmetric and asymmetric. As we mentioned, secure communication is concerned with the *confidentiality* of messages: we want to make sure that a person intercepting a message that has been sent learns nothing (or "very little") about the message's content. To this end, *encryption* algorithms are employed. We will review both the symmetric and asymmetric settings, but the focus of the remainder of this thesis is going to be entirely on algorithms for confidentiality in the asymmetric setting.

The Symmetric Setting In the symmetric setting, two parties share some secret information in advance — the *symmetric key* (or just *key*) — that they use whenever they wish to communicate secretly with each other. The party sending a message uses the key to *encrypt* the message, and the receiver uses the same key to *decrypt* it upon receipt. Figure 1.1 depicts typical communication in this setting.

The Asymmetric Setting The asymmetric setting aims to solve the biggest problem of symmetric encryption: the need to securely exchange a key beforehand. Indeed, an observant person will ask: if two parties using symmetric encryption could exchange a key without anyone knowing, why do they not use the same means to exchange messages? Historically, such key exchanges had been done by "trusted couriers", but in a

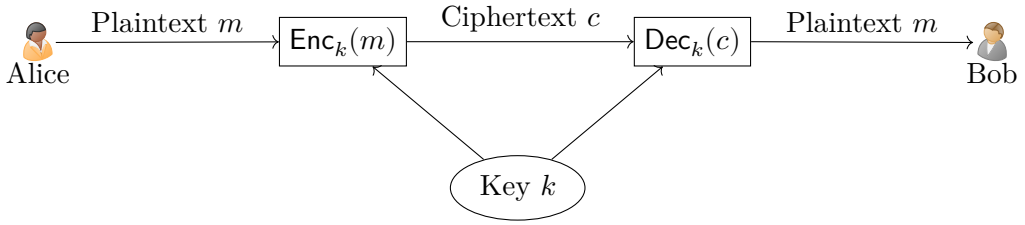

Figure 1.1.: Communication in the symmetric setting

network such as the Internet this is anything but a feasible solution. Cryptography in the asymmetric setting is called *public-key cryptography*. Public-key cryptography eliminates the need for trusted couriers by having users use pairs of keys: *public keys* which may be disseminated widely, and *private keys* which are known only to their owners.

Diffie and Hellman [5] invented public-key cryptography by publishing the so-called *Diffie–Hellman key exchange* (DHKE). DHKE allows two parties to *generate* a shared secret in such a way that the secret cannot be deduced by observing the communication. A few years later, Rivest, Shamir, and Adleman [30] published the RSA cryptosystem which can be used to encrypt and decrypt messages. It remains unbroken to this day and is one of the most widely used cryptosystems. In a public-key cryptosystem such as RSA, any person can encrypt a message using the public key of the intended receiver; but, once encrypted, the ciphertext can be decrypted only with the receiver's private key.

> Putting mail in the mailbox is analogous to encrypting with the public key; anyone can do it. Just open the slot and drop it in. Getting mail out of a mailbox is analogous to decrypting with the private key. Generally it's hard; you need welding torches. However, if you have the secret (the physical key to the mailbox), it's easy to get mail out of a mailbox. [31]

Public-key cryptography revolves largely around the notion of *one-way* and *trapdoor* functions. Informally, a one-way function is a function that is easy to compute on every input, but hard to invert given the image of a random input; on the other hand, a trapdoor function is a one-way function that can be inverted given some special information, called the "trapdoor". The RSA cryptosystem relies on the hardness of integer factorization to construct a trapdoor permutation for encryption. The only computationally feasible way to invert it is by knowing the trapdoor – the prime factors. However, Shor [33] formulated an algorithm which can factor integers efficiently and thus "break" RSA. There is only one drawback: to run it, one needs a quantum computer. The discovery of Shor's algorithm led researchers to consider *post-quantum algorithms*: public-key algorithms thought to be secure against attacks by quantum computers.

Interestingly, a cryptosystem from roughly the same era as DHKE and RSA is believed to be post-quantum: the *McEliece cryptosystem*, first formulated by McEliece [24]. The security of McEliece relies on the hardness of decoding a random linear code. Namely, a user publishes a scrambled Goppa code as the public key and keeps the "unscrambled"

code in private. Senders simply encode the message with the public code and add some errors. Attackers are faced with decoding the scrambled code which has no obvious structure, while the intended recipient has all the necessary information to unscramble and use (one of the) efficient decoding algorithms for Goppa codes.

The best generic attacks against the McEliece cryptosystem come from a family of algorithms known as *information set decoding* [2]. The main disadvantage of McEliece is its key size: to achieve 128-bit security against information set decoding, a key of around 192 kB is required. Gabidulin, Paramonov, and Tretjakov [10] proposed the GPT cryptosystem which uses error-correcting codes in *rank metric*. (The Goppa codes used in McEliece correct errors in the *Hamming metric*.) GPT is thought to require considerably smaller keys for equivalent security. The goal of this thesis is to provide a survey of GPT and formulate the information set decoding family of algorithms in the rank metric.

2. The GPT Cryptosystem

2.1. Foundations

Code-based cryptography is concerned with the use of error-correcting codes for cryptographic purposes. Much of the motivation and background for the theory comes from the landmark work of Shannon [32]. In it, he laid the foundations of *information theory*, which deals with the transmission of messages over noisy channels. A typical model of information transmission is given in fig. 2.1.

We assume that a sequence of symbols from a finite alphabet — the *message* — is transmitted over a noisy channel. (The noise is usually assumed to be additive.) Each symbol of the message has a probability to be affected by error; in a sense that instead of the symbol `a`, the receiver sees the symbol `b`. To overcome this problem, the transmitted information will not only contain the message, but will also include redundancy based on the message's contents.

Specifically, each message is mapped injectively into a *codeword* through a process called *encoding*. Instead of the message, the codeword is transmitted over the noisy channel. The receiver then uses a *decoding* algorithm to detect and possibly correct any errors that might have occurred during the transmission. In cryptography, the transmission is assumed to be error-free, but noise is added deliberately to the codeword as part of the encryption process.

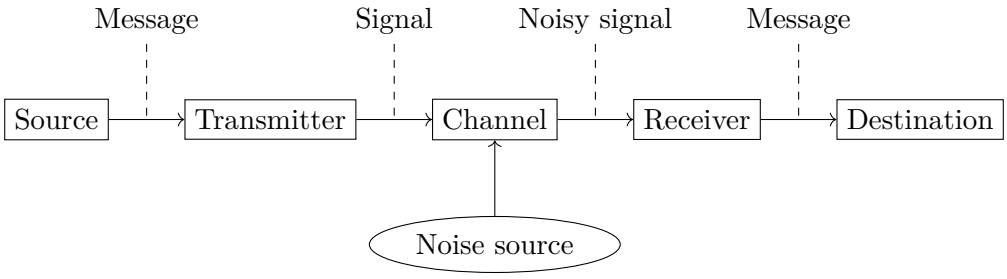

Figure 2.1.: Transmitting information over a noisy channel

2.2. Linear Codes

Section 2.2.1 defines linear codes and information sets, while section 2.2.2 provides an overview and briefly discusses the complexity of solving some computational problems related to the decoding of random linear codes.

2.2.1. Basic Concepts

Definition 1 (Linear code)**.** An $[n, k]$-code $\mathcal{C}$ over a finite field $\mathbb{F}$ is a k-dimensional subspace of the vector space $\mathbb{F}^n$. $\mathcal{C}$ is an $[n, k, d]$-code if $d = \min_{\boldsymbol{x},\boldsymbol{y}\in\mathcal{C}} \|\boldsymbol{x} - \boldsymbol{y}\|$ for some norm $\|\cdot\|$. The elements of $\mathcal{C}$ are called *codewords*. d is called the *minimum distance* of the code with respect to $\|\cdot\|$.

Definition 2 (Generator matrix)**.** A full-rank matrix $\boldsymbol{G} \in \mathbb{F}^{k\times n}$ is said to be a *generator matrix* for the $[n, k]$-code $\mathcal{C}$ if its rows span $\mathcal{C}$ over $\mathbb{F}$. In other words,

$$\mathcal{C} = \{\boldsymbol{x}\boldsymbol{G} : \boldsymbol{x} \in \mathbb{F}^k\}.$$

$\boldsymbol{G}$ defines an *encoding map* $f_{\boldsymbol{G}} : \mathbb{F}^k \to \mathbb{F}^n$ given by $\boldsymbol{x} \mapsto \boldsymbol{x}\boldsymbol{G}$.

Before defining the parity-check matrix, we introduce the dual of a code.

Definition 3 (Dual)**.** Let $\mathcal{C}$ be an $[n, k]$-code. The *dual* of $\mathcal{C}$ is the $[n, n-k]$-code $\mathcal{C}^\perp$ defined by:

$$\mathcal{C}^\perp := \{\boldsymbol{y} \in \mathbb{F}^n : \forall \boldsymbol{x} \in \mathcal{C}.\, \boldsymbol{x}\boldsymbol{y}^\top = 0\}.$$

Definition 4 (Parity-check matrix)**.** A *parity-check* matrix of a code $\mathcal{C}$ is a generator matrix of its dual, $\mathcal{C}^\perp$.

Let $\boldsymbol{G}$ be a generator matrix of an $[n, k]$-code $\mathcal{C}$ and $\boldsymbol{H}$ be a parity-check matrix of $\mathcal{C}$. Then $\boldsymbol{G}\boldsymbol{H}^\top = \boldsymbol{0}$. Conversely, any $\boldsymbol{H} \in \mathbb{F}^{(n-k)\times n}$ that satisfies $\boldsymbol{G}\boldsymbol{H}^\top = \boldsymbol{0}$ is a parity-check matrix of $\mathcal{C}$.

A code $\mathcal{C}$ has many generator matrix representations. $\boldsymbol{G}$ is said to be in *systematic form* if its first k columns form the $k \times k$ identity matrix, $\boldsymbol{I}_k$. For any systematic generator matrix $\boldsymbol{G}_{\mathsf{sys}}$, each entry of the message vector appears among the entries of the codeword. Given the generator matrix in systematic form $\boldsymbol{G}_{\mathsf{sys}} = \begin{bmatrix}\boldsymbol{I}_k & \boldsymbol{Q}\end{bmatrix}$, the parity-check matrix in *canonical form* is $\boldsymbol{H} = \begin{bmatrix}-\boldsymbol{Q}^\top & \boldsymbol{I}_{n-k}\end{bmatrix}$.

Example. Consider the matrix $\boldsymbol{G} \in \mathbb{F}_2^{4\times 7}$:

$$\boldsymbol{G} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 \end{bmatrix}.$$

We have that $\operatorname{rank} \boldsymbol{G} = 4$, so $\boldsymbol{G}$ generates a $[7, 4]$-code over $\mathbb{F}_2$. Let $\boldsymbol{H}$ be given as

$$\boldsymbol{H} = \begin{bmatrix} 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

We have $\boldsymbol{G}\boldsymbol{H}^\top = \boldsymbol{0}$ and $\operatorname{rank} \boldsymbol{H} = 3$, so $\boldsymbol{H}$ is a parity-check matrix of the code.

Definition 5 (Syndrome)**.** Let $\boldsymbol{H}$ be a parity-check matrix for $\mathcal{C}$. The *syndrome* of a vector $\boldsymbol{y} \in \mathbb{F}^n$ with respect to $\boldsymbol{H}$ is the (column) vector $\boldsymbol{H}\boldsymbol{y}^\top \in \mathbb{F}^{n-k}$.

Definition 6 (Information set)**.** An *information set* of an $[n, k]$-code $\mathcal{C}$ with generator matrix $\boldsymbol{G}$ is a size-k subset $\mathcal{I} \subset \{1, \dots, n\}$ such that the columns of $\boldsymbol{G}$ indexed by $\mathcal{I}$ form an invertible $k \times k$ submatrix, $\boldsymbol{G}_\mathcal{I}$. In particular, $\boldsymbol{G}_\mathcal{I}^{-1}\boldsymbol{G}$ is a generator matrix of $\mathcal{C}$ in systematic form (up to a permutation of columns). If $\boldsymbol{H}$ is a parity-check matrix, then the columns indexed by $\{1, \dots, n\} \setminus \mathcal{I}$ form an invertible $(n-k) \times (n-k)$ matrix.

Let $\boldsymbol{c} \coloneqq \boldsymbol{x}\boldsymbol{G}_\mathcal{I}^{-1}\boldsymbol{G}$ for some vector $\boldsymbol{x} \in \mathbb{F}^k$. The $\mathcal{I}$-indexed entries of $\boldsymbol{c}$ form the information vector $\boldsymbol{x}$. These entries are called *information symbols*; the rest of the entries are called *parity-check symbols*.

2.2.2. Computational Problems

The decoding of random codes lends itself to many computational problems. The *general decoding problem* can be stated as follows: given an encoding function $f\colon \mathbb{F}^k \to \mathbb{F}^n$ and an encoded message $f(\boldsymbol{x})$ that is inflicted with random noise, determine the original message $\boldsymbol{x}$. Here, f need not be linear and the noise need not be additive.

However, for practical reasons, we restrict ourselves only to the linear setting, in which the encoding encoding function is specified by the generator matrix $\boldsymbol{G}$ and generates a codeword

$$\boldsymbol{c} \coloneqq f_{\boldsymbol{G}}(\boldsymbol{x}) = \boldsymbol{x}\boldsymbol{G}.$$

The codeword is inflicted with additive noise and received as $\boldsymbol{y} \coloneqq \boldsymbol{c} + \boldsymbol{e}$, where $\boldsymbol{e} \in \mathbb{F}^n$ denotes the noise.

Definition 7 (Minimum Distance Decoding)**.** Let $\mathcal{C}$ be an $[n, k, d]$-code. Given a received word $\boldsymbol{y}$ and $w \in \mathbb{N}$, find a codeword $\boldsymbol{c} \in \mathcal{C}$ such that $\|\boldsymbol{y} - \boldsymbol{c}\| \leq w$. If no such codeword exists, output $\perp$. We denote the set of all solutions to this problem by $\mathsf{MDD}(\mathcal{C}, \boldsymbol{y}, w)$.

When $\|\cdot\|$ is the Hamming norm, the decision version of Minimum Distance Decoding was proven $\mathcal{NP}$-complete for binary codes by Berlekamp, McEliece, and Tilborg [3] and in the general case by Barg [1]. Minimum Distance Decoding can also be formulated as a problem of syndrome decoding.

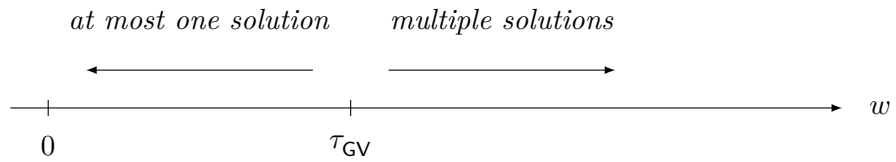

Figure 2.2.: Number of solutions in $\mathsf{CSD}(\boldsymbol{H}, \boldsymbol{s}, w)$

Definition 8 (Computational Syndrome Decoding)**.** Let $\mathcal{C}$ be an $[n, k, d]$-code with parity-check matrix $\boldsymbol{H}$. Given a syndrome $\boldsymbol{s} \in \mathbb{F}^{n-k}$ and $w \in \mathbb{N}$, find a word $\boldsymbol{e} \in \mathbb{F}^n$ such that $\boldsymbol{H}\boldsymbol{e}^\top = \boldsymbol{s}$ and $\|\boldsymbol{e}\| \leq w$. If no such word exists, output $\perp$. We denote the set of all solutions to this problem by $\mathsf{CSD}(\boldsymbol{H}, \boldsymbol{s}, w)$.

If we fix n and k, the number of solutions in $\mathsf{CSD}(\boldsymbol{H}, \boldsymbol{s}, w)$ is illustrated in fig. 2.2. τ_{GV} is the *Gilbert–Varshamov radius*: the maximum radius of spheres centered at codewords of $\mathcal{C}$ so that they together fit in $\mathbb{F}^n$. For the purposes of cryptanalysis, we will restrict ourselves to the case when $\mathsf{CSD}(\boldsymbol{H}, \boldsymbol{s}, w) \neq \emptyset$ and $w < \tau_{\mathsf{GV}}$; i.e., the case with *exactly one* solution. Closely related is the problem of finding a small-weight codeword $\boldsymbol{c} \in \mathcal{C}$.

Definition 9 (Subspace Weights)**.** Let $\mathcal{C}$ be an $[n, k, d]$-code. Given $w \in \mathbb{N}$, find a codeword $\boldsymbol{c} \in \mathcal{C}$ such that $\|\boldsymbol{c}\| \leq w$. If no such codeword exists, output $\perp$. We denote the set of all solutions to this problem by $\mathsf{SW}(\mathcal{C}, w)$.

To see the importance of Subspace Weights, let $\boldsymbol{y} = \boldsymbol{c} + \boldsymbol{e}$ be a received word, and consider $\mathcal{C}'$, the code generated by

$$\boldsymbol{G}' := \begin{bmatrix} \boldsymbol{G} \\ \boldsymbol{y} \end{bmatrix}.$$

In order for $\boldsymbol{c}$ to be uniquely decodable, $\|\boldsymbol{e}\| \leq t < \lfloor (d-1)/2 \rfloor$ must hold, where t is the error-correcting capacity of $\mathcal{C}$. Hence

$$\mathcal{C}' = \mathcal{C} \cup \{\boldsymbol{y} + \mathcal{C}\}$$

has minimum distance $d' = \|\boldsymbol{e}\|$ and therefore the codeword of minimum weight is $\boldsymbol{e}$. Solving $\mathsf{SW}(\mathcal{C}', w')$ for $w' \in \{0, \ldots, t\}$ will let us find $\boldsymbol{e}$.

2.3. The McEliece Cryptosystem

This section describes the McEliece cryptosystem: its parameters, key generation, and encryption and decryption procedures.

System parameters Choose $k, n \in \mathbb{N}$ such that $k < n$; as usual, n is the length and k is the dimension of the code. Choose $t \in \mathbb{N}$ to be the desired error-correcting capability. Recall that Γ corrects errors in the Hamming metric.

Key generation Generate a random classical Goppa code $\Gamma = \Gamma_q(a_1, \ldots, a_n; g)$ of dimension k over $\mathbb{F}_q$ with an error-correcting capability of w. Choose a random $n \times n$ permutation matrix $\boldsymbol{P}$ and an invertible $k \times k$ matrix $\boldsymbol{S}$.

Public key Tuple $(\hat{\boldsymbol{G}}, n, k, w)$, where $\hat{\boldsymbol{G}} := \boldsymbol{SGP}$. $\boldsymbol{G}$ is a generator matrix of Γ.

Private key Tuple $(\Gamma, \boldsymbol{G}, \boldsymbol{P}, \boldsymbol{S})$.

Information needs to be embedded in a length-k word $\boldsymbol{x} \in \mathbb{F}_q^k$ in order to be suitable for the encryption algorithm. Then, $\boldsymbol{x}$ can be encrypted with algorithm 2.1. The legitimate owner of the private key can use algorithm 2.2 to decode a ciphertext $\boldsymbol{y}$.

Algorithm 2.1: McEliece encryption

Input: Message $\boldsymbol{x} \in \mathbb{F}_q^k$, public generator matrix $\hat{\boldsymbol{G}} \in \mathbb{F}_q^{k \times n}$, and $w \in \mathbb{N}$
Output: Ciphertext $\boldsymbol{y} \in \mathbb{F}_q^n$
begin
  $\boldsymbol{c} \leftarrow \boldsymbol{x}\hat{\boldsymbol{G}}$
  $\boldsymbol{e} \leftarrow_{\$} \mathbb{F}_q^n$ with $\|\boldsymbol{e}\|_{\mathrm{H}} = w$
  return $\boldsymbol{y} = \boldsymbol{c} + \boldsymbol{e}$
end

Algorithm 2.2: McEliece decryption

Input: Ciphertext $\boldsymbol{y} \in \mathbb{F}_q^n$ and private key $(\Gamma, \boldsymbol{G}, \boldsymbol{P}, \boldsymbol{S})$
Output: Message $\boldsymbol{x} \in \mathbb{F}_q^k$
begin
  $\boldsymbol{yP}^{-1} = \boldsymbol{xSG} + \boldsymbol{eP}^{-1}$ `/* note that` $\|\boldsymbol{eP}^{-1}\|_{\mathrm{H}} = w$ `*/`
  $\boldsymbol{x} \leftarrow$ decode $\boldsymbol{yP}^{-1}$ using an efficient decoding algorithm for Γ
  return $\boldsymbol{x}$
end

2.4. Gabidulin Codes

Gabidulin codes [8] correct errors in the rank metric (also called pattern errors) efficiently, which in general is harder than correcting errors in the Hamming metric. Gabidulin,

Paramonov, and Tretjakov [10] proposed a cryptosystem based on Gabidulin codes, known as GPT. Because of its better resistance against general decoding attacks, smaller key sizes were proposed for GPT compared to the McEliece cryptosystem. In this section, we define Gabidulin codes and give a formal definition of the GPT cryptosystem.

2.4.1. The Rank Metric

Codes in the rank metric (e.g., Gabidulin codes) are linear codes over the finite field $\mathbb{F}_{q^m}$ for q (power of a) prime and $m \in \mathbb{N}$. As suggested by their name, they use a special concept of distance.

Definition 10 (Rank norm)**.** Let $\boldsymbol{x} := \begin{bmatrix} \boldsymbol{x}_1 & \cdots & \boldsymbol{x}_n \end{bmatrix} \in \mathbb{F}_{q^m}^n$ and $\{\beta_1, \ldots, \beta_m\}$ be a basis of $\mathbb{F}_{q^m}$ over $\mathbb{F}_q$. We can write

$$x_i = \sum_{j=1}^{m} x_{i,j}\beta_j \tag{2.1}$$

for all $i = 1, \ldots, n$; with $x_{i,j} \in \mathbb{F}_q$. The *rank norm* $\|\cdot\|_q$ is defined as

$$\|\boldsymbol{x}\|_q := \operatorname{rank}\left(\begin{bmatrix} x_{1,1} & \cdots & x_{1,m} \\ \vdots & \ddots & \vdots \\ x_{n,1} & \cdots & x_{n,m} \end{bmatrix}\right),$$

with the $x_{i,j}$'s as given by eq. (2.1).

The rank norm of $\boldsymbol{x} \in \mathbb{F}_{q^m}^n$ is independent of the choice of basis and induces a metric called the *rank metric* defined in definition 11. Note that every $\boldsymbol{T} \in \mathsf{GL}_n(\mathbb{F}_q)$ is an isometry for the rank metric: $\|\boldsymbol{x}\boldsymbol{T}\|_q = \|\boldsymbol{x}\|_q$.

Definition 11 (Rank metric)**.** Let $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{F}_{q^m}^n$. The function $\mathrm{d}_{\mathrm{R}}(\boldsymbol{x}, \boldsymbol{y}) := \|\boldsymbol{x} - \boldsymbol{y}\|_q$ is a metric over $\mathbb{F}_{q^m}^n$, called the *rank metric* (or *rank distance*).

Lemma 1. *Let $\|\cdot\|_{\mathrm{H}}$ denote the Hamming norm. Then, for all $\boldsymbol{x} \in \mathbb{F}_{q^m}^n$,*

$$\|\boldsymbol{x}\|_q \leq \|\boldsymbol{x}\|_{\mathrm{H}}.$$

Proof. Let $\|\boldsymbol{x}\|_{\mathrm{H}} =: k$. By definition, $\boldsymbol{x}$ has k nonzero entries. Thus, the remaining $n - k$ entries are all zero and do not influence $\|\boldsymbol{x}\|_q$ whatsoever. The result immediately follows. Equality holds if and only if all k nonzero entries are linearly independent over $\mathbb{F}_q$. □

Lemma 2 (Singleton bound)**.** *Let $\mathcal{C}$ be an $[n, k, d]$-code over $\mathbb{F}_q$ in the Hamming metric. Then, the* Singleton bound *states that*

$$q^k \leq q^{n-d+1} \implies k \leq n - d + 1,$$

which is usually written as

$$d \leq n - k + 1.$$

Proof. See [34]. □

Due to the Singleton bound in the Hamming metric, the minimum rank distance d of an $[n, k, d]$-code over $\mathbb{F}_{q^m}$ satisfies

$$d \leq n - k + 1. \tag{2.2}$$

An alternative bound — tighter than that in eq. (2.2) when $n \leq m$ — is given in [8]:

$$d \leq \frac{m}{n}(n - k) + 1.$$

Combining the two bounds gives

$$d \leq \min\{m, n\} - \max\{m, n\}k + 1.$$

2.4.2. Generating Gabidulin Codes

Codes which achieve the Singleton bound in the rank metric with equality are called Maximum Rank Distance (MRD) codes. Gabidulin [8] constructed a family of MRD codes over $\mathbb{F}_{q^m}$ of length $n \leq m$.

For any $x \in \mathbb{F}_{q^m}$ and any $i \in \mathbb{Z}$, the quantity x^{q^i} is denoted by $x^{[i]}$. The notation is extended to vectors and matrices in a component-wise manner. Lemma 3 shows some useful properties of the operation.

Lemma 3. *For any $\boldsymbol{A} \in \mathbb{F}_{q^m}^{l \times s}$ and $\boldsymbol{B} \in \mathbb{F}_{q^m}^{k \times n}$, and for any $\alpha, \beta \in \mathbb{F}_q$:*

1. *If $l = k$ and $s = n$, then*

$$(\alpha \boldsymbol{A} + \beta \boldsymbol{B})^{[i]} = \alpha \boldsymbol{A}^{[i]} + \beta \boldsymbol{B}^{[i]}.$$

2. *If $s = k$ then*

$$(\boldsymbol{A}\boldsymbol{B})^{[i]} = \boldsymbol{A}^{[i]}\boldsymbol{B}^{[i]}.$$

In particular, if $\boldsymbol{S} \in \mathsf{GL}_n(\mathbb{F}_{q^m})$, then $\boldsymbol{S}^{[i]} \in \mathsf{GL}_n(\mathbb{F}_{q^m})$ and

$$(\boldsymbol{S}^{[i]})^{-1} = (\boldsymbol{S}^{-1})^{[i]}.$$

Proof. The proof of the two points comes directly from the properties of the Frobenius operators: multiplicative and $\mathbb{F}_q$-linear. For the last point, note that since $\boldsymbol{S}\boldsymbol{S}^{-1} = \boldsymbol{I}_n$, it is also true that $\boldsymbol{S}^{[i]}(\boldsymbol{S}^{-1})^{[i]} = \boldsymbol{I}_n$. This implies that $\boldsymbol{S}^{[i]} \in \mathsf{GL}_n(\mathbb{F}_{q^m})$ and hence $(\boldsymbol{S}^{[i]})^{-1} = (\boldsymbol{S}^{-1})^{[i]}$. □

We are now ready to introduce the Gabidulin codes.

Definition 12 (Gabidulin code)**.** Let $\boldsymbol{g} := \begin{bmatrix} g_1 & \cdots & g_n \end{bmatrix} \in \mathbb{F}_{q^m}^n$ with all g_i independent over the base field. (This implies $n \leq m$.) The code spanned by

$$\boldsymbol{G} := \begin{bmatrix} g_1 & \cdots & g_n \\ g_1^{[1]} & \cdots & g_n^{[1]} \\ \vdots & \ddots & \vdots \\ g_1^{[k-1]} & \cdots & g_n^{[k-1]} \end{bmatrix} \tag{2.3}$$

is called a *Gabidulin code* with dimension k. The vector $\boldsymbol{g}$ is called a *generator vector* of the Gabidulin code. We denote this code $\mathfrak{G}_k(\boldsymbol{g})$. A matrix of the form eq. (2.3) is called a *q-Vandermonde matrix*.

Remark. The vector $\boldsymbol{g}$ is not unique; all vectors of the form $\gamma\boldsymbol{g}$ with $\gamma \in \mathbb{F}_{q^m} \setminus \{0\}$ generate the same code.

Theorem 1. *Gabidulin codes are Maximum Rank Distance (MRD) codes.*

Proof. It is sufficient to show that for any $\boldsymbol{X} \in \mathbb{F}_q^{k\times n}$ with $\operatorname{rank} \boldsymbol{X} = k$, $\operatorname{rank}(\boldsymbol{G}\boldsymbol{X}^\top) = k$ as well. Note that

$$\boldsymbol{G}\boldsymbol{X}^\top = \begin{bmatrix} f_1 & \cdots & f_n \\ f_1^{[1]} & \cdots & f_n^{[1]} \\ \vdots & \ddots & \vdots \\ f_1^{[k-1]} & \cdots & f_n^{[k-1]} \end{bmatrix}$$

with

$$\begin{bmatrix} f_1 & \cdots & f_k \end{bmatrix} = \begin{bmatrix} g_1 & \cdots & g_k \end{bmatrix} \boldsymbol{X}^\top.$$

Since $\|g\|_q = n$, $\|f\|_q = \min\{n, \operatorname{rank} \boldsymbol{X}\} = k$. From here, $\operatorname{rank}(\boldsymbol{G}\boldsymbol{X}^\top) = k$. □

An $[n,k]$-Gabidulin code has minimum distance $d = n - k + 1$ and corrects errors of rank up to $t := \lfloor (n-k)/2 \rfloor$. For all $\boldsymbol{T} \in \mathrm{GL}_n(\mathbb{F}_q)$, $\boldsymbol{G}\boldsymbol{T}$ is a generator matrix of the Gabidulin code with generator vector $\boldsymbol{g}\boldsymbol{T}$.

Lemma 4. *Let $\mathfrak{G}_k(\boldsymbol{g})$ be an $[n,k]$-Gabidulin code over $\mathbb{F}_{q^m}$ with generator vector $\boldsymbol{g}$. The dual of $\mathfrak{G}_k(\boldsymbol{g})$ is the Gabidulin code $\mathfrak{G}_{n-k}(\boldsymbol{h})$, where $\boldsymbol{h} = \boldsymbol{y}^{[-(n-k-1)]}$ for some $\boldsymbol{y} \in \mathfrak{G}_{n-1}(\boldsymbol{g})^\perp$.*

Proof. Remark from eq. (2.3) that

$$\mathfrak{G}_{n-1}(\boldsymbol{g})^\perp \subset \mathfrak{G}_{n-2}(\boldsymbol{g})^\perp \subset \cdots \subset \mathfrak{G}_k(\boldsymbol{g})^\perp.$$

Since $\mathfrak{G}_{n-1}(\boldsymbol{g})^\perp$ is an MRD code of dimension 1, its minimum distance is $d = n$. Thus, all of its nonzero elements are of rank n. Let $\boldsymbol{y} \in \mathfrak{G}_{n-1}(\boldsymbol{g})^\perp$ with $\boldsymbol{y} \neq \boldsymbol{0}$. We have

$$\forall i \in \{0, \ldots, n-2\}. \sum_{j=1}^{n} y_j g_j^{[i]} = 0.$$

This implies that

$$\forall i \in \{0, \dots, n-2\}. \sum_{j=1}^{n} y_j^{[-1]} g_j^{[i-1]} = 0,$$

and in particular,

$$\forall i \in \{0, \dots, n-3\}. \sum_{j=1}^{n} y_j^{[-1]} g_j^{[i-1]} = 0.$$

Thus $\boldsymbol{y}^{[-1]} \in \mathfrak{G}_{n-2}(\boldsymbol{g})^{\perp}$. We can argue by induction that

$$\forall u \in \{0, \dots, n-1\}. \boldsymbol{y}^{[-u]} \in \mathfrak{G}_{n-1-u}(\boldsymbol{g})^{\perp},$$

and for a given $u \in \{0, \dots, n-1\}$ we have

$$\forall i \in \{0, \dots, u\}. \boldsymbol{y}^{[-u+i]} \in \mathfrak{G}_{n-1-u+i}(\boldsymbol{g})^{\perp} \subset \mathfrak{G}_{n-1-u}(\boldsymbol{g})^{\perp}.$$

Notably, for $u = n-k-1$ and $\boldsymbol{h} := \boldsymbol{y}^{[-u]}$, we have $\boldsymbol{h}^{[i]} \in \mathfrak{G}_k(\boldsymbol{g})^{\perp}$ for all $i \in \{0, \dots, n-k-1\}$. In other words,

$$\mathfrak{G}_k(\boldsymbol{g})^{\perp} = \mathfrak{G}_{n-k}(\boldsymbol{g}).$$

□

Corollary 1. *The parity-check matrix of $\mathfrak{G}_k(\boldsymbol{g})$ can be written as*

$$\boldsymbol{H} = \begin{bmatrix} h_1 & \cdots & h_n \\ h_1^{[1]} & \cdots & h_n^{[1]} \\ \vdots & \ddots & \vdots \\ h_1^{[d-2]} & \cdots & h_n^{[d-2]} \end{bmatrix},$$

for an appropriate choice of $\boldsymbol{h}$. (Recall that in the case of Gabidulin codes, $d = n-k+1$.)

Example. Let $\mathbb{F}_2^2 = \mathbb{F}_2[\alpha]$ and

$$\boldsymbol{G} = \begin{bmatrix} 1 & \alpha \end{bmatrix}.$$

Then the code generated by $\boldsymbol{G}$ is

$$\begin{aligned} \mathcal{C} &= \left\{ \begin{bmatrix} 0 & 0 \end{bmatrix}, \begin{bmatrix} 1 & \alpha \end{bmatrix}, \begin{bmatrix} \alpha & \alpha^2 \end{bmatrix}, \begin{bmatrix} \alpha^2 & 1 \end{bmatrix} \right\} \\ &\cong \left\{ \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}, \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix}, \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} \right\}. \end{aligned}$$

It has dimension 1 (over $\mathbb{F}_{2^2}$) and minimum distance 2 (over $\mathbb{F}_2$). A respective parity-check matrix is

$$\boldsymbol{H} = \begin{bmatrix} \alpha & 1 \end{bmatrix}.$$

2.5. Decoding Gabidulin Codes

Definition 13 (Linearized polynomial)**.** A *linearized polynomial* $F(z)$ over $\mathbb{F}_{q^m}$ is a polynomial of the form $F(z) = \sum_{i=0}^{u} f_i z^{[i]}$, where $f_i \in \mathbb{F}_{q^m}$ for all $0 \leq i \leq u$. We refer to u as the *degree* of $F(z)$, denoted $\deg F(z)$.

Note that for any $\alpha \in \mathbb{F}_{q^m}, \alpha^{[m]} = \alpha^{[0]}$, hence $u < m$. Linearized polynomials can be viewed as linear operators over $\mathbb{F}_{q^m}$, thus their roots form a linear subspace of $\mathbb{F}_{q^m}$ with dimension at most equal to their degrees. They form an algebra under addition and the symbolic product, defined as $L(z) \otimes M(z) \coloneqq L(M(z))$. Note that the symbolic product is not commutative, so we require both left- and right-long divisions for linearized polynomials. These divisions can be computed by the *extended Euclidean algorithm* for linearized polynomials [36].

Recall that, for an appropriate choice of $\boldsymbol{h}$, the parity-check matrix of a Gabidulin code can be written as

$$\boldsymbol{H} = \begin{bmatrix} h_1 & \cdots & h_n \\ h_1^{[1]} & \cdots & h_n^{[1]} \\ \vdots & \ddots & \vdots \\ h_1^{[d-2]} & \cdots & h_n^{[d-2]} \end{bmatrix}.$$

Suppose we receive a vector $\boldsymbol{y} \coloneqq \boldsymbol{c} + \boldsymbol{e}$ with $\|\boldsymbol{e}\|_q =: w \leq \lfloor (d-1)/2 \rfloor$. The objective is to determine $\boldsymbol{e}$. We denote $\boldsymbol{e} = \begin{bmatrix} E_1 & \cdots & E_w \end{bmatrix} \boldsymbol{Y}$, where the $E_j \in \mathbb{F}_{q^m}$ are linearly independent and $\boldsymbol{Y} \in \mathbb{F}_q^{w \times n}$ has full rank. The decoding of a Gabidulin code defined by the parity-check matrix $\boldsymbol{H}$ consists of six steps:

Step 1. Calculate the syndrome $\boldsymbol{s} = \boldsymbol{H}\boldsymbol{y}^\top =: \begin{bmatrix} s_1 & \cdots & s_{d-1} \end{bmatrix}^\top$.

Step 2. Determine $\Lambda(z)$ and $F(z)$ such that $\deg F(z) < w$ and $F(z) = \Lambda(z) \otimes S(z) \pmod{z^{[d-1]}}$ using the extended Euclidean algorithm.

Step 3. Determine w linearly independent roots $E_1, \ldots, E_w$ of $\Lambda(z)$.

Step 4. Determine $\boldsymbol{x} \coloneqq \begin{bmatrix} x_1 & \cdots & x_r \end{bmatrix}$ using $\sum_{j=1}^{r} E_j x_j^{[p-1]} = s_p$ for all $1 \leq p \leq w$.

Step 5. Determine $\boldsymbol{Y}$ using $x_p = \sum_{j=1}^{n} Y_{p,j} h_j$ for all $1 \leq p \leq w$.

Step 6. Calculate $\boldsymbol{e} = \begin{bmatrix} E_1 & \cdots & E_w \end{bmatrix} \boldsymbol{Y}$.

The complexity of steps 1 to 6 is $\mathcal{O}(d^3 + dn)$.

2.6. Description of the GPT Cryptosystem

In this section, we describe the details of the GPT cryptosystem. The original proposal by Gabidulin, Paramonov, and Tretjakov [10] was attacked successfully by Gibson [14, 15]. A reparation resisting the attack was proposed by Gabidulin and Ourivski [9]. This reparation was later attacked by Overbeck [25, 26, 27], which is the focus of chapter 3. We also show that more elaborate reparations can all be reduced to a common form.

2.6.1. Original Proposal by Gabidulin, Paramonov, and Tretjakov [10]

System parameters Choose $k, n, m \in \mathbb{N}$ such that $k < n \leq m$. Define the error-correcting capacity $t := \lfloor (n-k)/2 \rfloor$.

Key generation Pick $\boldsymbol{g} \in \mathbb{F}_{q^m}^n$ with $\|\boldsymbol{g}\|_q = n$, and define $\boldsymbol{G}$ as in eq. (2.3); i.e., as a generator matrix of $\mathfrak{G}_k(\boldsymbol{g})$. Define the *distortion* transformation

$$\mathcal{D} \colon \mathbb{F}_{q^m}^{k \times n} \to \mathbb{F}_{q^m}^{k \times n}$$

as

$$\mathcal{D}(\boldsymbol{G}) := \boldsymbol{S}(\boldsymbol{G} + \boldsymbol{X});$$

where $\boldsymbol{X} \in \mathbb{F}_{q^m}^{k \times n}$ is a random matrix of prescribed rank $t_{\boldsymbol{X}}$ and $\boldsymbol{S} \in \mathsf{GL}_k(\mathbb{F}_{q^m})$.

Public key Tuple $(\boldsymbol{G}_{\mathsf{pub}}, t_{\mathsf{pub}})$, where $\boldsymbol{G}_{\mathsf{pub}} := \mathcal{D}(\boldsymbol{G})$ and $t_{\mathsf{pub}} := t - t_{\boldsymbol{X}}$.

Private key Tuple $(\boldsymbol{G}, \boldsymbol{S})$.

Encryption To encrypt a message $\boldsymbol{x} \in \mathbb{F}_{q^m}^k$, we choose a random error $\boldsymbol{e} \in \mathbb{F}_{q^m}^n$ with $\|\boldsymbol{e}\|_q \leq t_{\mathsf{pub}}$ and compute the ciphertext

$$\boldsymbol{y} = \boldsymbol{x}\boldsymbol{G}_{\mathsf{pub}} + \boldsymbol{e}.$$

Decryption To decrypt a ciphertext $\boldsymbol{y}$, we simply apply the decoding procedure for Gabidulin codes (section 2.5) to it. The received word can be decoded since it is corrupted by $\boldsymbol{x}\boldsymbol{S}\boldsymbol{X} + \boldsymbol{e}$ whose rank is less than or equal to t since

$$\|\boldsymbol{x}\boldsymbol{S}\boldsymbol{X}\|_q \leq t_{\boldsymbol{X}}$$

and

$$\|\boldsymbol{x}\boldsymbol{S}\boldsymbol{X} + \boldsymbol{e}\|_q \leq \|\boldsymbol{x}\boldsymbol{S}\boldsymbol{X}\|_q + \|\boldsymbol{e}\|_q \leq t.$$

However, Gibson [14, 15] showed that the GPT cryptosystem as defined above is vulnerable to a polynomial-time key-recovery attack. Consequently, Gabidulin and Ourivski [9] proposed a reparation by considering a more general transformation combining the distortion matrix $\boldsymbol{X}$ with a column scrambler $\boldsymbol{P}$ over the base field.

2.6.2. Reparation by Gabidulin and Ourivski [9]

In order to mask the structure better, a more elaborate distortion transformation was proposed. Let $l \in \mathbb{N}$ with $l \ll n$. The distortion transformation is now of the form

$$\mathcal{D}(\boldsymbol{G}) := \boldsymbol{S}\begin{bmatrix}\boldsymbol{X}_1 & \boldsymbol{G}+\boldsymbol{X}_2\end{bmatrix}\boldsymbol{P}, \tag{2.4}$$

where $\boldsymbol{X}_1 \in \mathbb{F}_{q^m}^{k\times l}$, $\boldsymbol{X}_2 \in \mathbb{F}_{q^m}^{k\times n}$ with $\operatorname{rank}\boldsymbol{X}_2 < t$, and $\boldsymbol{P} \in \mathsf{GL}_{n+l}(\mathbb{F}_q)$. The public generator matrix is again $\boldsymbol{G}_{\mathsf{pub}} := \mathcal{D}(\boldsymbol{G})$; while the public rank parameter is $t_{\mathsf{pub}} = t - t_{\boldsymbol{X}_2}$, where $t_{\boldsymbol{X}_2} := \operatorname{rank}\boldsymbol{X}_2$.

The decryption process computes

$$\boldsymbol{P}^{-1} := \begin{bmatrix}\boldsymbol{Q}_1 & \boldsymbol{Q}_2\end{bmatrix},$$

where $\boldsymbol{Q}_1 \in \mathbb{F}_q^{(n+l)\times l}$, and $\boldsymbol{Q}_2 \in \mathbb{F}_q^{(l+n)\times n}$. The last n components of $\boldsymbol{y}\boldsymbol{P}^{-1}$ form the vector

$$\boldsymbol{x}\boldsymbol{S}\boldsymbol{G} + \boldsymbol{x}\boldsymbol{S}\boldsymbol{X}_2 + \boldsymbol{e}\boldsymbol{Q}_2.$$

Now, since $\|\boldsymbol{e}\boldsymbol{Q}_2\|_q \le \|\boldsymbol{e}\|_q$ and $\|\boldsymbol{x}\boldsymbol{S}\boldsymbol{X}_2\|_q \le \|\boldsymbol{X}_2\|_q$, it follows that

$$\|\boldsymbol{x}\boldsymbol{S}\boldsymbol{X}_2 + \boldsymbol{e}\boldsymbol{Q}_2\|_q \le t.$$

In other words, applying the decoding procedure for Gabidulin codes to the last n components of $\boldsymbol{y}\boldsymbol{P}^{-1}$ allows the holder of the private key to compute $\boldsymbol{x}\boldsymbol{S}$, and thus $\boldsymbol{x}$.

$\boldsymbol{X} := \begin{bmatrix}\boldsymbol{X}_1 & \boldsymbol{X}_2\end{bmatrix} \in \mathbb{F}_{q^m}^{k\times(l+n)}$ — called the *distortion matrix* — is needed to mask the structure of $\boldsymbol{G}$. Intuitively, if the value of $t_{\boldsymbol{X}_2}$ — the *distortion parameter* — is higher, the work factor of any structural attack increases. However, a higher value for $t_{\boldsymbol{X}_2}$ implies a smaller rank of the error, making generic decoding more feasible. Gadouleau and Yan [13] outlined a procedure for choosing the optimal distortion parameter.

2.6.3. Common Form of the Public Key

We show that a distortion caused by $\boldsymbol{X} := \begin{bmatrix}\boldsymbol{X}_1 & \boldsymbol{X}_2\end{bmatrix}$ with $\operatorname{rank}\boldsymbol{X}_2 = t_{\boldsymbol{X}_2}$ can be concentrated in just $t_{\boldsymbol{X}_2}$ columns [19].

Theorem 2. *Let $\boldsymbol{G}_{\mathsf{pub}}$ be as in eq. (2.4) and assume that $\operatorname{rank}\boldsymbol{X}_2 = t_{\boldsymbol{X}_2}$. Then, there exist $\boldsymbol{P}^* \in \mathsf{GL}_{l+n}(\mathbb{F}_q)$, $\boldsymbol{X}^* \in \mathbb{F}_{q^m}^{k\times(l+t_{\boldsymbol{X}_2})}$, and $\boldsymbol{G}^*$ which generates an $[n-t_{\boldsymbol{X}_2}, k]$-Gabidulin code $\mathfrak{G}_k(\boldsymbol{g}^*)$ such that*

$$\boldsymbol{G}_{\mathsf{pub}} = \boldsymbol{S}\begin{bmatrix}\boldsymbol{X}^* & \boldsymbol{G}^*\end{bmatrix}\boldsymbol{P}^*.$$

Furthermore, $\mathfrak{G}_k(\boldsymbol{g}^*)$ can correct more than t_{pub} errors.

Proof. Since $\operatorname{rank} \boldsymbol{X}_2 = t_{\boldsymbol{X}_2}$, there exist $\boldsymbol{T}_2 \in \mathsf{GL}_n(\mathbb{F}_q)$, and $\boldsymbol{X}'_2 \in \mathbb{F}_{q^m}^{k \times t_{\boldsymbol{X}_2}}$ such that

$$\boldsymbol{X}_2 \boldsymbol{T}_2 = \begin{bmatrix} \boldsymbol{X}'_2 & \boldsymbol{0} \end{bmatrix}.$$

(Consider the process of Gaussian elimination.) Thus, letting

$$\boldsymbol{T} = \begin{bmatrix} \boldsymbol{I}_l & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{T}_2 \end{bmatrix},$$

we obtain

$$\begin{aligned} \boldsymbol{G}_{\mathsf{pub}} &= \boldsymbol{S} \begin{bmatrix} \boldsymbol{X}_1 & \boldsymbol{G} + \boldsymbol{X}_2 \end{bmatrix} \boldsymbol{P} \\ &= \boldsymbol{S} \begin{bmatrix} \boldsymbol{X}_1 & \boldsymbol{G}\boldsymbol{T}_2 + \boldsymbol{X}_2 \boldsymbol{T}_2 \end{bmatrix} \boldsymbol{T}^{-1} \boldsymbol{P} \\ &=: \boldsymbol{S} \begin{bmatrix} \boldsymbol{X}_1 & \boldsymbol{G}' + \boldsymbol{X}_2 \boldsymbol{T}_2 \end{bmatrix} \boldsymbol{Q}; \end{aligned}$$

where $\boldsymbol{G}' \coloneqq \boldsymbol{G}\boldsymbol{T}_2$, and $\boldsymbol{Q} \coloneqq \boldsymbol{T}^{-1}\boldsymbol{P}$. $\boldsymbol{G}'$ generates the $[n, k]$-Gabidulin code $\mathfrak{G}_k(\boldsymbol{g}')$, with $\boldsymbol{g}' = \boldsymbol{g}\boldsymbol{T}_2$. If we decompose $\boldsymbol{G}'$ as $\begin{bmatrix} \boldsymbol{G}'_1 & \boldsymbol{G}'_2 \end{bmatrix}$, where $\boldsymbol{G}'_1 \in \mathbb{F}_{q^m}^{k \times t_{\boldsymbol{X}_2}}$, and $\boldsymbol{G}'_2 \in \mathbb{F}_{q^m}^{k \times (n - t_{\boldsymbol{X}_2})}$, we obtain

$$\boldsymbol{G}' + \boldsymbol{X}_2 \boldsymbol{T}_2 = \begin{bmatrix} \boldsymbol{G}'_1 + \boldsymbol{X}'_2 & \boldsymbol{G}'_2 \end{bmatrix}.$$

By setting $\boldsymbol{X}^* = \begin{bmatrix} \boldsymbol{X}_1 & \boldsymbol{G}'_1 + \boldsymbol{X}'_2 \end{bmatrix}$, we get the desired result. $\boldsymbol{G}'_2$ generates the $[n - t_{\boldsymbol{X}_2}, k]$-Gabidulin code $\mathfrak{G}_k(\boldsymbol{g}'_2)$, where $\boldsymbol{g}'_2 = \begin{bmatrix} g'_{t_{\boldsymbol{X}_2}+1} & \cdots & g'_n \end{bmatrix}$. The error-correction capacity t^* of $\mathfrak{G}_k(\boldsymbol{g}'_2)$ is $(n - t_{\boldsymbol{X}_2} - k)/2 = t - t_{\boldsymbol{X}_2}/2 \implies t^* > t - t_{\boldsymbol{X}_2}$. □

3. Structural Attacks Against GPT

Gabidulin codes possess a rich structure that has been exploited to mount attacks on the GPT cryptosystem. In this chapter, we review the most notable of these attacks: Overbeck's attack. We prove that Overbeck's attack applies to a very general form of the GPT cryptosystem, and then state the properties that make the attack possible. At the end, we formulate Loidreau's recent reparation which avoids Overbeck's attack.

3.1. Intuition Behind the Attacks

Let $\mathcal{C}$ be an $[n, k, d]$-Gabidulin code generated by $\boldsymbol{G}$. Recall that $\boldsymbol{G}$ is of the form

$$\begin{bmatrix} g_1 & \cdots & g_n \\ g_1^{[1]} & \cdots & g_n^{[1]} \\ \vdots & \ddots & \vdots \\ g_1^{[k-1]} & \cdots & g_n^{[k-1]} \end{bmatrix}.$$

Now, one can see that

$$\mathcal{C}^{[1]} \cup \mathcal{C}$$

represents a Gabidulin code of dimension $k-1$. ($\mathcal{C}^{[1]}$ denotes the code generated by $\boldsymbol{G}^{[1]}$.) This was the basis of a polynomial time algorithm to retrieve the hidden Gabidulin structure in the GPT cryptosystem.

3.2. Distinguishing Properties

We now formally state the distinguishing properties of Gabidulin codes that have been exploited by Overbeck [25, 26, 27] to mount successful attacks.

Definition 14. For any $i \in \mathbb{N}$, let $\Lambda_i \colon \mathbb{F}_{q^m}^{k \times n} \to \mathbb{F}_{q^m}^{ik \times n}$ be the $\mathbb{F}_q$-linear operator defined as:

$$\Lambda_i(\boldsymbol{X}) := \begin{bmatrix} \boldsymbol{X}^{[0]} \\ \boldsymbol{X}^{[1]} \\ \vdots \\ \boldsymbol{X}^{[i]} \end{bmatrix}.$$

For any code $\mathcal{C}$ generated by $\boldsymbol{G}$, we denote by $\Lambda_i(\mathcal{C})$ the code generated by $\Lambda_i(\boldsymbol{G})$.

Lemma 5. *Let $\boldsymbol{g} \in \mathbb{F}_{q^m}^n$ with $\|\boldsymbol{g}\|_q = n$. For $k, i \in \mathbb{N}$ such that $k \leq n$ and $i \leq n-k-1$, we have:*

$$\Lambda_i(\mathfrak{G}_k(\boldsymbol{g})) = \mathfrak{G}_{k+i}(\boldsymbol{g}).$$

On the other hand, if $\boldsymbol{G}$ is a randomly-drawn matrix, we obtain something quite different.

Lemma 6. *If $\mathcal{C} \subset \mathbb{F}_{q^m}^n$ is a code generated by a random matrix $\boldsymbol{G} \in \mathbb{F}_{q^m}^{k \times n}$,*

$$\dim \Lambda_i(\mathcal{C}) = \min\{n, (i+1)k\}$$

with high probability.

3.3. Overbeck's Attack

Lemma 7. *Let $\boldsymbol{P} = \left[\begin{smallmatrix} \boldsymbol{A} & \boldsymbol{0} \\ \boldsymbol{C} & \boldsymbol{D} \end{smallmatrix}\right]$ where $\boldsymbol{A}$ and $\boldsymbol{D}$ are square matrices. $\boldsymbol{P}$ is nonsingular if and only if $\boldsymbol{A}$ and $\boldsymbol{D}$ are nonsingular, and*

$$\boldsymbol{P}^{-1} = \begin{bmatrix} \boldsymbol{A}^{-1} & \boldsymbol{0} \\ -\boldsymbol{D}^{-1}\boldsymbol{C}\boldsymbol{A}^{-1} & \boldsymbol{D}^{-1} \end{bmatrix}.$$

Assume that $\boldsymbol{G}_{\mathsf{pub}} = \boldsymbol{S}\begin{bmatrix}\boldsymbol{X} & \boldsymbol{G}\end{bmatrix}\boldsymbol{P}$ is the public generator matrix with $\boldsymbol{P} \in \mathsf{GL}_{l+n}(\mathbb{F}_q)$, $\boldsymbol{X} \in \mathbb{F}_{q^m}^{k \times l}$ and $\boldsymbol{G}$ generates a Gabidulin code $\mathfrak{G}_k(\boldsymbol{g})$. Observe that

$$\Lambda_i(\boldsymbol{G}_{\mathsf{pub}}) = \boldsymbol{S}_{\mathsf{ext}}\begin{bmatrix}\Lambda_i(\boldsymbol{X}) & \Lambda_i(\boldsymbol{G})\end{bmatrix}\boldsymbol{P},$$

where

$$\boldsymbol{S}_{\mathsf{ext}} := \begin{bmatrix} \boldsymbol{S}^{[0]} & \cdots & \boldsymbol{0} \\ \vdots & \ddots & \vdots \\ \boldsymbol{0} & \cdots & \boldsymbol{S}^{[i]} \end{bmatrix}.$$

Since $\Lambda_i(\boldsymbol{G})$ generates $\mathfrak{G}_{k+i}(\boldsymbol{g})$, there exists $\boldsymbol{S}' \in \mathsf{GL}_{k(i+1)}(\mathbb{F}_{q^m})$ such that

$$\boldsymbol{S}'\Lambda_i(\boldsymbol{G}_{\mathsf{pub}}) = \begin{bmatrix} \boldsymbol{X}^* & \boldsymbol{G}_{k+i} \\ \boldsymbol{X}^{**} & \boldsymbol{0} \end{bmatrix}. \tag{3.1}$$

Using eq. (3.1), one can see that when $i = n-k-1$,

$$\dim \Lambda_{n-k-1}(\mathcal{C}_{\mathsf{pub}}) = n - 1 + \operatorname{rank} \boldsymbol{X}^{**}.$$

When $\operatorname{rank} \boldsymbol{X}^{**} = l$, $\dim \Lambda_{n-k-1}(\mathcal{C}_{\mathsf{pub}}) = n + l - 1$, and thus $\dim \Lambda_{n-k-1}(\mathcal{C}_{\mathsf{pub}})^\perp = 1$. If $\boldsymbol{h} \in \mathfrak{G}_{n-1}(\boldsymbol{g})^\perp$ is a nonzero vector, and we set $\boldsymbol{h}^* = \begin{bmatrix}\boldsymbol{0} & \boldsymbol{h}\end{bmatrix}(\boldsymbol{P}^{-1})^\top$, then — under the assumption that $\operatorname{rank} \boldsymbol{X}^{**} = l$ — we have

$$\Lambda_{n-k-1}(\mathcal{C}_{\mathsf{pub}})^\perp = \operatorname{span}\{\boldsymbol{h}^*\}. \tag{3.2}$$

Theorem 3. *Let $\boldsymbol{v} \in \Lambda_{n-k-1}(\mathcal{C}_{\mathsf{pub}})^{\perp}$ be a nonzero vector. Any matrix $\boldsymbol{T} \in \mathsf{GL}_{l+n}(\mathbb{F}_q)$ that satisfies $\boldsymbol{vT} = \begin{bmatrix}\boldsymbol{0} & \boldsymbol{h}'\end{bmatrix}$ with $\boldsymbol{h}' \in \mathbb{F}_{q^m}^n$ is an alternative column scrambler; i.e., there exist $\boldsymbol{Z} \in \mathbb{F}_{q^m}^{k\times l}$ and $\boldsymbol{G}^*$ generating $\mathfrak{G}_k(\boldsymbol{g}^*)$ such that*

$$\boldsymbol{G}_{\mathsf{pub}} = \boldsymbol{S}\begin{bmatrix}\boldsymbol{Z} & \boldsymbol{G}^*\end{bmatrix}\boldsymbol{T}.$$

Proof. From eq. (3.2), there exists $\alpha \in \mathbb{F}_{q^m}$ such that $\boldsymbol{v} = \alpha\boldsymbol{h}^* = \begin{bmatrix}\boldsymbol{0} & \alpha\boldsymbol{h}\end{bmatrix}(\boldsymbol{P}^{-1})^{\top}$, where $\boldsymbol{h} \in \mathfrak{G}_{n-1}(\boldsymbol{g})^{\perp}$ is a nonzero vector. Let $\boldsymbol{T} \in \mathsf{GL}_{l+n}(\mathbb{F}_q)$ be such that $\boldsymbol{vT}^{\top} = \begin{bmatrix}\boldsymbol{0} & \boldsymbol{h}'\end{bmatrix}$, and consider the matrices $\boldsymbol{A} \in \mathbb{F}_q^{l\times l}$ and $\boldsymbol{D} \in \mathbb{F}_q^{n\times n}$ so that

$$\boldsymbol{TP}^{-1} = \begin{bmatrix}\boldsymbol{A} & \boldsymbol{B}\\ \boldsymbol{C} & \boldsymbol{D}\end{bmatrix}.$$

Now, we have

$$\boldsymbol{vT}^{\top} = \begin{bmatrix}\boldsymbol{0} & \alpha\boldsymbol{h}\end{bmatrix}(\boldsymbol{P}^{-1})^{\top}\boldsymbol{T}^{\top} = \begin{bmatrix}\boldsymbol{0} & \alpha\boldsymbol{h}\end{bmatrix}(\boldsymbol{P}^{-1}\boldsymbol{T})^{\top} = \begin{bmatrix}\boldsymbol{0} & \boldsymbol{h}'\end{bmatrix}.$$

Notice that $\boldsymbol{hB}^{\top} = \boldsymbol{0} \implies \boldsymbol{B} = \boldsymbol{0}$ since $\|\boldsymbol{h}\|_q = n$. Thus,

$$\boldsymbol{TP}^{-1} = \begin{bmatrix}\boldsymbol{A} & \boldsymbol{0}\\ \boldsymbol{C} & \boldsymbol{D}\end{bmatrix} \implies \boldsymbol{PT}^{-1} = \begin{bmatrix}\boldsymbol{A}' & \boldsymbol{0}\\ \boldsymbol{C}' & \boldsymbol{D}'\end{bmatrix}.$$

From here,

$$\boldsymbol{G}_{\mathsf{pub}}\boldsymbol{T}^{-1} = \boldsymbol{S}\begin{bmatrix}\boldsymbol{X} & \boldsymbol{G}\end{bmatrix}\begin{bmatrix}\boldsymbol{A}' & \boldsymbol{0}\\ \boldsymbol{C}' & \boldsymbol{D}'\end{bmatrix} = \boldsymbol{S}\begin{bmatrix}\boldsymbol{Z} & \boldsymbol{G}^*\end{bmatrix},$$

where $\boldsymbol{G}^* \coloneqq \boldsymbol{GD}'$ is a generator matrix of an $[n,k]$-Gabidulin code. □

3.4. Defending Against Overbeck's Attack

As we saw, Overbeck's attack is quite general as it can be applied to a wide variety of GPT-like cryptosystems. In order to defend against it, we need to see what makes it feasible to perform the attack. There are two key properties:

Property 1. The column scrambler $\boldsymbol{P}$ is defined over the base field; and

Property 2. The codimension of $\Lambda_{n-k-1}(\mathcal{C})$ is 1.

Here, we focus on property 1. We can relax the optimality of the code by scrambling the columns with a nonisometry of the metric. This was already done in the Hamming metric in the case of GRS codes.

Lemma 8. *Let $\mathcal{V} = \{\beta_1, \dots, \beta_\lambda\}$ be the $\mathbb{F}_q$-linear subspace generated by $\{\beta_1, \dots, \beta_\lambda\}$. Let $\boldsymbol{P} \in \mathsf{GL}_n(\mathcal{V})$. Then,*

$$\forall \boldsymbol{x} \in \mathbb{F}_{q^m}^n.\ \|\boldsymbol{xP}\|_q \leq \lambda\|\boldsymbol{x}\|_q.$$

Corollary 2. *Let $\mathcal{C}$ be an $[n, k, d]$-Gabidulin code over $\mathbb{F}_{q^m}$. Let $\mathcal{V}$ be a λ-dimensional subspace of $\mathbb{F}_{q^m}$, and let $\boldsymbol{P} \in \mathsf{GL}_n(\mathcal{V})$. Then,*

$$\mathcal{C}\boldsymbol{P}^{-1} := \{\boldsymbol{c}\boldsymbol{P}^{-1} : \boldsymbol{c} \in \mathcal{C}\}$$

has dimension k and minimum distance $d' \geq \lfloor c/\lambda \rfloor$.

Proof. Since $\boldsymbol{P}$ is nonsingular, $\mathcal{C}$ and $\mathcal{C}\boldsymbol{P}^{-1}$ have the same dimension. Now, suppose $d' < d/\lambda$, and let $\boldsymbol{c} \in \mathcal{C}\boldsymbol{P}^{-1}$ be a nonzero vector with $\|\boldsymbol{c}\|_q = d'$. By construction, $\boldsymbol{c}\boldsymbol{P} \in \mathcal{C}$, and by lemma 8, $\|\boldsymbol{c}\boldsymbol{P}\|_q \leq d'\lambda < d \implies \boldsymbol{c}\boldsymbol{P} = \boldsymbol{0}$. Thus, $\boldsymbol{c} = \boldsymbol{0}$, which contradicts the hypothesis. □

We can use this property to design a cryptosystem where the column scrambler $\boldsymbol{P}$ is defined over the extension field.

3.5. Reparation by Loidreau [23]

Loidreau [23] proposed and analyzed the security of a cryptosystem based on the aforementioned ideas. What follows is a description of the system and the parameters proposed by the author.

Private key

1. Gabidulin code $\mathfrak{G}_k(\boldsymbol{g})$ with $\|\boldsymbol{g}\|_q = n$, generator matrix $\boldsymbol{G}$ as in eq. (2.3), and error-correction capacity $t := \lfloor (n-k)/2 \rfloor$.
2. $\boldsymbol{S} \in \mathsf{GL}_k(\mathbb{F}_{q^m})$, the *row scrambler*.
3. $\mathcal{V}$, a λ-dimensional subspace of $\mathbb{F}_{q^m}$; and $\boldsymbol{P} \in \mathsf{GL}_n(\mathcal{V})$, the *column scrambler*.

Public key Tuple $(\boldsymbol{G}_{\mathsf{pub}}, t_{\mathsf{pub}})$, where

$$\boldsymbol{G}_{\mathsf{pub}} := \boldsymbol{S}\boldsymbol{G}\boldsymbol{P}^{-1} \text{ and } t_{\mathsf{pub}} = \left\lceil \frac{n-k}{2\lambda} \right\rceil.$$

Encryption To encrypt a message $\boldsymbol{x} \in \mathbb{F}_{q^m}^k$, we choose a random error $\boldsymbol{e} \in \mathbb{F}_{q^m}^n$ with $\|e\|_q \leq t_{\mathsf{pub}}$ and compute the ciphertext: $\boldsymbol{y} = \boldsymbol{x}\boldsymbol{G}_{\mathsf{pub}} + \boldsymbol{e}$.

Table 3.1.: Proposed parameters for the Loidreau cryptosystem

m	n	k	λ	Claimed security (bit)	Key size (kB)
50	50	32	3	64	3.6
80	80	11	3	110	8.3
96	64	40	3	120	11.5
128	90	24	3	240	21.5
128	120	80	5	240	51

Decryption Compute $\boldsymbol{yP} = \boldsymbol{xSG} + \boldsymbol{eP}$. We know that

$$\|\boldsymbol{eP}\|_q \leq \lambda \left\lceil \frac{n-k}{2\lambda} \right\rceil \leq \left\lceil \frac{n-k}{2} \right\rceil,$$

thus, we can decode using a decoding algorithm for $\mathfrak{G}_k(\boldsymbol{g})$ to obtain $\boldsymbol{xS}$. From here, we can retreive $\boldsymbol{x}$ immediately.

3.6. Analysis of the Cryptosystem

Having the column scrambler defined over the extension field nullifies Overbeck's attack since it is no longer true that

$$\boldsymbol{P}^{[i]} = \boldsymbol{P}.$$

As a matter of fact, there is no known reason for the public key to have any particular structure. Namely, even though the entries of $\boldsymbol{P}$ are all in some λ-dimensional subspace $\mathcal{V}$, there is no reason for the entries of $\boldsymbol{P}^{-1}$ to be in any subspace. The proposed parameters can be found in table 3.1.

4. Information Set Decoding

As we saw, decoding random linear codes is a fundamental problem not only in coding theory, but also in cryptography. We established that it is a "very hard" problem and is conjectured to be hard on average. We saw a family of codes — Gabidulin codes — which can be decoded efficiently, but most codes are not known to have an efficient decoding algorithm.

There are several known techniques for decoding random linear codes. In the Hamming metric, the best algorithms that we know of rely on information set decoding (ISD), originally proposed by McEliece [24] and inspired by the work of Prange [29]. In short, the idea behind ISD is to pick a sufficiently large set of error-free positions in a received word. Then, the information sequence can be obtained by linear algebra.

Since its introduction, ISD has been refined to the point of being able to decode random binary linear codes in the Hamming metric with complexity $\mathcal{O}(2^{n/20})$ [2]. However, ISD has not been explored in the context of the rank metric. In this chapter, we introduce and analyze ISD algorithms suitable for the rank metric. First, we introduce the MinRank problem in sections 4.1 and 4.2, as it will be used as a subproblem is one of the ISD algorithms.

4.1. The MinRank Problem

Definition 15 (MinRank over a field)**.** Let $\mathbb{K}$ be a field. Let $\boldsymbol{M}_0; \boldsymbol{M}_1, \dots, \boldsymbol{M}_m$ be matrices from $\mathbb{K}^{\mu \times \nu}$, and let $r \in \mathbb{N}$. The MinRank problem instance asks us to find a vector $\boldsymbol{\alpha} := \begin{bmatrix} \alpha_1 & \cdots & \alpha_m \end{bmatrix} \in \mathbb{K}^m$ such that

$$\operatorname{rank}\left(\sum_{i=1}^{m} \alpha_i \boldsymbol{M}_i - \boldsymbol{M}_0\right) \leq r.$$

We denote by $\mathsf{MR}(m, \mu, \nu, r, \mathbb{K}; \boldsymbol{M}_0; \boldsymbol{M}_1, \dots, \boldsymbol{M}_m)$ the set of all solutions to the problem.

Remark. Typically, $\mathbb{K} := \mathbb{F}_q$ for q (power of a) prime. We will implicitly assume this hereafter.

MinRank is known to be $\mathcal{NP}$-complete. As a matter of fact, any system of multivariate polynomial equations can be encoded as a MinRank instance. More interestingly, Minimum Distance Decoding in the rank metric can be reduced to solving a MinRank instance.

Theorem 4. *Let $\mathcal{C}$ be an $[n,k]$-code over $\mathbb{F}_{q^m}$ in the rank metric. There is a many-to-one reduction from* $\mathsf{MDD}(\mathcal{C}, \boldsymbol{y}, w)$ *to* $\mathsf{MR}(mk, m, k, w, \mathbb{F}_q; \boldsymbol{M}_0; \boldsymbol{M}_1, \dots, \boldsymbol{M}_{mk})$.

Proof sketch. Each entry of $\boldsymbol{y} - \boldsymbol{c} =: \boldsymbol{e}$ can be written in a basis $\mathcal{B} = (\boldsymbol{\beta}_1, \dots, \boldsymbol{\beta}_m)$. For each of the k coordinates of $\boldsymbol{c} = \boldsymbol{x}\boldsymbol{G}$, we can "expand" each $\boldsymbol{\beta}_i$ into an $m \times k$ matrix over $\mathbb{F}_q$, for a total of mk such matrices. □

4.2. Solving MinRank Instances

The two most practical and most studied methods of solving MinRank instances are:

- Guessing the kernel of the solution — the so-called "kernel attack"; and
- Modeling a MinRank instance as a system of polynomial (multivariate quadratic) equations.

4.2.1. The Kernel Attack

This was the first non-trivial attack against MinRank and was proposed by Goubin and Courtois [16]. The main idea is that instead of guessing $\boldsymbol{\alpha}$, we can guess the kernel of the resulting matrix. Then, assuming that the kernel was guessed correctly, we can *solve* for $\boldsymbol{\alpha}$.

For some parameter $\boldsymbol{\beta} \in \mathbb{F}_q^m$, denote by $H(\boldsymbol{\beta})$ the linear combination

$$\sum_{i=1}^{m} \beta_i \boldsymbol{M}_i - \boldsymbol{M}_0.$$

We would like to have $\operatorname{rank} H(\boldsymbol{\beta}) \leq r$, i.e., $\boldsymbol{\beta}$ to be a solution. If that were the case, then by the rank-nullity theorem of linear algebra, we would have $\dim \ker H(\boldsymbol{\beta}) \geq \nu - r$.

We can proceed by randomly choosing κ vectors $\boldsymbol{x}^{(i)} \in \mathbb{F}_q^\nu$ for $1 \leq i \leq \kappa$. If these vectors are such that they all fall into the kernel of a solution to the MinRank instance, then solving for $\boldsymbol{\beta}$ in

$$\begin{cases} H(\boldsymbol{\beta})\boldsymbol{x}^{(1)} = \boldsymbol{0} \\ H(\boldsymbol{\beta})\boldsymbol{x}^{(2)} = \boldsymbol{0} \\ \qquad\qquad \vdots \\ H(\boldsymbol{\beta})\boldsymbol{x}^{(\kappa)} = \boldsymbol{0} \end{cases} \tag{4.1}$$

would allow us to retrieve the correct solution. Equation (4.1) is a system in $\kappa\mu$ equations and m unknowns. Since we would like eq. (4.1) to never be under-determined and have its numbers of equations and unknowns match as closely as possible, we can choose $\kappa := \lceil m/\mu \rceil$.

It was established that there are at least $q^{\nu-r}$ vectors in $\ker H(\boldsymbol{\beta})$ whenever $\boldsymbol{\beta}$ is a solution. Having that in mind, we can see that

$$\Pr(\{\boldsymbol{x}^{(1)}, \dots, \boldsymbol{x}^{(\kappa)}\} \subseteq \ker H(\boldsymbol{\beta})) \geq q^{-\kappa r} = q^{-\lceil m/\mu \rceil r}.$$

From here, we get an overall complexity of $\mathcal{O}(m(\lceil m/\mu \rceil \mu)^2 q^{\lceil m/\mu \rceil r})$. The takeaway is that the kernel attack is significantly better than a brute-force enumeration whenever $\lceil m/\mu \rceil r \ll m$, which is almost always the case.

4.2.2. As a System of Multivariate Quadratic Equations

Notice that instead of guessing, we can explicitly construct the kernel's solution, i.e., treat the $\boldsymbol{x}^{(i)}$ in eq. (4.1) as unknowns. This modeling was first proposed by Kipnis and Shamir [18].

Define $H(\boldsymbol{\beta})$ as in section 4.2.1. We already saw that whenever $\boldsymbol{\beta}$ is a solution, there are at least $\nu - r$ linearly independent vectors in $\ker H(\boldsymbol{\beta})$. Proceed by trying to fix these vectors as follows:

$$\begin{aligned} \boldsymbol{x}^{(1)} &:= \begin{bmatrix} 1 & 0 & \cdots & 0 & x_1^{(1)} & \cdots & x_r^{(1)} \end{bmatrix}^\top \\ &\vdots \\ \boldsymbol{x}^{(\nu-r)} &:= \begin{bmatrix} 0 & 0 & \cdots & 1 & x_1^{(\nu-r)} & \cdots & x_r^{(\nu-r)} \end{bmatrix}^\top . \end{aligned} \tag{4.2}$$

Remark. Equation (4.2) is true up to a *change of basis*, so it might happen that it does not arrive at a solution.

Bearing this in mind, one can see that the multivariate quadratic (MQ) system

$$\left(\sum_{i=1}^{m} \beta_i \boldsymbol{M}_i - \boldsymbol{M}_0 \right) \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \\ x_1^{(1)} & x_1^{(2)} & \cdots & x_1^{(n-r)} \\ \vdots & \vdots & \ddots & \vdots \\ x_r^{(1)} & x_r^{(2)} & \cdots & x_r^{(n-r)} \end{bmatrix} = \mathbf{0}_{\mu \times (\nu-r)} \tag{4.3}$$

over $\mathbb{F}_q[\beta_1, \dots, \beta_m, x_1^{(1)}, \dots, x_r^{(\nu-r)}]$ consists of $\mu(\nu-r)$ equations and $r(\nu-r)+\mu$ unknowns. The complexity of solving eq. (4.3) is hard to evaluate and depends a lot on the method. Nevertheless, it is exponential in the worst case.

Kipnis and Shamir [18] proposed solving eq. (4.3) by *relinearization*. However, Gröbner bases are a more popular approach for solving any system of polynomial equations. One can think of them as analogous to the process of Gaussian elimination.

Solving eq. (4.3) by Gröbner bases Faugère, Levy-dit-Vehel, and Perret [7] proposed a one-to-one correspondence between the affine variety

$$\mathcal{V}(\mathcal{J}_{\mathsf{KS}}) := \{\boldsymbol{x} \in \mathbb{F}_q^{r(\nu-r)+m} : \forall f \in \mathcal{J}_{\mathsf{KS}}.\, f(\boldsymbol{x}) = 0\}$$

and the solution to eq. (4.3), where $\mathcal{J}_{\mathsf{KS}}$ is the ideal generated by the equations of eq. (4.3). This allows one to use Gröbner bases to solve the system. The theoretical complexity of computing a Gröbner basis of a polynomial system with m equations and n unknowns is given by

$$\mathcal{O}\left(m\binom{n+d_{\mathsf{reg}}}{d_{\mathsf{reg}}}^{\omega}\right),$$

where d_{reg} is the degree of regularity (i.e., the highest degree reached during the computation), and $2 \leq \omega \leq 3$ is the exponent in the complexity of matrix multiplication. However, in practice — and especially for MinRank instances since the equations of eq. (4.3) are bilinear — the complexity is much lower [7] and can often be experimentally bounded by a polynomial.

4.3. The Information Set Decoding Algorithms

In this section, we formulate and analyze two variants of information set decoding algorithms suitable for decoding in the rank metric: (1) algorithms using sets of coordinates (section 4.3.2); and (2) algorithms using projections onto subspaces (section 4.3.3). The former are a straightforward adaptation of the algorithms in the Hamming metric, while the latter use projections as the "rank metric analog" for sets of coordinates.

4.3.1. Algorithms in the Hamming Metric

All information set decoding algorithms studied in this thesis can be described in two steps:

Step 1. Choosing a random information set and assuming it satisfies certain condition w.r.t. the error pattern; and

Step 2. Decoding the received word $\boldsymbol{y}$ assuming that the information set chosen in step 1 is as assumed.

In this section, we are going to restrict our attention to *binary* codes in the Hamming metric. The definition of an information set (definition 6) immediately lends itself to the simplest ISD algorithm, *plain information set decoding* (Plain-ISD).

Plain information set decoding randomly chooses an information set and then computes the information by solving linear equations. In other words, it assumes that there are *no*

errors within the k positions indexed by the information set. Let $\boldsymbol{x} \in \mathbb{F}_2^k$ be the unknown message, and

$$\boldsymbol{y} \coloneqq \boldsymbol{x}\boldsymbol{G} + \boldsymbol{e} \in \mathbb{F}_2^n$$

be the received word. Suppose we pick an information set $\mathcal{I} \subset \{1, \dots, n\}$ with $|\mathcal{I}| = k$. If the $\mathcal{I}$-indexed coordinates of $\boldsymbol{y}$ are error-free, i.e., if $\boldsymbol{e}_{\mathcal{I}} = \boldsymbol{0}$, then we can compute $\boldsymbol{x}$ by simple linear algebra:

$$\boldsymbol{y}_{\mathcal{I}}\boldsymbol{G}_{\mathcal{I}}^{-1} = ((\boldsymbol{x}\boldsymbol{G})_{\mathcal{I}} + \boldsymbol{e}_{\mathcal{I}})\boldsymbol{G}_{\mathcal{I}}^{-1} = \boldsymbol{x}.$$

This idea originated from Prange [29]. In order to analyze the complexity of the algorithm, we need to determine the probability of success, i.e., picking an error-free information set, and the complexity of computing $\boldsymbol{x}$.

Denote by $\operatorname{supp}(\boldsymbol{e})$ the set of all coordinates of the nonzero entries of $\boldsymbol{e}$, $\operatorname{supp}(\boldsymbol{e}) = \{i : e_i \neq 0\}$. Then, the probability of success is

$$\Pr(\mathcal{I} \cap \operatorname{supp}(\boldsymbol{e}) = \emptyset) = \binom{n-k}{w} \binom{n}{w}^{-1},$$

where w is the Hamming weight of $\boldsymbol{e}$. To see that this is the case, notice that there are $\binom{n-k}{w}$ possible choices for $\boldsymbol{e}$ if the information set is to be error-free; whereas there are $\binom{n}{w}$ choices if this restriction is lifted. The cost of decoding $\boldsymbol{y}$ is simply the cost of computing the inverse of a matrix by Gaussian elimination. It is common to consider it as $\frac{(n-k)k^2}{2}$, although significant improvements are known.

The algorithm performs a series of independent iterations, where each iteration consists of a randomly chosen information set $\mathcal{I}$. As justified by theorem 5, the expected running time is

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

Theorem 5. *Consider a Las Vegas algorithm which succeeds to produce a correct answer with probability ρ and outputs $\perp$ with probability $1 - \rho$. Let X be a random variable that represents the running time of the algorithm. Then, its expected running time is*

$$\mathrm{E}[X] = \frac{1}{p}.$$

Proof. Since we will run the algorithm until we observe a correct answer, X follows a geometric distribution. More specifically, the probability that we have to run the algorithm i times is given by

$$\Pr(X = i) = \rho(1 - \rho)^{i-1}.$$

It is a well-known result that a random variable with such a distribution has expectation equal to $1/\rho$. □

Lee and Brickell [20] explored a generalization of plain information set decoding, allowing a set of $p \in \{0, \dots, w\}$ errors in the information set. The parameter p is typically chosen to be a small number; Peters [28] proved that $p = 2$ is in fact optimal in the binary case, improving the complexity by a factor $\Theta(n \log n)$ over plain information set decoding ($p = 0$). Assuming that $\boldsymbol{e}$ is uniformly random, the probability of having exactly p errors in the information set is

$$\Pr(|\mathcal{J} \cap \operatorname{supp}(\boldsymbol{e})| = p) = \binom{n-k}{w-p}\binom{k}{p}\binom{n}{w}^{-1},$$

which also gives the probability of success in one iteration of the Lee–Brickell algorithm.

4.3.2. Algorithms Using Sets of Coordinates in the Rank Metric

The algorithms outlined in section 4.3.1 employ the "classical" definition of an information set (definition 6). As we already mentioned, they have been the most successful way of attacking McEliece-like cryptosystems, but have not been explored in the context of the rank metric. We formulate both plain information set decoding and the improvement by Lee and Brickell [20], and analyze their applicability in the rank metric. We also provide some thoughts on the feasibility applying Stern's and Dumer's ideas [35, 6].

Plain Information Set Decoding

We saw that Plain-ISD randomly chooses an information set and assumes that the error indexed by it is of weight 0. This is a very natural notion for the Hamming metric; however, for the rank metric, it translates to the submatrix formed by the $\mathcal{J}$-indexed coordinates being the zero matrix. Algorithm 4.1 summarizes the procedure.

Algorithm 4.1: Plain-ISD using sets of coordinates in the rank metric

Input: Generator matrix $\boldsymbol{G} \in \mathbb{F}_{q^m}^{k \times n}$, received word $\boldsymbol{y} \in \mathbb{F}_q^n$, and target weight $w \in \mathbb{N}$
Output: Error $\boldsymbol{e} \in \mathbb{F}_q^n$ with $\|\boldsymbol{e}\|_q \leq w$ and $\boldsymbol{y} - \boldsymbol{e} \in \mathcal{C}$

```
begin
    repeat
        J ←$ {1, ..., n} with |J| = k
        e' ← y − y_J G_J^{-1} G
        if ||e'||_q ≤ w then return e'
    until a solution is found
end
```

Subsequent experiments will show that this algorithm is not applicable to the rank metric. Indeed, there is just a single matrix of rank 0 (the zero matrix), and the probability of the $\mathcal{J}$-indexed coordinates of $\boldsymbol{e}$ being the zero matrix becomes negligible even for small parameters.

Improvement by Lee and Brickell [20]

As discussed in section 4.3.1, Lee and Brickell [20] published a variation of Plain-ISD (called LB-ISD hereafter), by allowing $p \in \{0, \dots, w\}$ errors within the information set. These p errors are then decoded by brute-force enumeration. In the Hamming metric, this is equivalent to iterating over all words in a Hamming sphere of radius p. However, in the rank metric, we are presented with two options:

1. Iterate over rank-p matrices; or
2. Iterate over p-dimensional subspaces and solve a MinRank-like subproblem.

We will see that the second option is the more favorable one. Algorithm 4.2 shows the procedure. Evidently, the case of $p = 0$ corresponds directly to Plain-ISD.

Algorithm 4.2: LB-ISD using sets of coordinates in the rank metric

Input: Generator matrix $\boldsymbol{G} \in \mathbb{F}_{q^m}^{k \times n}$, received word $\boldsymbol{y} \in \mathbb{F}_{q^m}^{n}$, target weight $w \in \mathbb{N}$, and parameter $p \in \mathbb{N}$ with $p \leq w$

Output: Error $\boldsymbol{e} \in \mathbb{F}_{q^m}^{n}$ with $\|\boldsymbol{e}\|_q \leq w$ and $\boldsymbol{y} - \boldsymbol{e} \in \mathcal{C}$

begin

  repeat

    $\mathcal{J} \leftarrow_{\$} \{1, \dots, n\}$ with $|\mathcal{J}| = k$

    $\boldsymbol{e}' \leftarrow \boldsymbol{y} - \boldsymbol{y}_{\mathcal{J}} \boldsymbol{G}_{\mathcal{J}}^{-1} \boldsymbol{G}$

    forall $\mathcal{V} \leq \mathbb{F}_q^m$ *with* $\dim \mathcal{V} = p$ **do**

      $\langle \boldsymbol{\beta}_1, \dots, \boldsymbol{\beta}_p \rangle \leftarrow$ basis of $\mathcal{V}$

      $\boldsymbol{e}'' := \begin{bmatrix} e''_1 & \cdots & e''_k \end{bmatrix} \boldsymbol{G}_{\mathcal{J}}^{-1} \boldsymbol{G}$

      $e''_j := \sum_{l=1}^{p} a_{j,l} \boldsymbol{\beta}_l$ `/* the` $a_{j,l}$`'s are unknowns */`

9   **if** $\exists$(*assignment to the* $a_{j,l}$*'s*). $\|\boldsymbol{e}' - \boldsymbol{e}''\|_q \leq w$ **then**

        return $(\boldsymbol{e}' - \boldsymbol{e}'')$ *with the* $a_{j,l}$*'s substituted*

    end forall

  until *a solution is found*

end

Given a guess for $\mathcal{V}$, the subproblem on line 9 can be formulated as a MinRank instance using the idea outlined in the proof of theorem 4. The set of solutions to the subproblem is $\mathsf{MR}(kp, m, n, p, \mathbb{F}_q; \boldsymbol{e}'; \boldsymbol{B}_{1,1}, \dots, \boldsymbol{B}_{k,p})$, where

$$\boldsymbol{B}_{j,l} := \begin{bmatrix} \boldsymbol{0}_{m \times (k-j-1)} & \boldsymbol{\beta}_l^{\top} & \boldsymbol{0}_{m \times j} \end{bmatrix}.$$

More formally, consider $\boldsymbol{e}''$ written as a matrix over $\mathbb{F}_q$:

$$\begin{aligned} \boldsymbol{e}'' &= \begin{bmatrix} \sum_{l=1}^{p} a_{1,l} \boldsymbol{\beta}_l & \cdots & \sum_{l=1}^{p} a_{k,l} \boldsymbol{\beta}_l \end{bmatrix} \\ &= \sum_{j=1}^{k} \sum_{l=1}^{p} a_{j,l} \boldsymbol{B}_{j,l}, \end{aligned}$$

Table 4.1.: $p_{\mathsf{LB}}(p; 32, 32, 16, \mathbb{F}_2, 8)$ for different values of p

p	$p_{\mathsf{LB}}(p; 32, 32, 16, \mathbb{F}_2, 8)$
0	0.00
1	0.00
2	0.00
3	0.00
4	2.00×10^{-8}
5	4.13×10^{-6}
6	4.32×10^{-4}
7	0.03
8	0.97

with $\boldsymbol{B}_{j,l}$ as given above. This, along with the fact that we can take $\boldsymbol{e}'$ as our "base" matrix ($\boldsymbol{M}_0$ in definition 15) leads to the MinRank instance above.

The set of solutions can be retrieved in time $\mathcal{O}(kp(\lceil kp/m \rceil m)^2 q^{\lceil kp/m \rceil p})$ using the kernel attack (section 4.2.1). In order to fully evaluate the complexity of algorithm 4.2, we need to determine the probability of a successful iteration. This probability is formally defined in definition 16.

Definition 16 (Probability of submatrix having given rank)**.** Let $\boldsymbol{X} \in \mathbb{F}_q^{m \times n}$ be a matrix with $\operatorname{rank} \boldsymbol{X} = w$. Given a set of coordinates $\mathcal{J} \subset \{1, \ldots, n\}$ with $|\mathcal{J}| = k \leq n$, what is the probability that $\operatorname{rank} \boldsymbol{X}_{\mathcal{J}} = p$? We denote this probability by

$$p_{\mathsf{LB}}(p; m, n, k, \mathbb{F}_q, w).$$

Remark. Note that a solution for $p_{\mathsf{LB}}(p; m, n, k, \mathbb{F}_q, w)$ would have to average over all possible inputs. As an example, consider the two matrices

$$\begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix} \quad \text{and} \quad \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}.$$

Both matrices are of rank 3 – however, if one were to pick a 3×3 submatrix at random, the probability that the submatrix is also of rank 3 is higher in the first case than in the second. This is so because $\left[\begin{smallmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{smallmatrix}\right]$, $\left[\begin{smallmatrix} 1 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 1 & 1 \end{smallmatrix}\right]$, $\left[\begin{smallmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 0 & 0 & 1 \end{smallmatrix}\right]$, $\left[\begin{smallmatrix} 0 & 0 & 1 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \end{smallmatrix}\right]$, $\left[\begin{smallmatrix} 0 & 1 & 1 \\ 0 & 0 & 1 \\ 1 & 0 & 1 \end{smallmatrix}\right]$, and $\left[\begin{smallmatrix} 0 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \end{smallmatrix}\right]$ are all rank-3 submatrices of the first matrix, but are not submatrices at all of the second matrix.

Table 4.1 shows the value of $p_{\mathsf{LB}}(p; 32, 32, 16, \mathbb{F}_2, 8)$ for different values of p. The probability was determined empirically by 6×10^8 simulations using SageMath. The source code can be seen in listing A.1.

Knowing that the number of subspaces of dimension p in a vector space of dimension m over $\mathbb{F}_q$ is given by

$$\begin{bmatrix} m \\ p \end{bmatrix}_q = \begin{cases} \frac{(1-q^m)(1-q^{m-1})\cdots(1-q^{m-r+1})}{(1-q)(1-q^2)\cdots(1-q^r)} & \text{if } p \leq m \\ 0 & \text{if } p > m, \end{cases}$$

we see from theorem 5 that the complexity of algorithm 4.2 satisfies

$$\mathcal{O}\left(p_{\mathsf{LB}}(p; m, n, k, \mathbb{F}_q, w)^{-1} \begin{bmatrix} m \\ p \end{bmatrix}_q kp(\lceil kp/m \rceil m)^2 q^{\lceil kp/m \rceil p} \right). \tag{4.4}$$

Instead of solving a MinRank subproblem on line 9 by iterating over all p-dimensional subspaces, one can iterate over all weight-p — i.e., rank-p — matrices $\boldsymbol{e}''$ and check if

$$\|\boldsymbol{e}' - \boldsymbol{e}'' \boldsymbol{G}_{\mathcal{J}}^{-1} \boldsymbol{G}\|_q \leq w$$

immediately. In order to evaluate the complexity of such a variant of the algorithm, we need theorem 6 and corollary 3.

Theorem 6. *The number of surjective linear transformations from an n-dimensional vector space $V_n(q)$ to an m-dimensional vector space over $\mathbb{F}_q$ is*

$$\sum_{k=0}^{m} (-1)^{m-k} \begin{bmatrix} m \\ k \end{bmatrix}_q q^{nk + \binom{m-k}{2}}.$$

Proof. See [22, p. 338]. □

Corollary 3. *The number of $k \times n$ matrices over $\mathbb{F}_q$ that have rank p is*

$$N_q(k, n, p) := \begin{bmatrix} n \\ p \end{bmatrix}_q \sum_{l=0}^{p} (-1)^{r-l} \begin{bmatrix} r \\ l \end{bmatrix}_q q^{kl + \binom{r-l}{2}}.$$

Corollary 3 implies that an algorithm which iterates through all rank-p matrices would have a complexity given by eq. (4.5). However, we found out numerically that this quantity is always larger than the one given by eq. (4.4).

$$\mathcal{O}\left(p_{\mathsf{LB}}(p; m, n, k, \mathbb{F}_q, w)^{-1} N_q(m, k, p) \right). \tag{4.5}$$

Thoughts on the Ideas by Stern [35] and Dumer [6]

As opposed to Lee–Brickell's algorithm which decodes p errors in the information set by brute force, Stern–Dumer's algorithm partitions the information set into two subsets of size $k/2$ each, and allows for p errors within either of them. This has the effect of splitting the enumeration into two parts of size $k/2$ each.

However, the most notable improvement of Stern–Dumer's algorithm is the incorporation of the idea by Leon [21] to restrict the number of possible candidates for the error word $\boldsymbol{e}$ to those vectors having $l \in \{0, \dots, n-k\}$ zeroes in positions outside of the $\mathcal{J}$-indexed columns. Thus, the main difference to Lee–Brickell's algorithm is that Stern–Dumer's algorithm tries to build the weight-w error word by first looking for collisions among sums of l columns restricted to certain positions and then checks the weight of the column sums arising from the collisions.

Unfortunately, we already noted that the probability of having zero entries is overwhelmingly small when errors are chosen according to their rank weight rather than Hamming weight. This makes Stern–Dumer's algorithm unsuitable for the task, similarly to Plain-ISD.

4.3.3. Algorithms Using Projections in the Rank Metric

In the Hamming metric, the notion of *error support* is used to formally define the probability of success of information set decoding algorithms. We saw that the error support, $\mathrm{supp}(\boldsymbol{e})$, is given by the coordinates of the nonzero entries in $\boldsymbol{e}$, i.e., the entries contributing to the overall Hamming weight.

It is a well-known fact from linear algebra that a vector $x \in \mathbb{F}_{q^m}^n$ with $\|\boldsymbol{x}\|_q \leq p$ can be represented as $\boldsymbol{x} = \begin{bmatrix} x_1 & \cdots & x_n \end{bmatrix} = \begin{bmatrix} x'_1 & \cdots & x'_p \end{bmatrix} \boldsymbol{P}$, where $x'_j \in \mathbb{F}_{q^m}$ and $\boldsymbol{P}$ is a $p \times n$ matrix over $\mathbb{F}_q$ of full rank. The concept of *elementary linear subspaces* can be introduced as a consequence of this representation.

Definition 17 (Elementary linear subspace)**.** A subspace $\mathcal{V} \leq \mathbb{F}_{q^m}^n$ is said to be *elementary* if it has a basis B consisting of row vectors in $\mathbb{F}_q^n$. B is called an elementary basis of $\mathcal{V}$. For $0 \leq v \leq n$, we define $E_p(q^m, n)$ as the set of all ELS's with dimension p in $\mathbb{F}_{q^m}^n$.

Gadouleau and Yan [12] formally defined this concept and showed the connection between *sets of coordinates* in the Hamming metric and *elementary linear subspaces* in the rank metric. Following the notion of elementary linear subspaces, we can define $\mathrm{supp}(\boldsymbol{e})$ in the rank metric to be the $\mathbb{F}_q$-linear subspace of $\mathbb{F}_q^m$ spanned by the columns of $\boldsymbol{e}$. Now, what we would like to do is project the columns of $\boldsymbol{e}$ onto a subspace that we hope would be orthogonal or "near-orthogonal" to $\mathrm{supp}(\boldsymbol{e})$. This is highly desired because projecting the problem onto such a subspace would have the effect of eliminating $\boldsymbol{e}$ from the picture. The subsequent algorithms have this idea at their core.

Plain Information Set Decoding

Informally, we are interested in choosing a subspace of $\mathbb{F}_q^m$ would:

Condition 1. Be orthogonal or "near-orthogonal" to $\mathrm{supp}(\boldsymbol{e})$; and

Condition 2. Allow us to solve for the message $\boldsymbol{x} \in \mathbb{F}_{q^m}^k$.

We know that — in full generality — $e_i \in \mathbb{F}_q^m$ for all $1 \leq i \leq n$. Let $\mathcal{V}$ be a ζ-dimensional subspace of $\mathbb{F}_q^m$. Now, assume that it is also true that $e_i \in \mathcal{V}^\perp$ for all $1 \leq i \leq n$. (In other words, $\mathcal{V}$ satisfies condition 1.) We can write

$$\begin{cases} \boldsymbol{P}(y_1 - (\boldsymbol{xG})_1) &= \boldsymbol{0} \\ \boldsymbol{P}(y_2 - (\boldsymbol{xG})_2) &= \boldsymbol{0} \\ & \vdots \\ \boldsymbol{P}(y_n - (\boldsymbol{xG})_n) &= \boldsymbol{0}, \end{cases} \tag{4.6}$$

where $\boldsymbol{P}$ is a $\zeta \times m$ full-rank which projects from $\mathbb{F}_q^m$ onto $\mathcal{V}$, and $\boldsymbol{x}$ is a "matrix" of mk unknowns – $x_{1,1}, \ldots, x_{m,1}, \ldots, x_{1,k}, \ldots, x_{m,k} \in \mathbb{F}_q$. When looked as a linear system over $\mathbb{F}_q$, eq. (4.6) is a system of ζn equations in mk unknowns. Since we would like these numbers to be as close as possible but avoid the case of having more unknowns than equations, we take $\zeta := \lceil mk/n \rceil$. In other words, taking $\zeta := \lceil mk/n \rceil$ allows us to satisfy condition 2 simultaneously.

It is instructional to compare this approach to the kernel attack on MinRank (section 4.2.1). In the kernel attack, we are given the right-hand side of each equality of eq. (4.1) and are solving for the left-hand side; here, we are given the left-hand side of each equality of eq. (4.6) and are solving for the right-hand side. We can translate this idea to an algorithm in a straightforward manner, algorithm 4.3.

Analyzing the complexity of algorithm 4.3 requires us to define what "sufficiently many iterations" on line 5 means. We know that there are $\begin{bmatrix} m \\ w \end{bmatrix}_q$ possibilities for $\mathrm{supp}(\boldsymbol{e})$ in the general case. On the other hand, there are $\begin{bmatrix} m-\zeta \\ w \end{bmatrix}_q$ subspaces satisfying condition 1. (Recall that condition 2 is satisfied by the choice of ζ.) This implies that the probability of a successful iteration is

$$\begin{bmatrix} m-\zeta \\ w \end{bmatrix}_q \begin{bmatrix} m \\ w \end{bmatrix}_q^{-1};$$

which, in turn, means that on average

$$\begin{bmatrix} m-\zeta \\ w \end{bmatrix}_q^{-1} \begin{bmatrix} m \\ w \end{bmatrix}_q$$

iterations have to be performed (theorem 5). Solving for $\boldsymbol{x}$ can be done in time polynomial in the inputs to the algorithm, as it is a simple linear system over $\mathbb{F}_q$.

Algorithm 4.3: PLAIN-ISD using projections in the rank metric

Input: Generator matrix $\boldsymbol{G} \in \mathbb{F}_{q^m}^{k \times n}$, received word $\boldsymbol{y} \in \mathbb{F}_q^n$, and target weight $w \in \mathbb{N}$
Output: Error $\boldsymbol{e} \in \mathbb{F}_q^n$ with $\|\boldsymbol{e}\| \leq w$ and $\boldsymbol{y} - \boldsymbol{e} \in \mathcal{C}$

begin
  $\zeta \leftarrow \lceil \frac{mk}{n} \rceil$
  $\boldsymbol{x} := \begin{bmatrix} x_1 & \cdots & x_k \end{bmatrix}$ with $x_j \in \mathbb{F}_q^m$
  $x_j := \begin{bmatrix} x_{j,1} & \cdots & x_{j,m} \end{bmatrix}^\top$ with $x_{j,l} \in \mathbb{F}_q$
  repeat
    $\mathcal{V} \leftarrow$ random ζ-dimensional subspace of $\mathbb{F}_q^m$
    $\boldsymbol{P} \leftarrow \zeta \times m$ matrix that projects from $\mathbb{F}_q^m$ to $\mathcal{V}$
    solve for $\boldsymbol{x}$ in $\{\boldsymbol{P}(y_i - (\boldsymbol{x}\boldsymbol{G})_i) = \boldsymbol{0} : \forall 1 \leq i \leq n\}$
    forall *solutions for* $\boldsymbol{x}$ **do**
      $\boldsymbol{e} \leftarrow \boldsymbol{y} - \boldsymbol{x}\boldsymbol{G}$ with the appropriate values substituted in for $\boldsymbol{x}$
      if $\|\boldsymbol{e}\|_q \leq w$ **then return** $\boldsymbol{e}$
    end forall
  until *sufficiently many iterations have been performed*
end

The Lee–Brickell Algorithm

Bearing in mind the notion of $\operatorname{supp}(\boldsymbol{e})$, we can adapt the idea of Lee and Brickell [20] from the Hamming metric to the rank metric. Informally, we can allow for $\operatorname{supp}(\boldsymbol{e})$ and $\mathcal{V}$ to be "near-orthogonal", i.e., for a p-dimensional intersection between them. The dimension of the intersection is determined by a parameter $p \in \mathbb{N}$ with $0 \leq p \leq w$. PLAIN-ISD corresponds directly to $p = 0$.

Allowing for a nontrivial intersection has the effect of increasing the probability of a successful iteration. Namely, let $\operatorname{supp}(\boldsymbol{e}) \cap \mathcal{V} = \tilde{\mathcal{V}}$, with $\tilde{\mathcal{V}}$ being a p-dimensional subspace of $\mathcal{V}$. Now, "p dimensions" of $\operatorname{supp}(\boldsymbol{e})$ come from $\tilde{\mathcal{V}}$, while the remaining "$w-p$ dimensions" come from $\mathcal{V}^\perp$. Again, there are in total $\begin{bmatrix} m \\ w \end{bmatrix}_q$ possibilities for $\operatorname{supp}(\boldsymbol{e})$. This means that the probability of a successful iteration is now

$$\begin{bmatrix} \zeta \\ p \end{bmatrix}_q \begin{bmatrix} m-\zeta \\ w-p \end{bmatrix}_q \begin{bmatrix} m \\ w \end{bmatrix}_q^{-1}. \tag{4.7}$$

Let $\langle \tilde{\boldsymbol{v}}_1, \ldots, \tilde{\boldsymbol{v}}_p \rangle$ be a basis of $\tilde{\mathcal{V}}$. Equation (4.6) now becomes

$$\begin{cases} \boldsymbol{P}(y_1 - (\boldsymbol{x}\boldsymbol{G})_1) &= \sum_{j=1}^p \lambda_{1,j} \tilde{\boldsymbol{v}}_j \\ \boldsymbol{P}(y_2 - (\boldsymbol{x}\boldsymbol{G})_2) &= \sum_{j=1}^p \lambda_{2,j} \tilde{\boldsymbol{v}}_j \\ & \vdots \\ \boldsymbol{P}(y_n - (\boldsymbol{x}\boldsymbol{G})_n) &= \sum_{j=1}^p \lambda_{n,j} \tilde{\boldsymbol{v}}_j. \end{cases} \tag{4.8}$$

There are an additional np unknowns now; thus, we require that $\zeta = \lceil mk/n \rceil + p$. Algorithm 4.4 is an immediate consequence of eq. (4.8).

Algorithm 4.4: LB-ISD using projections in the rank metric

Input: Generator matrix $\boldsymbol{G} \in \mathbb{F}_{q^m}^{k \times n}$, received word $\boldsymbol{y} \in \mathbb{F}_q^n$, target weight $w \in \mathbb{N}$, and algorithm parameter $p \in \mathbb{N}$ with $0 \leq p \leq w$
Output: Error $\boldsymbol{e} \in \mathbb{F}_q^n$ with $\|\boldsymbol{e}\| \leq w$ and $\boldsymbol{y} - \boldsymbol{e} \in \mathcal{C}$
begin
  $\zeta \leftarrow \left\lceil \frac{mk}{n} \right\rceil + p$
  $\boldsymbol{x} := \begin{bmatrix} x_1 & \cdots & x_k \end{bmatrix}$ with $x_j \in \mathbb{F}_q^m$
  $x_j := \begin{bmatrix} x_{j,1} & \cdots & x_{j,m} \end{bmatrix}^\top$ with $x_{j,l} \in \mathbb{F}_q$
  repeat
    $\mathcal{V} \leftarrow$ random ζ-dimensional subspace of $\mathbb{F}_q^m$
    $\boldsymbol{P} \leftarrow \zeta \times m$ matrix that projects from $\mathbb{F}_q^m$ to $\mathcal{V}$
    foreach $\tilde{\mathcal{V}} \leq \mathcal{V}$ *with* $\dim \tilde{\mathcal{V}} = p$ **do**
      $\langle \tilde{\boldsymbol{v}}_1, \ldots, \tilde{\boldsymbol{v}}_p \rangle \leftarrow$ basis of $\tilde{\mathcal{V}}$
      solve for $\boldsymbol{x}$ and $\{\lambda_{i,j}\}$ in $\{\boldsymbol{P}(y_i - (\boldsymbol{x}\boldsymbol{G})_i) = \sum_{j=1}^{p} \lambda_{i,j} \tilde{\boldsymbol{v}}_j : \forall 1 \leq i \leq n\}$
      forall *solutions for* $\boldsymbol{x}$ *and* $\lambda_{i,j}$ **do**
        $\boldsymbol{e} \leftarrow \boldsymbol{y} - \boldsymbol{x}\boldsymbol{G}$ with the appropriate values for $\boldsymbol{x}$ substituted in
        if $\|\boldsymbol{e}\|_q \leq w$ **then return** $\boldsymbol{e}$
      end forall
    end foreach
  until *sufficiently many iterations have been performed*
end

Asymptotic analysis of algorithm 4.4 The probability of success given by eq. (4.7), along with the cost of iterating through all p-dimensional subspaces of a ζ-dimensional vector space implies — ignoring the cost of Gaussian elimination — a complexity of

$$\begin{bmatrix} m - \zeta \\ w - p \end{bmatrix}_q^{-1} \begin{bmatrix} m \\ w \end{bmatrix}_q .$$

Recall that the Gaussian binomial coefficient satisfies

$$\begin{bmatrix} n \\ k \end{bmatrix}_q \in \Theta(q^{k(n-k)}).$$

This means that, asymptotically, the complexity is

$$\Theta(q^{w\zeta + p(p+m-\zeta)}).$$

Table 4.2.: Complexity of LB-ISD for the parameters proposed in [23]

m	n	k	λ	Compl. of [11] (bit)	Opt. p	Compl. of algorithm 4.3 (bit)
50	50	32	3	96	0	96
80	80	11	3	121	0	121
96	64	40	3	240	0	240
128	90	24	3	352	0	352
128	120	80	5	320	0	320

Since $m > \zeta$, the above expression is minimized when $p = 0$. Thus, it is always optimal to take $p = 0$, i.e., the case of Plain-ISD. In that case, the above expression becomes

$$\Theta(q^{w\lceil mk/n \rceil}),$$

which can be seen as dual to the state-of-the-art algorithms by Gaborit, Ruatta, and Schrek [11] and Hauteville and Tillich [17]. We see from table 4.2 that Plain-ISD has the same asymptotic complexity as the algorithms in [11, 17]. Intuitively, this is because

$$\begin{bmatrix} n \\ k \end{bmatrix}_q = \begin{bmatrix} n \\ n-k \end{bmatrix}_q.$$

5. Conclusion and Future Perspective

In this thesis, we surveyed the GPT cryptosystem and proposed new techniques for decoding in the rank metric. These techniques are based on the concept of *information set decoding*, which has been studied extensively in the Hamming metric. We show that these techniques can be seen as dual to the state-of-the-art algorithms ([11, 17]) and achieve the same complexity.

In chapter 2, we defined the GPT cryptosystem in its original form and the subsequent reparation by Gabidulin and Ourivski [9]. We simplified and incorporated additional ideas into the proof of Kshevetskiy [19] which shows that we can always assume a certain form of a GPT public key suitable for cryptanalysis.

The main contributions are the algorithms in chapter 4. We formulated variants of plain information set decoding and Lee–Brickell's algorithm using a notion of error support suitable for the rank metric. Prior to that, we surveyed the connection between the MinRank problem in linear algebra and decoding in the rank metric. Additionally, the direct adaptation of information set decoding from the Hamming metric to the rank metric was experimentally found to be inefficient. However, it served as the basis for the algorithms in section 4.3.3, which are tailored specifically for the rank metric.

We leave it as future work to

- Adapt improvements in the spirit of Stern [35] and Dumer [6] to the notion of error support in the rank metric; and
- Provide working implementations of the algorithms in SageMath [4].

A. Source code

The SageMath [4] code shown in listing A.1 estimates the empirical distribution of p_{LB} defined in definition 16.

```
from collections import Counter

def submatrix_rank_simulation(m, n, k, KK, w,
                              num_outer=10^4, num_inner=10^4):
    #  Simulate the distribution of the rank of a random submatrix.

    #  `num_outer` matrices from M(K,m,n) of rank w are sampled.
    #  For each rank-w matrix, `num_outer` m×k random submatrices
    #  are taken and have their ranks tallied.
    if k > n:
        raise ValueError("`k` must not be greater than `n`")

    tally = Counter()
    col_indices = tuple(range(n))

    for i in range(num_outer):
        E = random_matrix(KK, m, n,
                          algorithm='echelonizable', rank=w)
        for _ in range(num_inner):
            E_sub = E[:, sample(col_indices, k)]
            tally[E_sub.rank()] += 1

    return tally
```

Listing A.1.: Estimating $p_{\mathsf{LB}}(p; 32, 32, 16, \mathbb{F}_2, 8)$ empirically

Bibliography

[1] S. Barg. "Some New NP-Complete Coding Problems". English. In: *Probl. Inf. Transm.* 30.3 (1994), pp. 209–214. ISSN: 0032-9460; 1608-3253/e (cit. on p. 6).

[2] A. Becker et al. *Decoding Random Binary Linear Codes in $2^{n/20}$: How $1 + 1 = 0$ Improves Information Set Decoding.* Cryptology ePrint Archive, Report 2012/026. Version 20120120:200144. Jan. 2012. eprint: https://eprint.iacr.org/2012/026 (cit. on pp. 3, 22).

[3] E. Berlekamp, R. McEliece, and H. van Tilborg. "On the Inherent Intractability of Certain Coding Problems". In: *IEEE Transactions on Information Theory* 24.3 (May 1978), pp. 384–386. ISSN: 0018-9448. DOI: 10.1109/TIT.1978.1055873 (cit. on p. 6).

[4] T. S. Developers. *SageMath, the Sage Mathematics Software System (Version 8.1).* 2017. URL: http://www.sagemath.org (cit. on pp. 36, 37).

[5] W. Diffie and M. Hellman. "New Directions in Cryptography". In: *IEEE Trans. Inf. Theor.* 22.6 (Nov. 1976), pp. 644–654. ISSN: 0018-9448. DOI: 10.1109/TIT.1976.1055638 (cit. on p. 2).

[6] I. Dumer. "Suboptimal Decoding of Linear Codes: Partition Technique". In: *IEEE Transactions on Information Theory* 42.6 (Nov. 1996), pp. 1971–1986. ISSN: 0018-9448. DOI: 10.1109/18.556688 (cit. on pp. 27, 31, 36).

[7] J.-C. Faugère, F. Levy-dit-Vehel, and L. Perret. "Cryptanalysis of MinRank". In: *Advances in Cryptology – CRYPTO 2008: 28th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2008. Proceedings.* Ed. by D. Wagner. Berlin, Heidelberg: Springer, 2008, pp. 280–296. ISBN: 978-3-540-85174-5. DOI: 10.1007/978-3-540-85174-5_16 (cit. on p. 25).

[8] E. M. Gabidulin. "Theory of Codes with Maximum Rank Distance". In: *Probl. Inf. Transm.* 21.1 (1985), pp. 3–16 (cit. on pp. 8, 10).

[9] E. M. Gabidulin and A. V. Ourivski. "Modified GPT PKC with Right Scrambler". In: *Electronic Notes in Discrete Mathematics* 6.Supplement C (2001). WCC2001, International Workshop on Coding and Cryptography, pp. 168–177. ISSN: 1571-0653. DOI: 10.1016/S1571-0653(04)00168-4 (cit. on pp. 14, 15, 36).

[10] E. M. Gabidulin, A. V. Paramonov, and O. V. Tretjakov. "Ideals over a Non-Commutative Ring and Their Application in Cryptology". In: *Proceedings of the 10th Annual International Conference on Theory and Application of Cryptographic Techniques.* EUROCRYPT'91. Brighton, UK: Springer-Verlag, 1991, pp. 482–489. ISBN: 3-540-54620-0 (cit. on pp. 3, 8, 14).

[11] P. Gaborit, O. Ruatta, and J. Schrek. "On the Complexity of the Rank Syndrome Decoding Problem". In: *CoRR* abs/1301.1026 (2013). arXiv: 1301.1026. URL: http://arxiv.org/abs/1301.1026 (cit. on pp. 35, 36).

[12] M. Gadouleau and Z. Yan. "On the Decoder Error Probability of Bounded Rank-Distance Decoders for Maximum Rank-Distance Codes". In: *IEEE Transactions on Information Theory* 54.7 (July 2008), pp. 3202–3206. ISSN: 0018-9448. DOI: 10.1109/TIT.2008.924697 (cit. on p. 31).

[13] M. Gadouleau and Z. Yan. "Optimal Distortion Parameter for the GPT Public-Key Cryptosystem". In: *IEEE/Sarnoff Symposium on Advances in Wired and Wireless Communication, 2005.* Apr. 2005, pp. 133–136. DOI: 10.1109/SARNOF.2005.1426530 (cit. on p. 15).

[14] J. K. Gibson. "Severely denting the Gabidulin version of the McEliece Public Key Cryptosystem". In: *Designs, Codes and Cryptography* 6.1 (July 1995), pp. 37–45. ISSN: 1573-7586. DOI: 10.1007/BF01390769 (cit. on pp. 14, 15).

[15] J. K. Gibson. "The Security of the Gabidulin Public Key Cryptosystem". In: *Advances in Cryptology — EUROCRYPT '96: International Conference on the Theory and Application of Cryptographic Techniques Saragossa, Spain, May 12–16, 1996 Proceedings.* Ed. by U. Maurer. Berlin, Heidelberg: Springer, 1996, pp. 212–223. ISBN: 978-3-540-68339-1. DOI: 10.1007/3-540-68339-9_19 (cit. on pp. 14, 15).

[16] L. Goubin and N. T. Courtois. "Cryptanalysis of the TTM Cryptosystem". In: *Advances in Cryptology — ASIACRYPT 2000: 6th International Conference on the Theory and Application of Cryptology and Information Security Kyoto, Japan, December 3–7, 2000 Proceedings.* Ed. by T. Okamoto. Berlin, Heidelberg: Springer, 2000, pp. 44–57. ISBN: 978-3-540-44448-0. DOI: 10.1007/3-540-44448-3_4 (cit. on p. 23).

[17] A. Hauteville and J. Tillich. "New Algorithms for Decoding in the Rank Metric and an Attack on the LRPC Cryptosystem". In: *CoRR* abs/1504.05431 (2015). arXiv: 1504.05431. URL: http://arxiv.org/abs/1504.05431 (cit. on pp. 35, 36).

[18] A. Kipnis and A. Shamir. "Cryptanalysis of the HFE Public Key Cryptosystem by Relinearization". In: *Advances in Cryptology — CRYPTO' 99: 19th Annual International Cryptology Conference Santa Barbara, California, USA, August 15–19, 1999 Proceedings.* Ed. by M. Wiener. Berlin, Heidelberg: Springer, 1999, pp. 19–30. ISBN: 978-3-540-48405-9. DOI: 10.1007/3-540-48405-1_2. URL: https://doi.org/10.1007/3-540-48405-1_2 (cit. on p. 24).

[19] A. Kshevetskiy. "Security of GPT-Like Public-Key Cryptosystems Based on Linear Rank Codes". In: *3rd International Workshop on Signal Design and Its Applications in Communications.* Sept. 2007, pp. 143–147. DOI: 10.1109/IWSDA.2007.4408344 (cit. on pp. 15, 36).

[20] P. J. Lee and E. F. Brickell. "An Observation on the Security of McEliece's Public-Key Cryptosystem". In: *Advances in Cryptology — EUROCRYPT '88: Workshop on the Theory and Application of Cryptographic Techniques Davos, Switzerland, May 25–27, 1988 Proceedings*. Ed. by D. Barstow et al. Berlin, Heidelberg: Springer, 1988, pp. 275–280. ISBN: 978-3-540-45961-3. DOI: 10.1007/3-540-45961-8_25 (cit. on pp. 27, 28, 33).

[21] J. S. Leon. "A Probabilistic Algorithm for Computing Minimum Weights of Large Error-Correcting Codes". In: *IEEE Transactions on Information Theory* 34.5 (Sept. 1988), pp. 1354–1359. ISSN: 0018-9448. DOI: 10.1109/18.21270 (cit. on p. 31).

[22] J. H. van Lint and R. M. Wilson. *A Course in Combinatorics*. 2nd ed. Cambridge University Press, 2001. 602 pp. ISBN: 9780521006019 (cit. on p. 30).

[23] P. Loidreau. *A New Rank Metric Codes Based Encryption Scheme*. Cryptology ePrint Archive, Report 2017/236. Version 20170311:144514. Mar. 2017. eprint: https://eprint.iacr.org/2017/236 (cit. on pp. iv, 20, 35).

[24] R. J. McEliece. *A Public-Key Cryptosystem Based on Algebraic Coding Theory*. Tech. rep. 44. Jet Propulsion Lab., CA, 1978, pp. 114–116 (cit. on pp. 2, 22).

[25] R. Overbeck. "A New Structural Attack for GPT and Variants". In: *Progress in Cryptology — Mycrypt 2005: First International Conference on Cryptology in Malaysia, Kuala Lumpur, Malaysia, September 28–30, 2005. Proceedings*. Ed. by E. Dawson and S. Vaudenay. Berlin, Heidelberg: Springer, 2005, pp. 50–63. ISBN: 978-3-540-32066-1. DOI: 10.1007/11554868_5 (cit. on pp. 14, 17).

[26] R. Overbeck. "Extending Gibson's Attacks on the GPT Cryptosystem". In: *Coding and Cryptography: International Workshop, WCC 2005, Bergen, Norway, March 14–18, 2005. Revised Selected Papers*. Ed. by Ø. Ytrehus. Berlin, Heidelberg: Springer, 2006, pp. 178–188. ISBN: 978-3-540-35482-6. DOI: 10.1007/11779360_15 (cit. on pp. 14, 17).

[27] R. Overbeck. "Structural Attacks for Public-Key Cryptosystems Based on Gabidulin Codes". In: *Journal of Cryptology* 21.2 (Apr. 2008), pp. 280–301. ISSN: 1432-1378. DOI: 10.1007/s00145-007-9003-9 (cit. on pp. 14, 17).

[28] C. P. Peters. "Curves, Codes, and Cryptography". Dissertation. Technische Universiteit Eindhoven, 2011. ISBN: 978-90-386-2476-1. DOI: 10.6100/IR711052. eprint: http://repository.tue.nl/8060eb20-09dc-437a-b5d7-8faae3ccc447 (cit. on p. 27).

[29] E. Prange. "The Use of Information Sets in Decoding Cyclic Codes". In: *IRE Transactions on Information Theory* 8.5 (Sept. 1962), pp. 5–9. ISSN: 0096-1000. DOI: 10.1109/TIT.1962.1057777 (cit. on pp. 22, 26).

[30] R. L. Rivest, A. Shamir, and L. Adleman. "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems". In: *Commun. ACM* 21.2 (Feb. 1978), pp. 120–126. ISSN: 0001-0782. DOI: 10.1145/359340.359342 (cit. on p. 2).

[31] B. Schneier. *Applied Cryptography. Protocols, Algorithms, and Source Code in C*. 2nd ed. John Wiley & Sons, 1996. 758 pp. ISBN: 978-1-119-09672-6 (cit. on p. 2).

[32] C. E. Shannon. "A Mathematical Theory of Communication". In: *Bell System Technical Journal* 27.3 (1948), pp. 379–423. ISSN: 1538-7305. DOI: 10.1002/j.1538-7305.1948.tb01338.x (cit. on p. 4).

[33] P. W. Shor. "Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer". In: *SIAM J. Comput.* 26.5 (Oct. 1997), pp. 1484–1509. ISSN: 0097-5397. DOI: 10.1137/S0097539795293172 (cit. on p. 2).

[34] R. Singleton. "Maximum Distance q-nary Codes". In: *IEEE Transactions on Information Theory* 10.2 (Apr. 1964), pp. 116–118. ISSN: 0018-9448. DOI: 10.1109/TIT.1964.1053661 (cit. on p. 10).

[35] J. Stern. "A Method for Finding Codewords of Small Weight". In: *Coding Theory and Applications: 3rd International Colloquium Toulon, France, November 2–4, 1988 Proceedings*. Ed. by G. Cohen and J. Wolfmann. Berlin, Heidelberg: Springer, 1989, pp. 106–113. ISBN: 978-3-540-46726-7. DOI: 10.1007/BFb0019850 (cit. on pp. 27, 31, 36).

[36] A. Wachter, V. Sidorenko, and M. Bossert. "A Fast Linearized Euclidean Algorithm for Decoding Gabidulin Codes". In: *Proceedings of the 12th International Workshop on Algebraic and Combinatorial Coding Theory (ACCT)*. Sept. 2010, pp. 298–303 (cit. on p. 13).