Using structured algebraic geometry codes in modern cryptography

Mathieu Lhotel

THESE DE DOCTORAT DE L'ETABLISSEMENT UNIVERSITE BOURGOGNE FRANCHE-COMTE

PREPAREE A L'Université de Franche-Comté

Ecole doctorale n°553
Carnot–Pasteur

Doctorat de Mathématiques

Par
Mr Lhotel Mathieu

Using structured algebraic geometry codes in modern cryptography

Utilisation de codes de géométrie algébrique structurés en cryptographie moderne

---

Thèse présentée et soutenue à Besançon, le 3 Juillet 2023

Composition du Jury :

M. Daniel Augot	Directeur de recherche, Inria Saclay (LIX)	Président
M. Marc Perret	Professeur, Université de Toulouse II	Rapporteur
M. Alain Couvreur	Directeur de recherche, Inria Saclay (LIX)	Rapporteur
Mme Christine Huyghe	Directrice de recherche, Université de Franche-Comté	Examinatrice
M. Philippe Lebacque	Maître de conférence, Université de Franche-Comté	Directeur de thèse
M. Hugues Randriambololona	Expert en cryptographie, ANSSI	Co-directeur de thèse

Remerciements

Tout d'abord, je tiens à remercier l'ensemble des membres du jury pour avoir accordé du temps à mes travaux. Plus particulièrement, je remercie Marc et Alain pour avoir accepté d'en rédiger un rapport ainsi que pour les corrections qu'ils m'ont suggérées. En dehors de leur participation au jury, je les remercie pour leurs nombreux conseils et explications.

Mes prochains remerciements vont à mes deux directeurs de thèse, Philippe et Hugues, qui ont permis à cette thèse de voir le jour. Malgré des moments difficiles, dus à la distance ou encore à la Covid-19, ils ont réussi à garder ces années motivantes pour moi. Je remercie Philippe, qui m'accompagne depuis mes projets de master et qui a suffisamment cru en moi pour poursuivre cette aventure. Je le remercie également pour nos discussions intéressantes et nos parties sur la borne d'arcade. Je remercie Hugues d'avoir accepté de co–encadrer cette thèse et pour ses précieux conseils et idées.

J'exprime mes plus chaleureux remerciements à Jade, sans qui cette thèse aurait eu beaucoup moins de saveur. Je la remercie pour nos discussions, tant professionnelles que personnelles. Je ne saurais énumérer tous les moments où nous avons improvisé une visio pour faire de longs calculs (pas toujours intéressant). Je suis persuadé que ses futurs étudiants seront plus que satisfaits de son encadrement.

Je souhaite également remercier tous les membres du projet Barracuda, qui m'ont offert de nombreuses opportunités de partager mon travail. Au même titre, je remercie les habitués du groupe de travail Code–Crypto, qui m'ont permis de rencontrer de nombreuses personnes travaillant sur les mêmes sujets que moi. Cela s'est révélé très bénéfique, surtout en début de thèse.

Je remercie tous les membres de Laboratoire de Mathématiques de Besançon, tant l'équipe de théorie des nombres que le personnel administratif ou encore l'équipe informatique, pour m'avoir offert un cadre de travail accueillant. Ils ont toujours su répondre efficacement à mes demandes et interrogations. Je pense aussi à mes collègues doctorants, avec qui j'ai pu échanger de nombreuses fois lors de séminaires.

Je tiens à remercier tout particulièrement deux doctorants qui ont commencé cette aventure en même temps que moi, Audrey et Charles. Leur présence à mes côtés durant ces trois années a été une réelle motivation pour moi. Avoir l'opportunité de donner mes premiers enseignements ou encore premières présentations en même temps qu'eux était très encourageant et bien moins stressant. Ces années auraient étés moins amusantes sans vous.

Un grand merci à mes amis qui m'ont soutenu et accompagné durant ces trois années (et pas seulement). Ils ont toujours été là, dans les bons moments comme dans les plus difficiles. Bien que de nombreux changements aient eu lieu au cours de ces trois dernières années, nous n'avons jamais perdu contact et j'espère que ce ne sera jamais le cas.

Mes plus profonds remerciements vont à ma famille, en particulier mes parents et mon frère. Ils m'ont toujours apporté leur soutien, si bien moral que financier, pendant mon cursus scolaire, surtout dans les moments les plus difficiles. Ils n'ont jamais perdu confiance en moi, bien qu'ils n'aient sûrement pas la moindre idée de ce que j'ai fait durant ces trois longues années ! Je remercie également Julien, pour ses conseils sur la langue anglaise, ainsi que pour son hospitalité dans des moments creux de la SNCF !

Enfin, merci à Clémence pour sa présence et son soutien, depuis plus d'un an maintenant. Merci d'être là pour m'écouter me plaindre de mes petits problèmes de doctorant.

Contents

Introduction

Context

From public key to post–quantum cryptography

In the area of cryptography, *public key cryptography* (or *asymmetric cryptography*) concerns schemes using a pair of keys: the *public key* which is known to anyone and the *secret key*, only known by the recipient of the message. The family of public key encryption schemes was introduced by Diffie and Hellman [DH76]. Before this, most of the systems used in cryptography were part of the so–called *secret key cryptography* (or *symmetric cryptography*), which uses the same key to encrypt and decrypt the message. In such systems, both parties have to agree on a secure way to share the key, which is no more the case while considering a pair of keys.

The first public key cryptosystem was introduced by Rivest, Shamir and Adleman [RSA78], and is known as the RSA cryptosystem. It is still widely used for secure data transmission, and its security mainly relies on the hardness of factorizing large integers into primes (in the sense that factorizing efficiently would broke the scheme). Since then, several other schemes have been proposed, whose security rely on other assumptions coming from number theory as well, such as the discrete logarithm problem. However, the recent threat of quantum computers leads to expand the area of cryptography: in fact, these number theoretical problems could be broken in polynomial time on a quantum computer, using Shor's algorithm [Sho94]. *Post–quantum cryptography* is the domain which regroups new cryptosystems, based on new assumptions that resist to quantum attacks. In this thesis, we are interested in *code–based cryptography*, which uses error–correcting codes. In this case, the security is based on the problem of decoding a random linear code, for which no quantum algorithm is known yet, reason why it is a good candidate in post–quantum cryptography.

Code–based cryptography

In 1978, McEliece [McE78] introduced the first public key encryption scheme based on linear codes. The main idea is to use a random code from a well–chosen family to generate the pair of keys. As the structure of the code is hidden (the secret key), it is a hard task to decipher a message for anyone who does not know a specific decoding algorithm for the chosen family of codes. The encryption of a plain message is done by using a basis of the code (*i.e.* a generator matrix), then some errors are introduced at random locations. The generator matrix (which is the public key) needs to be random to keep secret the structure of the code. The secret key is a part of the hidden structure of the code that allows to build an efficient decoding algorithm. The decryption step thus consists in applying the decoding algorithm to remove the errors, recovering a codeword in the public code. The initial message can then be derived from it using the public key.

For an attacker that does not know the secret key of the scheme, there are two solutions to recover the message. First, one could try to decode the cipher text as a noisy codeword of a random code. This problem, called *message recovery attack*, is related to the so–called *generic decoding problem*:

Problem 1. (Decisional) Generic Decoding Problem: *Let $\mathcal{C} \in \mathbb{F}^n$ be a linear code over a finite field $\mathbb{F}$, $t \leq n$ a positive integer and $\mathbf{y} \in \mathbb{F}^n$. Decide whether there exists a codeword $\mathbf{c} \in \mathcal{C}$ whose Hamming distance to $\mathbf{y}$ is less than or equal to t.*

This problem is supposed to be difficult in average, and was proven to be NP–complete in [BMvT78]. The second method consists in recovering the secret key from the public data. From this, it is possible to build a decoding algorithm that can decipher any message. In the literature, any such attack is usually called *key recovery attack*.

Improving the original proposal

In its initial proposal, McEliece [McE78] proposed to use classical binary Goppa codes, which is a subfamily of alternant codes. Up to now, all known attacks against it have exponential complexity in the parameters, hence the scheme is still considered secure. Adding to the fact that encryption and decryption are very fast, the McEliece's cryptosystem is at the center of modern cryptography, and one of the last code–based candidates for standardization of post–quantum cryptographic schemes to the NIST competition since the third round. However, it suffers a major drawback: the key sizes are too huge to be efficient. For more than forty years, several directions have been investigated to mitigate this problem. Among them, we distinguish two different angles: either replacing the family of classical Goppa codes with another one or using more structured codes (*i.e.* equipped with the action of some automorphism group), with the hope to keep the same advantages.

In 1986, Niederreiter [Nie86] suggested to use Generalized Reed–Solomon codes (GRS), but they are proved to be weak because of the structural attack of Sidelnikov and Shestakov [SS92]. As any GRS code can be seen as an *algebraic geometry* (AG) code on $\mathbb{P}^1$, Janwa and Moreno [JM96] then proposed to use AG codes from curves of arbitrary genus. More precisely, their paper includes three proposals: one based on AG codes, one on concatenated AG codes and a last one using subfield subcodes of AG codes (SSAG in short). For the version with concatenated codes, Sendrier found an effective attack in [Sen94]. For AG codes on curves with genus ≤ 2, an attack was found by Faure and Minder in [FM08, Min07, Fau09]. Finally, the scheme based on AG codes has been completely broken by Couvreur, Márquez–Corbella and Pellikaan [CMCP17], who proposed a filtration–based attack on AG codes for any genus, enabling decoding just by handling the public key and without knowledge on the geometric structure of the code. However, the authors underlined that subfield subcode of AG codes are resistant to this filtration attack. For this reason, we are led to consider the last proposition, that is schemes using SSAG codes, for which no efficient attack is known yet.

The other direction consists in using quasi–cyclic (QC) or quasi–dyadic (QD) codes. This additional structure allows to describe a given generator matrix with only a few rows, hence reducing key sizes. The case of QC codes has been proposed in [Gab05], where quasi–cyclic subcodes of BCH codes are suggested. Unfortunately, this family cannot be used as it does not have enough possible keys. This first paper was followed by many others, which proposed to use alternant or classical Goppa codes with different automorphism group like QC alternant codes [BCGO09] and QD Goppa codes [MB09, BLM11]. Notice that all these codes can be seen as subfield subcodes of AG codes on the projective line. However, since 2010, a new version of key recovery attack appeared, referred to as *algebraic attack*. This method consists in recovering the secret structure of an alternant code by solving a system of polynomial equations. In the generic case of alternant codes, this technique does not have polynomial complexity and cannot be used to recover the hidden structure in practice. However, in the case of QC alternant codes, the corresponding system of equations can be simplified, which leads to an attack. In particular, the authors in [FOP$^+$16] showed that the security of such schemes reduces to the security of a smaller code, the *folded code*, which ican be computed from the public data. This strategy was improved in [Bar18b], where it is proven that the initial key recovery problem on the original QC alternant code can be reduced to a smaller code that can be derived from the public one: the *invariant code*. In this last paper, Barelli improves the approach of [FOP$^+$16] and consider the case of automorphisms arising from a non–affine homography. In her thesis [Bar18a], she also initiated the study of QC SSAG codes from a Kummer cover of $\mathbb{P}^1$, showing that it has the same security as the scheme using alternant or Goppa codes.

Proximity tests to linear codes

Since their introduction in [LFKN90], *arithmetization techniques* for constructing short proof systems have been fruitfully applied to *probabilistically checkable proofs* (PCPs [BFLS91, AS92, ALM$^+$98]). Roughly speaking, in a probabilistic proof system for a binary relation $\mathcal{R}$, the arithmetization process transforms any instance–witness pair (x, w) into a word that belongs to some error–correcting code $\mathcal{C}$ if $(x, w) \in \mathcal{R}$, or is very far from $\mathcal{C}$ otherwise. Since the work of Kilian [Kil92] and Micali [Mic98], a lot of efforts have been put into making PCPs efficient enough to obtain practical sublinear non–interactive arguments for delegating computation. In search of reducing the work required to generate proofs, Interactive Oracle Proofs (IOPs, [BCS16]) have been introduced as a common generalization of PCPs and Interactive Proofs (IPs). At some point, aforementioned sublinear arguments require a proximity test to a Reed–Solomon code. As a solution, one can use an IOP of Proximity (IOPPs, [BCG$^+$17]) for RS codes. An IOPP for an error–correcting code consists in an interaction between a prover and a verifier, in which the prover seeks to convince that some word, given as oracle input

to the verifier, is indeed a codeword. In this case, the verifier accepts the proof; otherwise he rejects it with high probability.

The Fast Reed–Solomon Interactive Oracle Proof of Proximity (FRI) protocol, introduced in [BBHR18] and improved in [BKS18, BGKS20, BCI+20], is an IOPP for testing proximity to a RS code. As for its properties, it admits linear prover time, logarithmic verifier time and logarithmic query complexity. Being highly efficient, it is a crucial tool in systems deployed in the real life. The main drawback of considering RS codes for IOPP is that they must have an alphabet size larger than their length, meaning that practical IOP–based succinct arguments are designed over *large fields*. Moreover, the protocol FRI requires the set of evaluation points to have a specific structure: concretely, the evaluation set must be invariant under the action of a large group of order a power of two.

Considering AG codes instead of RS codes is not only natural, but could also weaken these limitations. In 2020, Bordage and Nardi [BN20] gave a clear criterion for constructing AG code–based IOPPs with linear proof length and sublinear query complexity, as well as a concrete instance for codes defined on Kummer type curves.

Contributions

Analysis of the McEliece scheme using quasi–cyclic SSAG codes

This contribution is a joint work with my supervisors and Elise Barelli, which has not been published yet.

In Chapter 3, we study the security of the secret key of McEliece schemes based on structured SSAG codes with a non–trivial permutation group, *i.e.* codes endowed with a group action arising from the underlying geometry of the curve from which the code is defined. We improve the technique initiated by Barelli in the Chapter 5 of her thesis [Bar18a], in which she shows that the security of the scheme reduces to the security of the *invariant code*. More precisely, she proposed a security reduction for quasi–cyclic SSAG codes constructed on Kummer–type curves, whose invariant subcode turns out to be an alternant code, *i.e.* a subfield subcode of an AG code on $\mathbb{P}^1$. To do so, she makes use of the fact that the quotient curve has a trivial divisor class group to recover the secret structure of the public code from the knowledge of the invariant one. Since the support and multiplier of alternant codes are weak to algebraic attacks [FOPT10], there is no advantage to consider structured SSAG codes whose invariant subcode is defined over $\mathbb{P}^1$ in McEliece's like cryptosystems, compared with the case of classical Goppa codes. In this thesis, we improve this method, which can consequently be applied to a more general framework.

The invariant code. Consider the following setting: given a Galois cover $\pi : \mathcal{Y} \to \mathcal{X}$ of smooth and irreducible projective curves over $\mathbb{F}_{q^m}$, we consider an SSAG code $\mathcal{C} := \mathsf{SSAG}_q(\mathcal{Y}, \mathcal{Q}, G)$, invariant under the action of some permutation σ induced by an automorphism of $\mathcal{Y}$, as public code in a McEliece scheme. If we denote by $\mathcal{X} := \mathcal{Y}/\langle\sigma\rangle$ the corresponding quotient curve, it turns out (as it was already noticed in [Bar18a, Corollary 5.3]) that the invariant code $\mathcal{C}^\sigma$ of $\mathcal{C}$ is also an SSAG code, defined over the quotient curve $\mathcal{X}$. More precisely,

$$\mathcal{C}^\sigma = \mathsf{SSAG}_q\left(\mathcal{X}, \mathcal{P}, \widetilde{G}\right),$$

for some support $\mathcal{P} \in \mathcal{X}(\mathbb{F}_{q^m})$ and divisor $\widetilde{G} \in \mathrm{Div}(\mathcal{X})$, which are explicitly described in terms of $\mathcal{Q}, G$ and the ramification in the cover $\mathcal{Y} \to \mathcal{X}$. To obtain an effective security reduction, we make the following assumptions on the quotient curve $\mathcal{X}$:

- $\mathcal{X}$ has a unique point at infinity P_∞, which totally ramifies in $\mathcal{Y} \to \mathcal{X}$;
- $\mathcal{X}$ admits a canonical divisor which is equivalent to $(2\mathfrak{g}(\mathcal{X}) - 2)P_\infty$.

Clearly, these hypotheses are satisfied by the projective line, thus in the case treated in [Bar18a, Chapter 5]. Our assumptions might sound restrictive, but there is actually a fairly large class of curves satisfying both of them, namely the class of $C_{a,b}$ curves (introduced in [Miu93]).

Security reduction. The geometric structure of the invariant code can be exploited to recover the secret elements of the public code. Roughly speaking, under some technical assumptions, the knowledge of the triple $(\mathcal{X}, \mathcal{P}, \widetilde{G})$ can be used to find $(\mathcal{Y}, \mathcal{Q}, G)$. Following the idea of [Bar18a, Chapter 5], the crucial step is to find a defining equation of the initial curve $\mathcal{Y}$. If the cover $\mathcal{Y} \to \mathcal{X}$ corresponds to an extension L/K of algebraic function fields, this can be done by finding the minimal polynomial of some element $y \in L$ such that $L = K(y)$. More precisely, the idea is to build a linear system whose solution, ideally unique, is the evaluation vector $\mathbf{y} := (y(Q))_{Q \in \mathcal{Q}}$. Once $\mathbf{y}$ it is found, we know all the data of the secret support $\mathcal{Q}$, which can then be used to recover the desired equation of $\mathcal{Y}$ using an appropriate interpolation method. The secret divisor G is then recovered by taking the pullback of the invariant one, *i.e* $G = \pi^* \widetilde{G}$. In our setting, the quotient curve might have positive genus, hence there is no reason for its divisor class group to be trivial. Consequently, we cannot hope to describe Riemann–Roch spaces on $\mathcal{Y}$ in terms of invariant data (as done by Barelli in her prior work). Thankfully, we show that this is not mandatory to build our linear system, as it is enough to consider a sufficiently large and explicit divisor $D \in \mathrm{Div}(\mathcal{X})$ such that

$$\pi^* \mathcal{L}_{\mathcal{X}}(D) \cdot y \in \mathcal{L}_{\mathcal{Y}}(G^{\perp}),$$

where $G^{\perp} \in \mathrm{Div}(\mathcal{Y})$ is the divisor associated to the dual code of the public SSAG.

Concrete instances. As illustrative examples, we consider the frameworks where $\mathcal{Y} \to \mathcal{X}$ is a Kummer cover or an elementary abelian p–cover (in Sections 3.3.2 and 3.3.3 respectively).

The first case corresponds to curves $\mathcal{Y}$ with equation of the form $y^{\ell} = f$, where f is a rational function that lies in the quotient curve and ℓ is an integer prime to the characteristic and such that $\ell \mid q^m - 1$. Therefore, the action of the automorphism σ acting on the public SSAG code is uniquely determined by the choice of a primitive ℓ–th root of unity ξ, *i.e.*

$$\sigma(y) = \xi \cdot y.$$

Consequently, the evaluation vector $\mathbf{y}$ satisfies a *geometric progression* on each orbit or size ℓ. Taking this into account to add more equations to our system, it is reasonable to find a unique solution.

In the second case, the curve $\mathcal{Y}$ has a defining equation of the form $y^{p^u} - y = f$, where $f \in \mathbb{F}_{q^m}(\mathcal{X})$, $u \geq 1$ and $p := \mathrm{char}(\mathbb{F}_{q^m})$. Assuming that $\mathbb{F}_{p^u} \subseteq \mathbb{F}_{q^m}$, the automorphism σ is here characterized by the choice of some element $\beta \in \mathbb{F}_{p^u}$, *i.e*

$$\sigma(y) = y + \beta.$$

This time, this action is traduced in terms of *arithmetic progression* in the vector $\mathbf{y}$.

In both cases, under some technical assumptions on the function f, all our computations realized on Magma end up with a unique solution to our linear systems, which is the desired vector $\mathbf{y}$.

Behaviour of the square of the dual of Goppa–like AG codes

This work is the result of a collaboration with Jade Nardi and Sabira El Khalfaoui. A preprint is available on arxiv [KLN23].

Since classical Goppa codes remain good candidates for post–quantum code–based cryptography, it is interesting to study specific AG codes on any genus curves, whose algebraic structure mimic theirs. Several attempts have been proposed in that direction, starting with Janwa and Moreno [JM96], who introduced *Goppa codes* on smooth and irreducible projective curves. Later on, Couvreur [Cou14] defined *Cartier codes* as specific subcode of SSAG codes. In Chapter 4 of this thesis, we introduce the family of *Goppa–like AG codes* and study the behaviour of the square of their dual, hence generalizing the distinguisher found in [MT21] in the case of alternant and classical Goppa codes.

Goppa–like AG codes. Let D be an effective divisor on a curve $\mathcal{X}$ over $\mathbb{F}_{q^m}$. Take a rational function $g \notin \mathcal{L}(D)$ and a set of evaluation points $\mathcal{P} \in \mathcal{X}(\mathbb{F}_{q^m})$ such that $\mathcal{P} \cap \mathrm{Supp}(D) = \varnothing$ and $\mathcal{P} \cap \mathrm{Supp}((g)) = \varnothing$. We define the Goppa–like AG code associated to the AG code $\mathcal{C} = C_{\mathcal{L}}(\mathcal{X}, \mathcal{P}, D + (g))$ as the subfield subcode of its dual, *i.e.*

$$\Gamma(\mathcal{P}, D, g) := \mathcal{C}^{\perp}|_{\mathbb{F}_q}.$$

The terminology *Goppa–like* is justified by the fact that our construction coincides with classical Goppa codes while considering codes over $\mathbb{P}^1$, in which case the rational function g plays the role

of the Goppa polynomial. Compared with the family of Goppa codes introduced in [JM96], the addition of the function g defines a multiplicator for the AG code which is algebraically related to the evaluation set. Moreover, it facilitates the use of SSAG codes as public keys for McEliece cryptosystems: in fact, given an error correcting capability t, we can fix a divisor $D \in \mathrm{Div}(\mathcal{X})$ such that $\deg(D) \geq 2t + 2\mathfrak{g}(\mathcal{X}) + 1$. Then, we obtain a family of codes in which public keys can be picked by running a set of functions g outside $\mathcal{L}(D)$. Our codes can also be compared with Cartier codes [Cou14] in the following way: given the same support $\mathcal{P}$ and divisor $D + (g)$, the Cartier code $\mathrm{Car}_q(\mathcal{P}, D + (g))$ is a subfield subcode of $\Gamma(\mathcal{P}, D, g)$. Moreover, we also provide a sufficient condition in terms of degree of the divisor for the two constructions to be equal.

Distinguisher attack in Goppa–like case. In [MT21], the authors benefited from the trace structure of the dual of a subfield subcode to display a distinguisher for alternants and classical Goppa codes. Roughly speaking, a *distinguisher* for a linear code consists in distinguishing a generator matrix of the code from a random matrix. In the case of [MT21], it is obtained by using the well–known behaviour of GRS codes with respect to the Schur product. In fact, the dimension of the square of an r–dimensional GRS code is smaller than expected from a random code, *i.e.*

$$\dim_{\mathbb{F}_{q^m}} \mathsf{GRS}_r(\mathbf{x}, \mathbf{y}) = 2r - 1, \text{ when } r < \frac{n}{2}.$$

instead of the expected quadratic upper bound $\binom{r+1}{2}$. As generalization of GRS codes, it is natural to observe that AG codes behave similarly under the square operation, the only difference being a non–trivial contribution due to the genus of the curve. Combining this specific structure with results on the product of Riemann–Roch spaces, we obtain the following bound on the dimension of the square of the dual of Goppa–like AG codes:

Proposition 4.9. *With above notation, assume* $s := \deg(D) \geq \mathfrak{g}(\mathcal{X})$*, and set*

$$k := \dim_{\mathbb{F}_{q^m}} \mathcal{C} \quad \textit{and} \quad e := \min\left(\left\lfloor \frac{m}{2} \right\rfloor, \left\lfloor \log_q\left(\frac{k^2}{s}\right) \right\rfloor\right).$$

Then

$$\dim_{\mathbb{F}_q}(\Gamma(\mathcal{P}, D, g)^{\perp})^{\star 2} \leq \binom{mk+1}{2} - \frac{m}{2}\left(k(k-1)(2e+1) - 2s\left(\frac{q^{e+1}-1}{q-1}\right)\right).$$

Weighted Euclidean division and one–point Goppa–like codes from $C_{a,b}$ curves. In the case of Goppa–like AG codes on $C_{a,b}$ curves associated with the one–point divisor $D := sP_\infty$ (where P_∞ is the unique point at infinity), we can improve the bound given in Proposition 4.10. In fact, such codes can be seen as the evaluation of bivariate polynomials, coming from the specific structure of the Riemann–Roch space $\mathcal{L}(sP_\infty)$, *i.e.*

$$\mathcal{L}(sP_\infty) = \mathrm{Span}\left(x^i y^j \mid 0 \leq i, 0 \leq j \leq a-1 \text{ and } ai + bj \leq s\right).$$

By defining a weighted degree on bivariate polynomials belonging to the ring $\mathcal{S} = \cup_{s \geq 0} \mathcal{L}(sP_\infty)$, we then manage to perform division algorithms via Gröbner basis. As expected in the univariate case, the remainder has a weighted degree *generally* smaller than the divisor's. Under some additional conditions on the divisor $D = sP_\infty + (g)$, we then prove

Theorem 4.22. *Let* $s' > s$ *and take a rational function* $g \in \mathcal{L}(s'P_\infty)$ *with* $\deg_{a,b}(g) = s'$*. Suppose that* $s \geq (s'-s)q + 2\mathfrak{g}_{a,b} - 1$ *and set*

$$k := \dim_{\mathbb{F}_{q^m}} C_{\mathcal{L}}(\mathcal{X}, \mathcal{P}, sP_\infty + (g)) \quad \textit{and} \quad e^* := \min\left(\left\lfloor \frac{m}{2} \right\rfloor, \left\lceil \log_q\left(\frac{k^2}{s'(q-1)^2}\right) \right\rceil + 1\right).$$

Then

$$\dim_{\mathbb{F}_q}(\Gamma(\mathcal{P}, sP_\infty, g)^{\perp})^{\star 2} \leq \binom{mk+1}{2} - \frac{m}{2}(k^2(2e^*+1) + k - 2s'(q^{e^*} - q^{e^*-1} + 1)).$$

Efficiency of the distinguisher. Our results generalize the ones of [MT21] in the sense that our bounds coincide with theirs in the genus zero case. Consequently (and without much a surprise), we are also only able to distinguish high rate Goppa–like AG codes. As for efficiency, several computations realized on Magma tend to show that the bound given in Theorem 4.24 is sharp whenever the code seems random. More precisely, it is likely to be the case whenever the function g has simple zeroes. Compared with the case of classical Goppa codes, the Goppa polynomial is usually assumed to be square–free, which is the reason why the bound for classical Goppa codes given in [MT21] is sharp. However, as already noticed by Mora and Tillich, it seems complicated to turn this distinguisher into an efficient structural attack.

Two examples: Elliptic and Hermitian codes. In Section 4.4, we analyze our distinguisher both in the case of codes from an elliptic curve and the Hermitian curve (which belong to the family of $C_{a,b}$ curves). The first case being *close* to classical Goppa codes (*i.e.* the genus of such curves is one), our maximal distinguishable rates are roughly the same. Considering Hermitian codes, we show that the high genus imposed by the choice of the curve makes the distinguisher ineffective. In fact, the upper bound on the dimension given in Theorem 4.24 is always bigger than the maximal possible length of the code. Adding the fact that such codes can be encoded efficiently [BRS22], it is encouraging to consider the family of one–point Goppa–like AG codes from the Hermitian curve as public keys in a McEliece cryptosystem. Some computations to compare this family of codes with classical Goppa codes can be found at the end of Chapter 4.

IOP of Proximity to AG codes along the Hermitian tower

This contribution is the result of a collaboration with Sarah Bordage, Jade Nardi and Hugues Randriambololona [BNLR22].

In [BN20], Bordage and Nardi proposed to replace Reed–Solomon codes with AG codes while testing proximity to linear codes in IOP of Proximity. The initial idea was to remove the limitations imposed by RS codes by considering more structured codes. Hence, they provide a generic criterion for constructing AG code–based IOPP, by defining in a general framework sequences of AG codes compatible with proximity testing, which are called *foldable* codes. In the Chapter 5 of this thesis, we give an explicit family of foldable AG codes defined on the *Hermitian tower*, and study the properties of the IOPP derived from it.

Foldable AG codes. Let $\mathcal{X}$ be a curve defined over some finite field $\mathbb{F}$, equipped with a finite solvable group $\mathcal{G} \subseteq \mathrm{Aut}(\mathcal{X})$. By solvability of $\mathcal{G}$, there exists a sequence of normal subgroups $\{\mathrm{Id}\} := \mathcal{G}_0 \rhd \mathcal{G}_1 \rhd \cdots \rhd \mathcal{G}_r := \mathcal{G}$ such that each quotient $\Gamma_i := \mathcal{G}_i/\mathcal{G}_{i-1}$ is cyclic. We thus obtain a sequence of curves

$$\mathcal{X} := \mathcal{X}_r \to \mathcal{X}_{r-1} \to \cdots \to \mathcal{X}_0 := \mathcal{X}/\mathcal{G},$$

such that each $\mathcal{X}_{i-1}$ arises as the quotient of $\mathcal{X}_i$ by Γ_i. A proximity text to some AG code $\mathcal{C} := C_{\mathcal{L}}(\mathcal{X}, \mathcal{P}, G)$ consists in deciding whether some function $f : \mathcal{P} \to \mathbb{F}$ belongs to $\mathcal{C}$ or not. To do so, we construct a sequence of AG codes $(\mathcal{C}_i)_{i=0}^r$ with decreasing length such that $\mathcal{C}_i = C_{\mathcal{L}}(\mathcal{X}_i, \mathcal{P}_i, G_i)$ is defined over $\mathcal{X}_i$ and $\mathcal{C}_r := \mathcal{C}$. Be choosing carefully the structure of each code, the first proximity test can be turned into a membership test of a function $f' : \mathcal{P}_0 \to \mathbb{F}$ in the smallest code $\mathcal{C}_0$. Roughly speaking, $\mathcal{P}$ is chosen globally $\mathcal{G}$–invariant and each $\mathcal{P}_{i-1}$ is obtained as the projection of $\mathcal{P}_i$ to the quotient curve $\mathcal{X}_{i-1}$. The tricky part is to define a suitable sequence of divisors $(G_i)_{i=0}^r$ such that each Riemann–Roch space $\mathcal{L}_{\mathcal{X}_i}(G_i)$ can be explicitly described in terms of Riemann–Roch spaces in the quotient curve $\mathcal{X}_{i-1}$. Such conditions are examined in Section 5.1.2 while defining *compatible* divisors. If we manage to construct such a sequence of codes, the initial code $\mathcal{C}$ is said to be *foldable*.

The case of the Hermitian tower. Consider the infinite tower of function fields $(F_i)_{i\geq 0}$ over $\mathbb{F}_{q^2}$ such that $F_0 = \mathbb{F}_{q^2}(x_0)$ and $F_i = F_{i-1}(x_i)$, recursively defined by

$$x_i^q + x_i = x_{i-1}^{q+1}, \text{ for all } i \geq 1.$$

It corresponds to an infinite sequence of curves

$$\cdots \to \mathcal{X}_i \to \mathcal{X}_{i-1} \to \cdots \to \mathcal{X}_0 = \mathbb{P}^1,$$

called the *Hermitian tower*. Contrary to the case of Kummer foldable codes (see. [BN20]), we have no hope to use Maharaj's theorem [Mah04, Theorem 2.2] to obtain a suitable decomposition of Riemann–Roch spaces, as it requires the order of the automorphism group to be prime with the characteristic of the field, which is obviously not the case in our setting. To overcome this difficulty, we can consider one–point divisors, supported by the unique point at infinity $P_\infty^{(i)} \in \mathcal{X}_i(\mathbb{F}_{q^2})$. Doing so, we can obtain a desired decomposition *by hand*, using the fact that the basis of $\mathcal{L}_{\mathcal{X}_i}(mP_\infty^{(i)})$ is well–known, *i.e.*

$$\mathcal{L}_{\mathcal{X}_i}(mP_\infty^{(i)}) = \mathrm{Span}\left(x_0^{a_0} \cdots x_i^{a_i} \mid 0 \leq a_0, 0 \leq a_j \leq q-1 \text{ and } \sum_{j=0}^{i} a_j q^{i-j}(q+1)^j \leq m \right).$$

Due to the Weierstrass gap theory, we also need to increase the degree of our divisors at each step (with respect to the sequence of genera in the tower), in order to guarantee the existence of

balancing functions (which are needed to prove the soundness of our IOPP). At the end, we propose the following sequence of foldable AG codes:

$$\mathcal{C}_i := C_{\mathcal{L}}(\mathcal{X}_i, \mathcal{P}_i, d_i P_\infty^{(i)}), \text{ for } i \geq 1; \tag{1}$$

where $\mathcal{P}_i \subseteq \mathcal{X}_i(\mathbb{F}_{q^2}) \setminus \left\{P_\infty^{(i)}\right\}$ is a set of length at most q^{i+2}, and the integers d_i's are recursively defined by $d_{i-1} = \lfloor \frac{d_i}{q} \rfloor + 2\mathfrak{g}(\mathcal{X}_{i-1})$.

AG–IOPP system with foldable Hermitian codes. In Section 5.4, we define an IOPP to test proximity of some function $f^{(i_{\max})} : \mathcal{P}_{i_{\max}} \to \mathbb{F}_{q^2}$ to the AG code $C_{\mathcal{L}}(\mathcal{X}_{i_{\max}}, \mathcal{P}_{i_{\max}}, d_{i_{\max}} P_\infty^{(i_{\max})})$. It corresponds to a $i_{\max}$–round interactive proof in which the initial proximity test is reduced to a proximity test of some function $f^{(0)}$, defined as a *fold* of $f^{(i_{\max})}$, to the smallest AG code defined on $\mathcal{X}_0$. Its main properties are summarized in the following informal theorem:

Theorem (Informal, see [BNLR22, Theorem 45])**.** *Let $\mathcal{C}_{i_{\max}}$ be an AG code as in Equation* (1)*, with length n at most equal to $q^{i_{\max}+2}$. Then the* IOPP *system described in Section 5.4.1 has perfect completeness, small soundness error for every proximity parameter δ, and the following properties:*

rounds complexity	$r(n) < \log(n)$
proof length	$\ell(n) < n$
query complexity	$q(n) \leq tq \log(n) + 1$
prover complexity	$t_p(n) = \mathcal{O}\left(n \cdot \mathrm{M}_{\mathbb{F}_{q^2}}(q) \log(q)\right)$
verifier complexity	$t_v(n) = \mathcal{O}\left(\log(n) \cdot \mathrm{M}_{\mathbb{F}_{q^2}}(q) \log(q)\right)$,

where $\mathrm{M}_{\mathbb{F}_{q^2}}(d)$ denotes the cost of multiplying two degree-d univariate polynomials over $\mathbb{F}_{q^2}$.

Résumé

Contexte

De cryptographie à clé publique à cryptographie post–quantique

Dans le domaine de la cryptographie, la *cryptographie à clé publique* (ou *cryptographie asymétrique*) regroupe les schémas utilisant une paire de clés : la *clé publique* connue de tous et la *clé secrète*, seulement connue du destinataire des messages chiffrés. La famille de schémas de chiffrement à clé publique a été présentée pour la première fois par Diffie et Hellman [DH76]. Avant cela, la plupart des schémas cryptographiques provenaient de la *cryptographie à clé secrète* (ou *cryptographie symétrique*), qui utilise la même clé pour chiffrer et déchiffrer le message. Dans de tels systèmes, les deux participants doivent au préalable se mettre d'accord sur un moyen sûr et efficace d'échanger la clé, ce qui n'est plus nécessaire en cryptographie à clé publique.

Le premier schéma en cryptographie asymétrique fût proposé par Rivest, Shamir et Adelman [RSA78] : c'est le schéma de chiffrement RSA. Il est encore aujourd'hui largement utilisé pour la transmission de données sécurisées, et sa sécurité est principalement fondée sur la difficulté de factoriser de grands entiers en produit de nombres premiers (dans le sens où une factorisation efficace casserait le schéma). Depuis, plusieurs autres schémas ont été proposés, dont la sécurité repose sur d'autres problèmes issus de la théorie des nombres, comme le problème du logarithme discret. Cependant, la menace grandissante des ordinateurs quantiques impose une diversification de la cryptographie : en effet, ces problèmes de théorie des nombres pourraient être cassés en temps polynomial sur un ordinateur quantique à l'aide de l'algorithme de Shor [Sho94]. La *cryptographie post–quantique* est le domaine qui regroupe les nouveaux systèmes de chiffrement, dont la sécurité s'appuie sur de nouvelles hypothèses, différentes de celles issues de théories des nombres, et qui résistent aux algorithmes quantiques. Dans cette thèse, on s'intéresse à la *cryptographie à base de codes*, qui utilise des codes correcteurs d'erreurs. Dans ce cas, la principale hypothèse de sécurité est le problème du décodage d'un code linéaire aléatoire, pour lequel aucun algorithme quantique n'est connu à ce jour. Pour cette raison, la cryptographie à base de codes est un bon candidat en cryptographie post–quantique.

Cryptographie à base de codes correcteurs d'erreurs

En 1978, McEliece propose le premier système de chiffrement à clé publique à base de codes linéaires. L'idée principale est d'utiliser un code aléatoire issu d'une famille bien choisie pour générer la paire de clés. La structure du code étant cachée (c'est la clé secrète), il est difficile de déchiffrer un message pour quiconque ne connaît pas un algorithme de décodage efficace pour la famille de codes choisie. Le chiffrement d'un message est réalisé en utilisant une base du code (*c.-à-d.* une matrice génératrice), et des erreurs sont ajoutées à des positions aléatoires. Pour garder secrète la structure du code, la matrice génératrice doit sembler aléatoire : c'est la clé publique. La clé secrète est un algorithme de décodage efficace, qui peut être obtenu à partir de la structure secrète du code. L'étape de déchiffrement consiste à récupérer d'abord un mot de code en appliquant l'algorithme de décodage au message crypté pour retirer les erreurs. Le message initial peut ensuite être retrouvé à partir du mot de code à partir de la clé publique.

Pour un attaquant qui ne connaît pas la clé secrète du schéma, il existe deux solutions pour retrouver le message. D'abord, il peut tenter de déchiffrer le message crypté en tant que mot bruité dans un code linéaire aléatoire. Ce problème, appelé *attaque sur le message*, est lié au *problème de décodage*.

Problème 1 (Problème de décodage (version Décisionnel))**.** Soit $\mathcal{C} \in \mathbb{F}^n$ un code linéaire sur un corps fini $\mathbb{F}$, $t \leq n$ un entier positif et $\mathbf{y} \in \mathbb{F}^n$. Décider s'il existe un mot de code $\mathbf{c} \in \mathcal{C}$ dont la distance de Hamming à $\mathbf{y}$ est inférieure ou égal à t.

Ce problème est connu pour être difficile en moyenne, et a été prouvé NP–complet dans [BMvT78]. La seconde méthode concerne la récupération de la clé secrète à partir de la seule clé publique. Ceci fait, il est possible de construire un algorithme de décodage qui peut déchiffrer n'importe quel message. Dans la littérature, ce type d'attaque est appelé *attaque sur la clé*.

Améliorer la proposition initiale

Dans son schéma de 1978, McEliece [McE78] propose d'utiliser la famille de code de Goppa binaires classiques, qui est une sous–famille des codes alternants. Jusqu'à aujourd'hui, toutes les attaques contre ce schéma ont une complexité exponentielle en les paramètres du code publique, donc le cryptosystème est encore considéré sécurisé. Additionné au fait que le chiffrement et le déchiffrement sont très rapides, le cryptosystème de McEliece est au centre de la cryptographie moderne, et l'un des derniers candidats à base de codes correcteurs pour la compétition lancée par le NIST en vue d'une standardisation de la cryptographie post–quantique. Cependant, il possède un défaut majeur : les tailles de clés sont trop grosses pour que le schéma soit efficace. Depuis plus de quarante ans, des recherches ont été menées dans différentes directions pour essayer de corriger ce problème. Parmi elles, on distingue deux tendances : remplacer la famille de codes de Goppa classiques par une autre famille et utiliser des codes plus structurés, avec l'espoir de garder les mêmes avantages.

En 1986, Niederreiter [Nie86] suggéra d'utiliser les codes de Reed–Solomon généralisés (GRS), mais ils sont la cible d'une attaque structurelle par Sidelnikov et Shestakov [SS92]. Comme tout code GRS peut être vu comme un code de géométrie algébrique (AG) sur $\mathbb{P}^1$, Janwa et Moreno [JM96] ont proposé d'utiliser les codes AG construits sur des courbes de genre quelconque. Plus précisément, leur papier contient trois propositions : une sur les codes AG, une sur les codes AG concaténés et une dernière utilisant des sous–codes sur un sous–corps de codes AG (appelés plus simplement des codes SSAG). Pour la version avec les codes concaténés, Sendrier a trouvé une attaque efficace dans [Sen94]. Les codes AG sur des courbes de genre ≤ 2 sont la cible d'une attaque de Faure et Minder [FM08, Min07, Fau09]. Plus tard, Couvreur, Márquez–Corbella et Pellikaan [CMCP17] ont complètement cassé le schéma à partir de codes AG, en proposant une attaque par filtration qui, en genre quelconque, permet de décoder un message avec la seule connaissance de la clé publique. Cependant, les auteurs ont souligné le fait que les sous–codes sur un sous–corps de codes AG sont résistants à cette attaque par filtration. Pour cette raison, nous sommes amenés à considérer la dernière proposition, c'est–à–dire les schémas utilisant les codes SSAG, pour lesquels aucune attaque efficace n'est connue pour le moment.

L'autre piste consiste à utiliser des codes quasi cycliques (QC) ou quasi dyadiques (QD). Cette structure additionnelle permet de décrire les matrices génératrices avec seulement quelques lignes, réduisant ainsi la tailles des clés. Le cas des codes quasi cycliques a d'abord été proposé dans [Gab05], ou des sous–codes de codes BCH sont suggérés. Malheureusement, cette famille de codes n'est pas satisfaisante, car elle possède trop peu de clés possibles. Cet article a été suivi par de nombreux autres, qui proposent d'utiliser les codes alternants ou de Goppa classiques avec différents groupes d'automorphismes, comme les codes alternants quasi cycliques [BCGO09] ou encore les codes de Goppa quasi dyadiques [MB09, BLM11]. On remarque au passage que tous ces codes peuvent être vu comme des sous–codes sur un sous-corps de codes AG définis sur la droite projective. Cependant, depuis 2010, une nouvelle version d'attaque sur la clé est apparue, appelée *attaque algébrique*. Cette méthode consiste à retrouver la structure secrète d'un code alternant en résolvant un système d'équations polynomiales. Dans le cas général des codes alternants, cette technique n'a pas une complexité polynomiale et ne peut donc pas être utilisée pour retrouver la structure cachée en pratique. Cependant, dans le cas de codes alternants quasi–cycliques, le système d'équations correspond peut être simplifié, ce qui donne une attaque. En particulier, les auteurs de [FOP$^+$16] ont montré que la sécurité de ces schémas de chiffrement se réduit à la sécurité d'un code plus petit, le *code replié*, qui peut facilement être obtenu à partir des données publiques. Cette stratégie a été améliorée dans [Bar18b], ou il est montré que le problème de récupération de clé initial associé à un code alternant quasi cyclique peut se réduire au même problème sur un code plus petit que l'on peut déduire du code public : le *code invariant*. Dans ce dernier papier, Barelli améliore l'approche de [FOP$^+$16] et considère le cas d'automorphismes provenant d'une homographie non affine. De plus, dans sa thèse [Bar18a], elle étudie la sécurité du schéma de McEliece utilisant des codes SSAG quasi cycliques construit sur un revêtement de Kummer de $\mathbb{P}^1$, montrant ainsi que ce cryptosystème possède la même sécurité que celui utilisant des codes de Goppa ou alternants.

Test de proximité à des codes linéaires

Introduit dans [LFKN90], les *techniques d'arithmétisation* pour construire des systèmes de preuves courtes ont depuis été appliquées aux *preuves vérifiables probabilistiquement* (PCPs [BFLS91, AS92, ALM+98]). Sans entrer dans les détails, dans un système de preuve probabiliste pour une relation binaire $\mathcal{R}$ donnée, le processus d'arithmétisation transforme un couple exemple–témoin (x, w) en un mot qui appartient à un certain code correcteur d'erreurs $\mathcal{C}$ si $(x, w) \in \mathcal{R}$, ou est très loin de $\mathcal{C}$ sinon. Depuis les travaux de Kilian [Kil92] et Micali [Mic98], beaucoup d'efforts ont étés déployés pour rendre les systèmes PCPs suffisamment efficaces pour obtenir des arguments non interactifs sous–linéaires pour déléguer les calculs. Dans l'optique de réduire les efforts requis pour générer des preuves, la notion de Preuves par Oracle Interactif (IOPs [BCS16]) est créée comme une généralisation commune des protocoles PCPs et des Preuves Intéractives (IPs). À un moment, les arguments sous–linéaires mentionnés plus haut nécessitent un test de proximité à un code de Reed–Solomon. Comme solution, on peut utiliser un IOP de Proximité (IOPPs [BCG+17]) pour les RS codes. Un IOPP pour un code correcteur consiste en une interaction entre un prouveur et un vérifieur, dans laquelle le prouver cherche à convaincre qu'un mot, donné en entrée au vérifieur, appartient bien au code. Dans ce cas, le vérifieur accepte la preuve; dans le cas contraire, il la rejette avec grande probabilité.

Le protocole FRI, d'abord proposé dans [BBHR18] puis amélioré dans [BKS18, BGKS20, BCI+20], est un IOPP pour tester la proximité à un code de Reed–Solomon. En ce qui concerne son efficacité, il nécessite un temps linéaire pour le prouver, logarithmique pour le vérifieur et une complexité de requêtes logarithmique également. Étant très efficace, c'est un outil important déployé dans de nombreux systèmes de nos jours. Le principal point faible de considérer les codes RS pour des tests de proximité est que la taille de l'alphabet doit être plus grosse que la longueur du code. Par conséquent, les IOP concrets à base de codes RS sont construits sur des corps très gros.

De plus, le protocole FRI demande que l'ensemble d'évaluation du RS code correspondant ait une structure spéciale : concrètement, il doit être invariant sous l'action d'un grand groupe d'ordre une puissance de deux.

Le fait de considérer les codes AG au lieu des codes RS est non seulement naturel, mais pourrait aussi nous affranchir de ses restrictions. En 2020, Bordage et Nardi donnent un critère précis pour construire des systèmes IOPP à base de codes AG ayant des preuves en temps linéaire et une complexité de requêtes sous–linéaire. À titre d'exemple, elles proposent également un système concret pour des codes AG construit sur des courbes de Kummer.

Contributions

Analyse du schéma de McEliece utilisant des codes SSAG quasi–cycliques

Ces travaux, encore non publiés, sont le fruit d'une collaboration avec Elise Barelli et mes deux directeurs de thèse.

Dans le Chapitre 3, nous étudions la sécurité de la clé secrète du schéma de McEliece à base de codes SSAG structurés avec un groupe de permutation non trivial, c'est–à–dire équipés d'une action de groupe issue de la géométrie de la courbe à partir de laquelle le code est construit. Nous améliorons la technique du Chapitre 5 de la thèse d'Élise Barelli [Bar18a], dans laquelle elle montre que la sécurité de la clé secrète se réduit à celle du *code invariant*. Plus précisément, elle propose une réduction de sécurité pour les codes SSAG quasi cycliques construits sur des courbes de Kummer, dont le sous–code invariant est un code alternant, c'est–à–dire le sous–code sur un sous–corps d'un code AG sur $\mathbb{P}^1$. Pour ce faire, elle utilise entre autre le fait que la courbe quotient possède un groupe de classe trivial pour retrouver la structure du code public à partir de celle du code invariant. Comme le support et le multiplicateur d'un code alternant sont fragiles aux *attaques algébriques*, il n'y a aucun avantage à considérer cette famille de codes plutôt que celle des codes de Goppa classiques. Dans cette thèse, nous améliorons cette technique de sorte qu'elle puisse être appliquée dans un contexte plus général.

Le code invariant. Considérons la situation suivante : étant donné un revêtement Galoisien de courbes projectives lisses et irréductibles $\pi : \mathcal{Y} \to \mathcal{X}$ sur $\mathbb{F}_{q^m}$, on considère le code $\mathcal{C} = \mathsf{SSAG}_q(\mathcal{Y}, \mathcal{Q}, G)$, invariant sous l'action d'une permutation σ induite par un automorphisme de la courbe $\mathcal{Y}$, comme code public dans un schéma de McEliece. Si $\mathcal{X} := \mathcal{Y}/\langle\sigma\rangle$ désigne la courbe quotient, le code invariant $\mathcal{C}^\sigma$ de $\mathcal{C}$ est aussi un code SSAG (voir [Bar18a, Corollaire 5.3]), définit

sur la courbe $\mathcal{X}$. Plus précisément, on a

$$\mathcal{C}^\sigma = \mathsf{SSAG}_q\left(\mathcal{X}, \mathcal{P}, \widetilde{G}\right),$$

pour un certain support $\mathcal{P} \subseteq \mathcal{X}(\mathbb{F}_{q^m})$ et diviseur $\widetilde{G} \in \mathrm{Div}(\mathcal{X})$ qui sont décrits explicitement en termes de $\mathcal{Q}$, G et de la ramification dans le revêtement $\mathcal{Y} \to \mathcal{X}$. Pour que notre réduction de sécurité fonctionne, il nous faut rajouter deux hypothèses sur la courbe quotient $\mathcal{X}$:

1. $\mathcal{X}$ possède un unique point à l'infini P_∞, qui est totalement ramifié dans $\mathcal{Y} \to \mathcal{X}$;
2. Il existe un diviseur canonique sur $\mathcal{X}$ qui est équivalent à $(2\mathfrak{g}(\mathcal{X}) - 2)P_\infty$.

Ces hypothèses sont clairement vérifiées pour la droite projective, c'est–à–dire dans le cas traité dans [Bar18a, Chapitre 5]. Elles pourraient sembler restrictives de premier abord, mais il existe en fait une classe assez grosse et bien connue de courbes qui les vérifie : les courbes $C_{a,b}$ [Miu93].

Réduction de sécurité. La structure géométrique du code invariant peut être exploitée pour récupérer les éléments secrets du code public. Globalement, sous certaines hypothèses techniques, le triplet $(\mathcal{X}, \mathcal{P}, \widetilde{G})$ peut être utilisé pour retrouver $(\mathcal{Y}, \mathcal{Q}, G)$. En effet, en reprenant l'idée de [Bar18a, Chapitre 5], l'étape clé est de retrouver une équation de la courbe initiale $\mathcal{Y}$. Si le revêtement $\mathcal{Y} \to \mathcal{X}$ correspond à l'extension de corps de fonctions L/K, cela peut être réalisé en trouvant le polynôme minimal d'un élément $y \in L$ primitif sur K. plus précisément, l'idée est de construire un système d'équations linéaires dont le vecteur d'évaluation $\mathbf{y} := (y(Q))_{Q\in\mathcal{Q}}$ est solution (idéalement unique). Une fois $\mathbf{y}$ retrouvé, on peut utiliser le support secret $\mathcal{Q}$ désormais connu pour récupérer une équation de $\mathcal{Y}$ en utilisant la méthode d'interpolation appropriée. Le diviseur G est quant à lui reconstruit comme le tiré en arrière du diviseur invariant, *i.e.* $G = \pi^*\widetilde{G}$. Dans notre contexte, la courbe quotient peut avoir un genre strictement positif, donc il n'y a pas de raison que son groupe de classes soit trivial. Par suite, on a peu d'espoir d'être en mesure d'exprimer les espaces de Riemann–Roch sur $\mathcal{Y}$ en termes de données connues sur la courbe invariante (comme le faisait Barelli). Fort heureusement, nous montrons que cela n'est pas nécessaire, puisque pour construire notre système d'équation, il nous suffit de considérer un diviseur suffisamment gros et explicite $D \in \mathrm{Div}(\mathcal{X})$ tel que

$$\pi^*\mathcal{L}_\mathcal{X}(D) \cdot y \in \mathcal{L}_\mathcal{Y}(G^\perp),$$

où $G^\perp \in \mathrm{Div}(\mathcal{Y})$ est le diviseur du code dual du SSAG public.

Exemples concrets. En tant qu'applications, on s'intéresse aux cas ou $\mathcal{Y} \to \mathcal{X}$ est un revêtement du Kummer ou un revêtement élémentaire p–abélien (dans les Sections 3.3.2 et 3.3.3 respectivement).

Le cas des revêtements de Kummer correspond à des courbes dont l'équation est de la forme $y^\ell = f$, ou f est une fonction rationnelle sur la courbe quotient, et ℓ est un entier positif premier à la caractéristique tel que $\ell \mid q^m - 1$. Ainsi, l'action de l'automorphisme σ agissant sur le code SSAG public est entièrement déterminée par le choix d'une racine primitive ℓ–ième de l'unité ξ, *i.e.*

$$\sigma(y) = \xi \cdot y.$$

Par conséquent, le vecteur d'évaluation $\mathbf{y}$ recherché vérifie une condition de *progression géométrique* sur chacune de ses orbites de taille ℓ. En utilisant cette remarque pour rajouter des équations à notre système, il est raisonnable d'espérer obtenir une solution unique.

Dans le cas d'un revêtement p-abélien, la courbe $\mathcal{Y}$ est définie par une équation de la forme $y^{p^u} - y = f$, où $f \in \mathbb{F}_{q^m}(\mathcal{X})$, $u \geq 1$ et p est la caractéristique de $\mathbb{F}_{q^m}$. En supposant de plus que $\mathbb{F}_{p^u} \subseteq \mathbb{F}_{q^m}$, l'automorphisme σ est dans ce cas caractérisé par le choix d'un élément $\beta \in \mathbb{F}_{p^u}$, *i.e.*

$$\sigma(y) = y + \beta.$$

Cette fois, l'action ci–dessus est traduite en termes de *progression arithmétique* sur le vecteur d'évaluation $\mathbf{y}$.

Dans les deux cas, sous l'ajout de certaines hypothèses techniques sur la fonction rationnelle f, plusieurs tests réalisés sur Magma semblent indiquer que l'on se retrouve toujours avec une solution unique, qui est le vecteur recherché.

Comportement du carré du dual des codes AG Goppa–like

Cette contribution est le résultat d'une collaboration avec Jade Nardi et Sabira El Khalfaoui. Un préprint est disponible sur arxiv [KLN23].

Comme les codes de Goppa classiques restent de bons candidats à la cryptographie post–quantique à base de codes correcteurs, il est intéressant d'étudier des codes AG spécifiques, construit sur des courbes de genre quelconque, dont la structure copie la leur. Plusieurs tentatives ont étés réalisées dans cette direction, à commencer par Janwa et Moreno [JM96], qui ont défini des *codes de Goppa* sur des courbes projectives lisses et irréductibles. Plus tard, Couvreur [Cou14] propose l'étude de certains sous–codes de codes SSAG : les *codes de Cartier*. Dans le Chapitre 4 de cette thèse, nous définissons la famille des codes AG dits *Goppa–like*, et étudions le comportement du carré de leur dual, généralisant ainsi le distingueur de [MT21] pour les codes alternants et les codes de Goppa classiques.

Codes AG Goppa–like. Soit D un diviseur effectif sur une courbe $\mathcal{X}$ définit sur le corps fini $\mathbb{F}_{q^m}$. Fixons une fonction $g \notin \mathcal{L}(D)$ et un ensemble de points d'évaluation $\mathcal{P} \in \mathcal{X}(\mathbb{F}_{q^m})$ tels que $\mathcal{P} \cap \mathrm{Supp}(D) = \varnothing$ et $\mathcal{P} \cap \mathrm{Supp}((g)) = \varnothing$. On définit le code Goppa–like associé au code AG $\mathcal{C} = C_{\mathcal{L}}(\mathcal{X}, \mathcal{P}, D + (g))$ comme le sous–code sur $\mathbb{F}_q$ de son dual, c'est–à–dire

$$\Gamma(\mathcal{P}, D, g) := \mathcal{C}^{\perp}|_{\mathbb{F}_q}.$$

La terminologie *Goppa–like* est justifiée par le fait que notre construction coïncide avec celle des codes de Goppa classiques dans le cas de codes construit sur $\mathbb{P}^1$, auquel cas le fonction g joue le rôle du polynôme de Goppa. Comparé aux codes de Goppa proposés dans [JM96], l'ajout de la fonction g permet de définir un multiplicateur pour le code AG qui est algébriquement lié à l'ensemble d'évaluation. De plus, ce multiplicateur facilite l'utilisation des ces codes dans le contexte d'un schéma de McEliece : en effet, étant donné un taux de correction d'erreurs t, on peut commencer par choisir un diviseur $D \in \mathrm{Div}(\mathcal{X})$ tel que $\deg(D) \geq 2t + 2\mathfrak{g}(\mathcal{X}) + 1$. On obtient alors une famille de codes dans laquelle on peut sélectionner nos clés publiques en choisissant la fonction g dans un ensemble n'appartenant pas à $\mathcal{L}(D)$. Il est également possible de comparer nos codes aux codes de Cartier [Cou14] de la manière suivante : étant fixé un support $\mathcal{P}$ et un diviseur $D + (g)$, le code de Cartier $\mathrm{Car}_q(\mathcal{P}, D + (g))$ se trouve être un sous–code de $\Gamma(\mathcal{P}, D, g)$. Nous donnons aussi une condition nécessaire sur le degré du diviseur pour que les deux constructions rendent le même code.

Attaque par distingueur dans le cas Goppa–like. Dans [MT21], les auteurs bénéficient de la structure particulière de la trace du dual d'un sous–code sur un sous–corps pour établir un distingueur pour les codes alternants et les codes de Goppa classiques. En quelques mots, un *distingueur* pour un code linéaire permet de décider si une de ses matrices génératrices semble aléatoire ou non. Dans le cas de [MT21], le distingueur est obtenu à partir du comportement bien connu des codes GRS vis–à–vis du produit de Schur. En effet, la dimension du carré d'un code GRS de dimension r est bien plus petite que celle attendue pour un code aléatoire, *i.e.*

$$\dim_{\mathbb{F}_{q^m}} \mathsf{GRS}_r(\mathbf{x}, \mathbf{y}) = 2r - 1, \text{ si } r < \frac{n}{2}$$

au lieu de la borne supérieur quadratique $\binom{r+1}{2}$ attendue. En tant que généralisation des codes GRS, il n'est pas étonnant de constater que les carrés codes de géométries algébriques réagissent de la même manière, la seule différence notable étant l'apparition d'une contribution non triviale liée au genre de la courbe. En associant cette propriété avec des résultats sur le produit d'espaces de Riemann–Roch, nous obtenons la borne suivante sur la dimension du carré du dual d'un code AG Goppa–like:

Proposition 4.9. *Avec les notations précédentes, supposons que $s = \deg(D) \geq \mathfrak{g}(\mathcal{X})$, et posons*

$$k := \dim_{\mathbb{F}_{q^m}} \mathcal{C} \quad \textit{et} \quad e := \min\left(\left\lfloor \frac{m}{2} \right\rfloor, \left\lfloor \log_q\left(\frac{k^2}{s}\right) \right\rfloor\right).$$

Alors

$$\dim_{\mathbb{F}_q}(\Gamma(\mathcal{P}, D, g)^{\perp})^{\star 2} \leq \binom{mk+1}{2} - \frac{m}{2}\left(k(k-1)(2e+1) - 2s\left(\frac{q^{e+1}-1}{q-1}\right)\right).$$

Division euclidienne à poids et codes AG Goppa–like à un point sur les courbes $C_{a,b}$. Dans le cas particulier des codes AG Goppa–like construit sur une courbe $C_{a,b}$ associé au diviseur à un point $D = sP_\infty$ (P_∞ étant l'unique point à l'infini), nous parvenons à améliorer la borne donnée dans la Proposition 4.10. En effet, de tels codes peuvent être vus comme l'évaluation de polynômes bivariés, dû à la structure particulière de l'espace de Riemann–Roch $\mathcal{L}(sP_\infty)$, *i.e.*

$$\mathcal{L}(sP_\infty) = \text{Span}\left(x^i y^j \mid 0 \leq i, 0 \leq j \leq a-1 \text{ and } ai + bj \leq s\right).$$

En définissant un degré à poids sur chaque polynôme bivarié de l'anneau $\mathcal{S} = \cup_{s\geq 0}\mathcal{L}(sP_\infty)$, nous introduisons une notion de division euclidienne généralisée via des bases de Gröbner. Comme attendu dans le cas univarié, le reste de nos divisions a *généralement* un degré plus petit que celui du diviseur. Sous des conditions techniques supplémentaires sur le degré du diviseur $D = sP_\infty + (g)$, nous démontrons alors :

Theorem 4.22. *Soient $s' > s$ deux entiers et $g \in \mathcal{L}(s'P_\infty)$ de degré à poids $\deg_{a,b}(g) = s'$ une fonction rationnelle. Supposons que $s \geq (s'-s)q + 2\mathfrak{g}_{a,b} - 1$ et posons*

$$k := \dim_{\mathbb{F}_{q^m}} C_{\mathcal{L}}(\mathcal{X}, \mathcal{P}, sP_\infty + (g)) \quad \textit{et} \quad e^* := \min\left(\left\lfloor \frac{m}{2} \right\rfloor, \left\lceil \log_q\left(\frac{k^2}{s'(q-1)^2}\right)\right\rceil + 1\right).$$

Alors

$$\dim_{\mathbb{F}_q}(\Gamma(\mathcal{P}, sP_\infty, g)^\perp)^{\star 2} \leq \binom{mk+1}{2} - \frac{m}{2}(k^2(2e^*+1) + k - 2s'(q^{e^*} - q^{e^*-1} + 1)).$$

Efficacité du distingueur. Nos résultats généralisent ceux de [MT21] dans le sens où nos bornes coïncident avec les leurs dans le cas du genre zéro. Par conséquent (et sans réelle surprise), nous sommes aussi seulement en mesure de distinguer des codes AG Goppa–like de ratio élevé. Concernant l'efficacité de notre distingueur, plusieurs tests réalisés sur Magma semblent montrer que la borne du Théorème 4.24 est optimale lorsque le code *semble* aléatoire. Plus précisément, nous pensons que c'est le cas lorsque la fonction rationnelle g ne possède que des zéros simples. Si l'on compare avec le cas des codes de Goppa classiques, il est très souvent supposé que le polynôme de Goppa est sans facteur carré, raison pour laquelle la borne donnée pour les codes de Goppa classiques dans [MT21] est optimale. Cependant, comme Mora et Tillich l'avaient déjà remarqué, il semble difficile d'utiliser ce distingueur dans une attaque structurelle sur le code.

Les exemples des codes elliptiques et Hermitiens. Dans la Section 4.4, nous analysons notre distingueur dans le cas de codes construits sur une courbe elliptique ou sur la courbe Hermitienne (qui sont toutes deux des courbes $C_{a,b}$). Le premier cas étant relativement *proche* des codes de Goppa classiques (les courbes elliptiques sont de genre 1), nos plus gros ratios distinguables sont plus ou moins les mêmes. En ce qui concerne les codes hermitiens, nous montrons que le genre élevé de la courbe rend le distingueur inefficace. En effet, la borne supérieure sur la dimension donnée dans le Théorème 4.24 est toujours supérieure à la longueur maximale du code. Ajouté au fait que ces codes peuvent être encodés efficacement [BRS22], il est encourageant de considérer la famille d'AG codes Goppa–like à un point construit sur la courbe Hermitienne pour générer la paire de clés dans le contexte du cryptosystème de McEliece. En fin de Chapitre 4, nous proposons un comparatif d'efficacité entre cette famille de codes et celle des codes de Goppa classiques.

IOP de Proximité aux codes AG construit sur la tour Hermitienne

Ces travaux ont été menés en collaboration avec Sarah Bordage, Jade Nardi and Hugues Randriambololona [BNLR22].

Dans [BN20], Bordage et Nardi ont proposé de remplacer les codes de Reed–Solomon dans les IOPs de Proximité aux codes linéaires. L'idée initiale était de pallier aux limitations imposées par les codes RS en considérant des codes plus structurés. Elles proposent alors un critère général pour construire des IOPPs à base de codes AG, en donnant un contexte global pour qu'une famille de codes AG soit compatible avec les tests de proximité : on parle alors de codes *repliables*. Dans le Chapitre 5 de cette thèse, nous donnons une famille explicite de codes AG repliables définie sur la *tour Hermitienne*, et étudions les propriétés du système IOPP qui en découle.

Codes AG repliables. Soit $\mathcal{X}$ une courbe définie sur un corps fini $\mathbb{F}$ et $\mathcal{G} \subseteq \mathrm{Aut}(\mathcal{X})$ un groupe d'automorphisme résoluble. Il existe une suite de sous–groupes distingués $\{\mathrm{Id}\} := \mathcal{G}_0 \rhd \mathcal{G}_1 \rhd \cdots \rhd \mathcal{G}_r := \mathcal{G}$ tel que tous les quotients $\Gamma_i := \mathcal{G}_i/\mathcal{G}_{i-1}$ sont cycliques. Cela correspond à une suite de revêtements de courbes

$$\mathcal{X} := \mathcal{X}_r \to \mathcal{X}_{r-1} \to \cdots \to \mathcal{X}_0 := \mathcal{X}/\mathcal{G},$$

ou chaque $\mathcal{X}_{i-1}$ est le quotient de $\mathcal{X}_i$ par le Γ_i. Un test de proximité à un code AG $\mathcal{C} := C_{\mathcal{L}}(\mathcal{X}, \mathcal{P}, G)$ consiste à déterminer si une fonction $f : \mathcal{P} \to \mathbb{F}$ appartient à $\mathcal{C}$ ou non. Pour ce faire, l'idée est de construire une suite de codes AG $(\mathcal{C}_i)_{i=0}^r$ de longueur décroissante telle que $\mathcal{C}_i = C_{\mathcal{C}}(\mathcal{X}_i, \mathcal{P}_i, G_i)$ est défini sur la i-ème courbe $\mathcal{X}_i$ et $\mathcal{C}_r := \mathcal{C}$. En choisissant efficacement la structure de chacun des codes, le premier test de proximité peut se réduire à un test d'appartenance d'une nouvelle fonction $f' : \mathcal{P}_0 \to \mathbb{F}$ au plus petit code $\mathcal{C}_0$. Plus précisément, $\mathcal{P}$ est choisi globalement $\mathcal{G}$–invariant et chaque $\mathcal{P}_{i-1}$ est obtenu comme la projection de $\mathcal{P}_i$ sur la courbe $\mathcal{X}_{i-1}$. Le point clé est de construire une suite de diviseurs $(G_i)_{i=0}^r$ de sorte que l'espace de Riemann–Roch $\mathcal{L}_{\mathcal{X}_i}(G_i)$ puisse être décrit de manière explicite à l'aide d'espaces de Riemann–Roch sur la courbe quotient $\mathcal{X}_{i-1}$. De telles conditions donneront lieu à la notion de diviseurs *compatibles* en Section 5.1.2. S'il est possible de construire une telle suite de codes, le code initial $\mathcal{C}$ est dit *repliable*.

Le cas de la tour Hermitienne. Considérons la tour infinie de corps de fonctions $(F_i)_{i\geq 0}$ sur $\mathbb{F}_{q^2}$ tel que $F_0 = \mathbb{F}_{q^2}(x_0)$ est le corps de fonction rationnel et $F_i = F_{i-1}(x_i)$, où

$$x_i^q + x_i = x_{i-1}^{q+1}, \text{ pour tout } i \geq 1.$$

Cette tour correspond à une suite infinie de courbes

$$\cdots \to \mathcal{X}_i \to \mathcal{X}_{i-1} \to \cdots \to \mathcal{X}_0 = \mathbb{P}^1,$$

appelée la *tour Hermitienne*. Contrairement au cas des codes repliables sur des courbes de type Kummer (cf. [BN20]), nous n'avons ici aucune chance de pouvoir utiliser le théorème de Maharaj [Mah04, Theorem 2.2] pour obtenir la décomposition d'espaces de Riemann–Roch recherchée, puisque l'une des hypothèses demande à ce que l'ordre de l'automorphisme agissant sur le code soit premier à la caractéristique, ce qui n'est bien sûr pas le cas ici. Pour palier à ce problème, nous pouvons de nouveau considérer des codes à un point, supporté par un multiple de l'unique point à l'infini $P_\infty^{(i)} \in \mathcal{X}_i(\mathbb{F}_{q^2})$. Ce faisant, nous pouvons exploiter la structure spécifique des espaces $\mathcal{L}_{\mathcal{X}_i}(mP_\infty^{(i)})$, *i.e.*

$$\mathcal{L}_{\mathcal{X}_i}(mP_\infty^{(i)}) = \mathrm{Vect}\left(x_0^{a_0} \cdots x_i^{a_i} \mid 0 \leq a_0, 0 \leq a_j \leq q-1 \text{ et } \sum_{j=0}^{i} a_j q^{i-j}(q+1)^j \leq m \right),$$

pour obtenir *à la main* la décomposition que l'on recherche. Prenant en compte la théorie des sauts de Weierstrass dans notre construction, nous demandons aussi à ce que le degré de nos diviseurs soit augmenté à chaque étape (d'un facteur dépendant du genre des courbes successives) afin de garantir l'existence de *fonctions balances* (qui sont primordiales pour démontrer la soundness du système IOPP correspondant). À termes, nous proposons la suite de codes AG repliables suivante :

$$\mathcal{C}_i := C_{\mathcal{L}}(\mathcal{X}_i, \mathcal{P}_i, d_i P_\infty^{(i)}), \text{ pour } i \geq 1; \tag{2}$$

où $\mathcal{P}_i \subseteq \mathcal{X}_i(\mathbb{F}_{q^2}) \setminus \left\{P_\infty^{(i)}\right\}$ est un ensemble de points rationnels de longueur au plus q^{i+2}, et la suite d'entiers d_i's sont définis de manière récursive par $d_{i-1} = \lfloor \frac{d_i}{q} \rfloor + 2\mathfrak{g}(\mathcal{X}_{i-1})$.

Système AG–IOPP avec des codes Hermitiens repliables. Dans la section 5.4, nous définissons un système IOPP pour tester la proximité d'une fonction $f^{(i_{\max})} : \mathcal{P}_{i_{\max}} \to \mathbb{F}_{q^2}$ au code AG $C_{\mathcal{L}}(\mathcal{X}_{i_{\max}}, \mathcal{P}_{i_{\max}}, d_{i_{\max}} P_\infty^{(i_{\max})})$. Il s'agit d'une preuve interactive en $i_{\max}$–tours dans laquelle le test de proximité initial est réduit à un test de proximité d'une fonction $f^{(0)}$, définie comme un *replié* de $f^{(i_{\max})}$, au plus petit code définit sur $\mathcal{X}_0$. Ses principales propriétés sont résumées dans le théorème (informel) qui suit :

Theorem (Informel, voir [BNLR22, Théorème 45])**.** *Soit $\mathcal{C}_{i_{\max}}$ un code AG comme dans l'équation* (1)*, de longueur $n \leq q^{i_{\max}+2}$. Alors le système* IOPP *décrit dans la Section 5.4.1 est parfaitement complet, a une petite erreur de soundness pour tout paramètre de proximité δ, et les propriétés suivantes :*

complexité des tours	$r(n) < \log(n)$
longueur de preuve	$\ell(n) < n$
complexité des requêtes	$q(n) \leq tq\log(n) + 1$
complexité du prouver	$t_p(n) = \mathcal{O}\left(n \cdot \mathrm{M}_{\mathbb{F}_{q^2}}(q)\log(q)\right)$
complexité du vérifieur	$t_v(n) = \mathcal{O}\left(\log(n) \cdot \mathrm{M}_{\mathbb{F}_{q^2}}(q)\log(q)\right),$

où $\mathrm{M}_{\mathbb{F}_{q^2}}(d)$ *désigne le coût de multiplication de deux polynômes univariés de degré* d *sur* $\mathbb{F}_{q^2}$.

Chapter 1

Algebraic geometry codes

1.1 Coding theory

Let $\mathbb{F}_q$ be the finite field with q elements, where q is a power of some prime number p. For the upcoming bases in coding theory and without further precision, we refer to [MS86].

1.1.1 Linear codes

Definition 1.1 (Linear Code)**.** Let $k \leq n$ be two non–negative integers. A *linear* $[n,k]_q$ *code* over $\mathbb{F}_q$ is a subspace $\mathcal{C} \subseteq \mathbb{F}_q^n$ of dimension k. The integers n and k are respectively the length and the dimension of $\mathcal{C}$, and any vector of $\mathcal{C}$ is called a *codeword*. The rate of $\mathcal{C}$ is the ratio $R := \frac{k}{n}$.
A *generator matrix* $\mathbf{M}$ of $\mathcal{C}$ is a $k \times n$ matrix over $\mathbb{F}_q$ whose rows are a basis of $\mathcal{C}$ as a vector space. In particular, we have

$$\mathcal{C} = \left\{ \mathbf{xM} \mid \mathbf{x} \in \mathbb{F}_q^k \right\}.$$

A *parity check* matrix $\mathbf{H}$ of $\mathcal{C}$ is a $(n-k) \times n$ matrix over $\mathbb{F}_q$ such that

$$\forall \mathbf{c} \in \mathcal{C},\ \mathbf{c} \in \mathcal{C} \iff \mathbf{Hc}^T = \mathbf{0}.$$

Generator matrices of codes are not unique, and it is the same for parity check ones. However, it is often convenient to have a generator matrix with the specific form

$$\mathbf{M} = (\mathbf{I}_k \mid \mathbf{A}),$$

where $\mathbf{A}$ is a $k \times (n-k)$ matrix over $\mathbb{F}_q$ and $\mathbf{I}_k$ is the identity matrix of size k. In this situation, $\mathbf{M}$ is said to be *systematic*. A linear code does not always have a systematic generator matrix but if so, this matrix is unique and the code is said to be *systematic*.

Definition 1.2 (Dual code)**.** Let $\mathcal{C}$ be a linear code over $\mathbb{F}_q$. Its *dual code*, denoted by $\mathcal{C}^\perp$, consists in all vectors which are orthogonal to all codewords of $\mathcal{C}$. More precisely,

$$\mathcal{C}^\perp := \{\mathbf{y} \in \mathbb{F}_q^n \mid \mathbf{yc}^{\mathrm{T}} = 0 \text{ for all } \mathbf{c} \in \mathcal{C}\}.$$

It is easy to see that any parity check matrix of $\mathcal{C}$ is a generator matrix of its dual $\mathcal{C}^\perp$. As a consequence, $\mathcal{C}^\perp$ has same length n and dimension $n - \dim_{\mathbb{F}_q}(\mathcal{C})$.

Definition 1.3 (Hamming distance)**.** The *Hamming distance* between two vectors $\mathbf{x}, \mathbf{y} \in \mathbb{F}_q^n$, denoted by $d_H(\mathbf{x}, \mathbf{y})$, is defined by

$$d_H(\mathbf{x}, \mathbf{y}) := |\{i \in \{1, \cdots, n\} \mid x_i \neq y_i\}|,$$

where $\mathbf{x} = (x_1, \cdots, x_n)$ and $\mathbf{y} = (y_1, \cdots, y_n)$.
The Hamming weight of a vector $\mathbf{x} \in \mathbb{F}_q^n$ is defined by its distance to the zero vector, *i.e.* its number of non–zero components:

$$w_H(\mathbf{x}) := d_H(\mathbf{x}, \mathbf{0}) = |\{i \in \{1, \cdots, n\} \mid x_i \neq 0\}|.$$

Since we will only deal with the Hamming metric, we simply talk about the distance between two codewords or the weight of a codeword.

Definition 1.4 (Minimum distance)**.** The *minimum distance* of a code $\mathcal{C}$, denoted by $d(\mathcal{C})$ or just d, is the minimum Hamming distance between two of its codewords, namely

$$d(\mathcal{C}) := \min\{d_H(\mathbf{x},\mathbf{y}) \mid \mathbf{x},\mathbf{y} \in \mathcal{C} \text{ and } \mathbf{x} \neq \mathbf{y}\}.$$

The minimum distance of a linear code $\mathcal{C}$ can also be seen as the minimum weight of its non–zero codewords, that is to say

$$d = \min\{w_H(\mathbf{x}) \mid \mathbf{x} \in \mathcal{C} \setminus \{0\}\}.$$

Later on, any linear code $\mathcal{C}$ over $\mathbb{F}_q$ will be described in terms of its length, dimension and minimum distance. For this reason, such a code will be referred to as an $[n,k,d]_q$ code (or just $[n,k]_q$ code). The following famous theorem makes the link between these parameters.

Theorem 1.5 (Singleton bound)**.** *If $\mathcal{C}$ is an $[n,k,d]_q$ code, then*

$$n - k \geq d - 1.$$

Codes with $k + d = n + 1$ are in a sense optimal, and thus are called MDS codes (maximal distance separable codes). The easiest example is the case of Reed–Solomon codes:

Example 1.6. Let r be a positive integer and choose a generator β of $\mathbb{F}_q^*$, *i.e.* such that $\mathbb{F}_q^* = \{\beta, \beta^2, \ldots, \beta^{q-1}\}$. The Reed–Solomon code of length $n = q-1$ and dimension r over $\mathbb{F}_q$ is defined by

$$\mathsf{RS}_r := \{(f(\beta), f(\beta^2), \cdots, f(\beta^{q-1})) \mid f \in \mathbb{F}_q[T] \text{ and } \deg(f) < r\}.$$

It is easily checked that RS_r is MDS, *i.e.* $n = k + d - 1$.

1.1.2 Punctured code and subfield subcode

In this section, we describe several ways to construct new codes from existing ones.

Definition 1.7 (Puncturing)**.** Let $\mathcal{C} \subseteq \mathbb{F}_q^n$ be a linear code and $\mathcal{I} \subseteq \{1, \cdots, n\}$ a set of coordinates. The *punctured* code of $\mathcal{C}$ at $\mathcal{I}$ is defined by

$$\mathrm{Punct}_{\mathcal{I}}(\mathcal{C}) := \left\{(c_i)_{i \in \{1,\cdots,n\}\setminus\mathcal{I}} \mid \mathbf{c} = (c_1, \cdots, c_n) \in \mathcal{C}\right\}.$$

This is a code of length $n - |\mathcal{I}|$.

Proposition 1.8. *Let $\mathcal{C} \subseteq \mathbb{F}_q^n$ be a $[n,k,d]_q$ linear code and $\mathcal{I} \subseteq \{1, \cdots, n\}$. Then* $\mathrm{Punct}_{\mathcal{I}}(\mathcal{C})$ *is an $[n - |\mathcal{I}|, k', d']_q$ code with*

$$k - |\mathcal{I}| \leq k' \leq k \quad \textit{and} \quad d - |\mathcal{I}| \leq d' \leq d.$$

From now on, let $m \geq 1$ be a positive integer and consider the finite extension $\mathbb{F}_{q^m}$ of $\mathbb{F}_q$. Below, we describe two ways to construct new codes from existing ones: considering a linear code over $\mathbb{F}_{q^m}$, it may happen that some of its codewords lie in $\mathbb{F}_q^n$, leading to the following definition:

Definition 1.9 (Subfield subcode)**.** Let $\mathcal{C} \subseteq \mathbb{F}_{q^m}^n$ be a linear code over $\mathbb{F}_{q^m}$. Its *subfield subcode* over $\mathbb{F}_q$, denoted by $\mathcal{C}|_{\mathbb{F}_q}$, is the subcode over $\mathbb{F}_q$ consisting in all codewords of $\mathcal{C}$ which lie in $\mathbb{F}_q^n$, *i.e.*

$$\mathcal{C}|_{\mathbb{F}_q} := \mathcal{C} \cap \mathbb{F}_q^n.$$

Usually, it is an hard task to find the exact parameters of a subfield subcode. A first estimation is given by the following theorem:

Theorem 1.10. *Let $\mathcal{C}$ be linear $[n,k,d]_{q^m}$ code. Then $\mathcal{C}|_{\mathbb{F}_q}$ in an $[n,k',d']_q$ code with*

$$k' \geq n - m(n-k) \quad \textit{and} \quad d' \geq d.$$

Another construction that permits to build a code over $\mathbb{F}_q$ starting from a code $\mathcal{C}$ over $\mathbb{F}_{q^m}$ uses the trace operator.

Definition 1.11 (Trace code)**.** Given an extension $\mathbb{F}_{q^m}/\mathbb{F}_q$ of finite field, the *trace* operator

$$\mathrm{Tr}_{\mathbb{F}_{q^m}/\mathbb{F}_q} : \mathbb{F}_{q^m} \to \mathbb{F}_q$$

is defined for all $x \in \mathbb{F}_{q^m}$ by

$$\mathrm{Tr}_{\mathbb{F}_{q^m}/\mathbb{F}_q}(x) := \sum_{i=0}^{m-1} x^{q^i}.$$

This definition naturally extends to vectors $\mathbf{x} \in \mathbb{F}_{q^m}^n$, so that the trace acts component–wise:

$$\mathrm{Tr}_{\mathbb{F}_{q^m}/\mathbb{F}_q}(\mathbf{x}) = \left(\mathrm{Tr}_{\mathbb{F}_{q^m}/\mathbb{F}_q}(x_1), \cdots, \mathrm{Tr}_{\mathbb{F}_{q^m}/\mathbb{F}_q}(x_n)\right),$$

and thus to codes $\mathcal{C}$ over $\mathbb{F}_{q^m}$:

$$\mathrm{Tr}_{\mathbb{F}_{q^m}/\mathbb{F}_q}(\mathcal{C}) = \left\{\mathrm{Tr}_{\mathbb{F}_{q^m}/\mathbb{F}_q}(\mathbf{c}) \mid \mathbf{c} \in \mathcal{C}\right\}.$$

The code $\mathrm{Tr}_{\mathbb{F}_{q^m}/\mathbb{F}_q}(\mathcal{C})$ is called the *trace code* of $\mathcal{C}$.

Given a linear code $\mathcal{C}$ over $\mathbb{F}_{q^m}$, we have a trivial upper bound on the dimension of its trace code over $\mathbb{F}_q$, *i.e.*

$$\dim_{\mathbb{F}_q} \mathrm{Tr}_{\mathbb{F}_{q^m}/\mathbb{F}_q}(\mathcal{C}) \leq m \cdot \dim_{\mathbb{F}_{q^m}}(\mathcal{C}).$$

We conclude this section by stating Delsarte's theorem, which makes the link between subfield subcode and trace code.

Theorem 1.12 ([Del75, Theorem 2])**.** *Let $\mathcal{C} \subseteq \mathbb{F}_{q^m}^n$ be any linear code, then*

$$(\mathcal{C} \cap \mathbb{F}_q^n)^\perp = \mathrm{Tr}_{\mathbb{F}_{q^m}/\mathbb{F}_q}\left(\mathcal{C}^\perp\right).$$

Remark 1.13. When there is no ambiguity on the fields (which will be the case in Section 1.1.4 and the whole Chapter 4), we may write $\mathrm{Tr}(\cdot)$ instead of $\mathrm{Tr}_{\mathbb{F}_{q^m}/\mathbb{F}_q}(\cdot)$.

1.1.3 Permutation group and invariant code

Let $\mathfrak{S}_n$ be the group of permutations of $\{1, \cdots, n\}$.

Definition 1.14 (Permutation group)**.** Let $\mathcal{C}$ be a linear code of length n over $\mathbb{F}_q$ and $\sigma \in \mathfrak{S}_n$ a permutation acting on $\mathcal{C}$ via $\mathbf{c}^\sigma = \left(c_{\sigma(1)}, \cdots, c_{\sigma(n)}\right)$, for every codeword $\mathbf{c} \in \mathcal{C}$. We say that $\mathcal{C}$ is *σ–invariant* if $\mathcal{C}^\sigma = \mathcal{C}$, in which case we say that σ is a *permutation* of $\mathcal{C}$. The *permutation group* of $\mathcal{C}$ is defined as the subset of all such permutations, *i.e.*

$$\mathrm{Perm}(\mathcal{C}) := \{\sigma \in \mathfrak{S}_n \mid \mathcal{C}^\sigma = \mathcal{C}\}.$$

Definition 1.15 (Invariant code)**.** Given a linear code $\mathcal{C} \subseteq \mathbb{F}_q^n$ together with a permutation subgroup $\Sigma \subseteq \mathrm{Perm}(\mathcal{C})$, we define the *invariant code* of $\mathcal{C}$ under Σ as the subcode

$$\mathcal{C}^\Sigma := \{c \in \mathcal{C} \mid c^\sigma = c,\ \forall \sigma \in \Sigma\} \subseteq \mathcal{C}.$$

Defined this way, the invariant code has repeated entries (*i.e.* the coordinates of its codewords are constant on each orbit under the action of Σ), so we usually use another one: the *punctured invariant code*, denoted by $\overline{\mathcal{C}}^\Sigma$. More precisely, all the orbits of Σ on $\{1, \ldots, n\}$ have same cardinality $|\Sigma|$, and $\overline{\mathcal{C}}^\Sigma$ is obtained from $\mathcal{C}^\Sigma$ by keeping only the first entry in each of them.

Example 1.16. With above notation, assume that $\ell \mid n = \mathrm{length}(\mathcal{C})$ and that $\Sigma = \langle \sigma_\ell \rangle$ is cyclic of order ℓ, generated by the ℓ–quasi–cyclic shift σ_ℓ (which is cyclic on each of the $\frac{n}{\ell}$ blocks of length ℓ of $\{1, \ldots, n\}$). Considering the set of indices $\mathcal{I}_\ell := \{1, ..., n\} \backslash \{1, \ell + 1, \ldots, n - \ell + 1\}$, the punctured invariant code is in this case defined by

$$\overline{\mathcal{C}}^\Sigma := \mathrm{Punct}_{\mathcal{I}_\ell}\left(\mathcal{C}^\Sigma\right) = \mathrm{Punct}_{\mathcal{I}_\ell}(\ker((\tilde{\sigma}_\ell - \mathrm{id})|_{\mathcal{C}})),$$

where $\overline{\sigma}_\ell : \mathbb{F}_q^n \to \mathbb{F}_q^n$ is induced by σ_ℓ and id is the identity map. In particular, the code $\overline{\mathcal{C}}^\Sigma$ has length $\frac{n}{\ell}$.

Notice that by definition, it is possible to construct a generator matrix of the (punctured) invariant code from the knowledge of a generator matrix of $\mathcal{C}$ and the induced permutation. Throughout this thesis, we always work with the punctured invariant code $\overline{\mathcal{C}}^\Sigma$, and simply write $\mathcal{C}^\Sigma$. Hence, when talking about the *invariant code*, we always implicitly assume it is the punctured one, meaning that is has smaller length.

1.1.4 Schur product of codes

In the discussion below, to make it more readable, we shall write $\mathrm{Tr}(\cdot)$ instead of $\mathrm{Tr}_{\mathbb{F}_{q^m}/\mathbb{F}_q}(\cdot)$ when considering the Trace operator.

Definition 1.17 (Schur product of codes)**.** Let $\mathcal{C}$ and $\mathcal{D}$ be two linear codes over $\mathbb{F}_{q^m}$ with same length n. We define their *Schur product* by

$$\mathcal{C} \star \mathcal{D} := \langle \mathbf{c} \star \mathbf{d} \mid \mathbf{c} \in \mathcal{C}, \mathbf{d} \in \mathcal{D} \rangle_{\mathbb{F}_{q^m}},$$

where $\mathbf{c} \star \mathbf{d}$ stands for the component–wise product of vectors. If $\mathcal{C} = \mathcal{D}$, we call $\mathcal{C}^{\star 2} := \mathcal{C} \star \mathcal{C}$ the square code of $\mathcal{C}$.

It is clear that if $\mathcal{C}$ and $\mathcal{D}$ have respective dimension k_1 and k_2, an obvious bound on the dimension of their Schur product is $k_1 k_2$. However, this bound is not relevant when $\mathcal{C} \cap \mathcal{D} \neq \{0\}$. In particular, for any $[n, k]_{q^m}$ code $\mathcal{C}$, we have

$$\dim_{\mathbb{F}_{q^m}} \mathcal{C}^{\star 2} \leq \min\left(n, \binom{k+1}{2}\right). \tag{1.1}$$

For a *random* linear code $\mathcal{C}$ (*i.e.* its generators matrices cannot be distinguished from random matrices) whose square does not fill the full space, the dimension of its square is $\binom{k+1}{2}$ with high probability (see. [CCMZ15]).

In this thesis (see Chapter 4), we focus on square code considerations to construct a distinguisher, *i.e.* a way to decide if a given matrix generates a structured code or is a random matrix. In particular, we will make good use of the following result, which describes the structure of the Schur product of two trace codes:

Proposition 1.18 ([MT21, Proposition 12])**.** *Let $\mathcal{C}$ and $\mathcal{D}$ be two linear codes with same length n over $\mathbb{F}_{q^m}$. Then*

$$\mathrm{Tr}(\mathcal{C}) \star \mathrm{Tr}(\mathcal{D}) \subseteq \sum_{i=0}^{m-1} \mathrm{Tr}\Big(\mathcal{C} \star \mathcal{D}^{q^i}\Big),$$

where $\mathcal{D}^{q^i} := \Big\{(d_1^{q^i}, \ldots, d_n^{q^i}) \mid \mathbf{d} \in \mathcal{D}\Big\}$.

Considering square codes, this result can be improved:

Proposition 1.19 ([MT21, Proposition 15])**.** *Let $\mathcal{C}$ be a linear code over $\mathbb{F}_{q^m}$. Then*

$$\mathrm{Tr}(\mathcal{C})^{\star 2} \subseteq \sum_{i=0}^{\lfloor \frac{m}{2} \rfloor} \mathrm{Tr}\Big(\mathcal{C} \star \mathcal{C}^{q^i}\Big).$$

Moreover, if m is even,

$$\dim_{\mathbb{F}_q} \mathrm{Tr}\Big(\mathcal{C} \star \mathcal{C}^{q^{\frac{m}{2}}}\Big) \leq \frac{m}{2}(\dim_{\mathbb{F}_{q^m}} \mathcal{C})^2.$$

An estimation on the dimension of the square of the trace of a linear code can be derived from the above proposition.

Corollary 1.20 ([MT21, Corollary 16])**.** *Let $\mathcal{C}$ be any $\mathbb{F}_{q^m}$–linear code. Then*

$$\dim_{\mathbb{F}_q} \mathrm{Tr}(\mathcal{C})^{\star 2} \leq m \cdot \dim_{\mathbb{F}_{q^m}} \mathcal{C}^{\star 2} + \binom{m}{2}(\dim_{\mathbb{F}_{q^m}} \mathcal{C})^2.$$

Furthermore, if $\dim_{\mathbb{F}_q} \mathrm{Tr}(\mathcal{C}) = m \cdot \dim_{\mathbb{F}_{q^m}}(\mathcal{C})$*, then*

$$\dim_{\mathbb{F}_q} \mathrm{Tr}(\mathcal{C})^{\star 2} - \binom{\dim_{\mathbb{F}_q} \mathrm{Tr}(\mathcal{C}) + 1}{2} \leq m \cdot \left(\dim_{\mathbb{F}_{q^m}} \mathcal{C}^{\star 2} - \binom{\dim_{\mathbb{F}_{q^m}} \mathcal{C} + 1}{2}\right).$$

The above corollary implies that if the dimension of a square code is smaller than the one expected for a random code, namely

$$\dim_{\mathbb{F}_{q^m}}(\mathcal{C}^{\star 2}) < \binom{\dim_{\mathbb{F}_{q^m}} \mathcal{C} + 1}{2},$$

then this property is retained for the trace code, *i.e.*

$$\dim_{\mathbb{F}_q} \mathrm{Tr}(\mathcal{C})^{\star 2} < \binom{\dim_{\mathbb{F}_q} \mathrm{Tr}(\mathcal{C}) + 1}{2}.$$

As we will see in Chapter 4, this fact is especially true for Reed–Solomon codes, and more importantly for algebraic geometry codes.

1.2 Tools of algebraic geometry

In this section, we introduce some required tools in algebraic geometry, both in the case of curves and function fields. Our main references are [Mor93, Chapter 1] for curves and [Sti09] for the function field point of view.

As we are only interested in the finite field case, we define our objects over some finite field $\mathbb{F}_q$ and we denote by $\overline{\mathbb{F}_q}$ its algebraic closure.

1.2.1 Algebraic curves over finite field

Let $\mathbb{A}^n$ and $\mathbb{P}^n$ be the n–dimensional affine and projective spaces over $\mathbb{F}_q$ respectively. As usual, any *point* P in $\mathbb{P}^n$ is an equivalence class of $(n+1)$–tuples, denoted by $P = [x_1 : \cdots : x_{n+1}]$, with $x_i \in \overline{\mathbb{F}_q}$ not all zero, under the relation:

$$[x_1 : \cdots : x_{n+1}] \equiv [y_1 : \cdots : y_{n+1}] \iff \exists \lambda \in \mathbb{F}_q^* \text{ such that } \forall i \in \{1, \cdots, n\},\ x_i = \lambda y_i.$$

Definition 1.21 (Projective set)**.** A polynomial $F \in \mathbb{F}_q[X_1, \cdots, X_{n+1}]$ is *homogeneous of degree d* if for any $(x_1, \cdots, x_{n+1}) \in \mathbb{F}_q^{n+1}$ and any $\lambda \in \mathbb{F}_q^*$, we have

$$F(\lambda x_1, \cdots, \lambda x_{n+1}) = \lambda^d F(x_1, \cdots, x_{n+1}).$$

Given a subset $S \subseteq \mathbb{F}_q[X_1, \cdots, X_{n+1}]$ of homogeneous polynomials, we define its *zero set* by

$$Z(S) = \{P \in \mathbb{P}^n \mid F(P) = 0\ ,\ \forall F \in S\}.$$

A subset $\mathcal{Y} \subseteq \mathbb{P}^n$ is called a *projective algebraic set* (or just projective set) if $\mathcal{Y} = Z(S)$ for some set of homogeneous polynomials S. A projective set $\mathcal{Y}$ is said to be *irreducible* if it is non–empty and if it cannot be written as the union of two distinct algebraic subsets $\mathcal{Y} = \mathcal{Y}_1 \cup \mathcal{Y}_2$, such that $\mathcal{Y}_1 \not\subseteq \mathcal{Y}_2$ and $\mathcal{Y}_2 \not\subseteq \mathcal{Y}_1$.

To define the notion of projective varieties and curves, we first need to introduce a topology on projective sets.

Definition 1.22 (Zarisky topology)**.** The *Zarisky topology* on the projective space $\mathbb{P}^n$ is defined by taking the open sets as the complement of projective sets.

Definition 1.23 (Projective variety)**.** A *projective variety* is an irreducible closed subset of $\mathbb{P}^n$, under the Zarisky topology. An open subset of a projective variety is referred to as a *quasi–projective* variety.

Later on, both projective or quasi–projective varieties will be called varieties. Given a variety $\mathcal{Y}$, we define its homogeneous ideal, denoted by $I(\mathcal{Y})$, as the ideal

$$I(\mathcal{Y}) = \{F \in \mathbb{F}_q[X_1, \cdots, X_{n+1}] \text{ homogeneous} \mid F(P) = 0,\ \forall P \in \mathcal{Y}\}.$$

Let $F, G \in \mathbb{F}_q[X_1, \cdots, X_{n+1}]$ be two homogeneous polynomials with same degree such that $G \notin I(\mathcal{Y})$. Then the fraction $\dfrac{F}{G} \in \mathbb{F}_q(X_1, \cdots, X_{n+1})$ is called a *rational function* on $\mathcal{Y}$. The elements $\frac{F}{G}$ and $\frac{F'}{G'}$ define the same rational function if the polynomial $FG' - F'G$ is identically zero on $\mathcal{Y}$.

Definition 1.24 (Function field of a variety)**.** The *function field* $\mathbb{F}_q(\mathcal{Y})$ of a variety $\mathcal{Y}$ is the field of rational functions on $\mathcal{Y}$, and the *dimension* of the variety $\mathcal{Y}$ is defined as the transcendence degree of $\mathbb{F}_q(\mathcal{Y})$ over $\mathbb{F}_q$.

We now have all the tools in hand to define the notion of projective curves.

Definition 1.25 (Projective curve)**.** A *projective curve* (or just curve) over the finite field $\mathbb{F}_q$, denoted by $\mathcal{X}/\mathbb{F}_q$ (or $\mathcal{X}$ when there is no ambiguity on the base field), is a variety of dimension one over $\mathbb{F}_q$, *i.e.* a variety whose function field $\mathbb{F}_q(\mathcal{X})$ has transcendence degree one over $\mathbb{F}_q$.

Example 1.26. In the affine plane over $\mathbb{F}_q$, we consider the variety $\mathcal{X}$ defined by the homogeneous polynomial $Y^3 - X^3 - Z^3$. Setting $x : X/Z$ and $y := Y/Z$, the function field $\mathbb{F}_q(\mathcal{X})$ of $\mathcal{X}$ consists in all elements of the form $\frac{P}{Q}$, with $P, Q \in \mathbb{F}_q[x, y]$. Since $y^3 = x^3 + 1$, the transcendence degree of $\mathbb{F}_q(\mathcal{X})$ over $\mathbb{F}_q$ is one, hence $\mathcal{X}$ is a projective curve.

In Definition 1.24, we saw that to any curve $\mathcal{X}/\mathbb{F}_q$, we can associate its function field. The theory of function fields will be studied in the next subsection. From now on, we only consider projective plane curves, *i.e.* curves $\mathcal{X} \subseteq \mathbb{P}^2$. This case is easier to understand and actually sufficient for the results presented in this thesis. It also helps in precising our definition of smoothness.

Definition 1.27 (Smooth curves)**.** Let $\mathcal{X}$ be a projective plane curve defined by some homogeneous polynomial $F \in \mathbb{F}_q[X, Y, Z]$. A point $P \in \mathcal{X}$ is said to be *non singular* if at least one of the partial derivatives $\frac{\partial F}{\partial X}$, $\frac{\partial F}{\partial Y}$ or $\frac{\partial F}{\partial Z}$ is not zero at P. The curve is *smooth* if all its points are non singular.

Example 1.28. Consider the projective plane curve of Example 1.26, defined by the homogeneous polynomial $F(X, Y, Z) = Y^3 - X^3 - Z^3$ over $\mathbb{F}_q$. The partial derivatives of F are $-3X^2, 3Y^2$ and $-3Z^2$, hence the curve is smooth whenever the characteristic of $\mathbb{F}_q$ is not 3.

Definition 1.29 (Local ring of a point)**.** Let $\mathcal{X}$ be a curve and $P \in \mathcal{X}$. A rational function $f \in \mathbb{F}_q(\mathcal{X})$ is *regular* at P if it can be written of the form $f = \frac{H}{G}$, with G non zero at P. The set of all regular functions at P forms a ring $\mathcal{O}_P$, called the *local ring* at P.

The terminology "local ring" makes sense since $\mathcal{O}_P$ is in fact a local ring, whose unique maximal ideal $\mathrm{m}_P \subseteq \mathcal{O}_P$ consists in all functions $f \in \mathcal{O}_P$ such that $f(P) = 0$.

In the next section, we will show that there exists a correspondence between smooth irreducible projective curves and algebraic function fields in one variable. In particular, this permits to transfer the notions from one language to the other. To conclude this section, we define two important objects which are specific to curves defined over finite fields.

Definition 1.30 (Rational points)**.** Let $\mathcal{X} \subseteq \mathbb{P}^n\left(\overline{\mathbb{F}_q}\right)$ be a curve defined over $\mathbb{F}_q$, *i.e.* its defining homogeneous polynomials have coefficients in $\mathbb{F}_q$. By definition, the points on $\mathcal{X}$ have coordinates in the algebraic closure $\overline{\mathbb{F}_q}$, but some of them may lie in $\mathbb{F}_q$ itself. These are called $\mathbb{F}_q$*–rational points* (or simply *rational points*). The set of all rational points of $\mathcal{X}$ is denoted by $\mathcal{X}(\mathbb{F}_q)$.

Later, we will see that the set of rational points of a curve is always finite, and that we can estimate its cardinality. It will be a crucial tool to define the family of algebraic geometry codes.

Example 1.31. The Klein curve $\mathcal{K}_3$ over $\mathbb{F}_4$ is defined by the homogeneous equation

$$X^3Y + Y^3Z + Z^3X = 0.$$

If we write $\mathbb{F}_4 = \{0, 1, \alpha, \alpha + 1\}$, then

$$\mathcal{K}_3(\mathbb{F}_4) = \{[0:0:1], [\alpha:\alpha+1:1], [\alpha+1, \alpha:1], [1:0:0], [0:1:0]\},$$

i.e. $\mathcal{K}_3$ has 5 $\mathbb{F}_4$–rational points.

Definition 1.32. A *closed point* of a projective plane curve $\mathcal{X}$ over $\mathbb{F}_q$ is an orbit under the Frobenius automorphism $\mathrm{Frob}_q : [x : y : z] \mapsto [x^q : y^q : z^q]$. Its *degree* is the cardinality of the orbit.

1.2.2 Algebraic function fields

For this section and without further details, we refer to [Sti09]. Below, we start by giving the one to one correspondence between smooth irreducible projective plane curves and function fields in one variable, which motivates the study of function field theory. Thanks to this, we then give an algebraic point of view of the notions presented in Section 1.2.1. Throughout this thesis, we will use either one or the other depending on the situation.

As in the previous section, we define objects over some finite field $\mathbb{F}_q$ even if it works for any arbitrary field.

Definition 1.33 (Algebraic function field)**.** An *algebraic function field* (or just *function field*) $K/\mathbb{F}_q$ of one variable over $\mathbb{F}_q$ is a field extension $K \supseteq \mathbb{F}_q$ which is a finite algebraic extension of $\mathbb{F}_q(x)$, where $x \in K$ is transcendental over $\mathbb{F}_q$.

Example 1.34. The simplest example of function field is the *rational function field*: an extension $K/\mathbb{F}_q$ is said to be rational if $K = \mathbb{F}_q(x)$ for some $x \in K$ which is transcendental over $\mathbb{F}_q$. In this case, each element $z \in \mathbb{F}_q(x)$ admits a unique representation

$$z = a \cdot \prod_i p_i(x)^{n_i},$$

where $0 \neq a \in \mathbb{F}_q$, the polynomials $p_i(x) \in \mathbb{F}_q[x]$ are monic, pairwise distinct and irreducible and $n_i \in \mathbb{Z}$.

From Definition 1.24, we know that the field of rational functions $\mathbb{F}_q(\mathcal{X})$ of a projective curve $\mathcal{X}$ is a function field in one variable. The converse is also true, as it is stated in the following theorem.

Theorem 1.35 ([Liu02, Proposition 7.3.13 and Remark 7.3.14])**.** *There is an anti–equivalence between the following two categories:*

- *smooth, irreducible, projective curves over $\mathbb{F}_q$, with non–constant morphisms of curves over $\mathbb{F}_q$,*
- *function fields over $\mathbb{F}_q$, with field morphisms over $\mathbb{F}_q$,*

which to a curve $\mathcal{X}$ associates its function field $K = \mathbb{F}_q(\mathcal{X})$.

The above theorem states that we can either deal with curves or function fields. Depending on the situation, we might prefer one to the other. In accordance with [Sti09, Section 1.4 ff.], when speaking about a function field K over $\mathbb{F}_q$, we always implicitly assume that the field of constants of K is equal to $\mathbb{F}_q$, *i.e.* $\mathbb{F}_q$ is algebraically closed in K. Alternatively, it means that all our curves will be geometrically irreducible.

Example 1.36. In Example 1.34, we defined the rational function field $\mathbb{F}_q(x)$ over $\mathbb{F}_q$. The curve associated to it according to Theorem 1.35 is the projective line $\mathbb{P}^1$ over $\mathbb{F}_q$.

Definition 1.37 (Valuation ring)**.** A *valuation ring* of $K/\mathbb{F}_q$ is a ring $\mathcal{O} \subseteq K$ such that

1. $\mathbb{F}_q \subsetneq \mathcal{O} \subsetneq K$, and
2. for any $x \in K$, we have $x \in \mathcal{O}$ or $x^{-1} \in \mathcal{O}$.

Proposition 1.38. *Let $\mathcal{O}$ be a valuation ring of $K/\mathbb{F}_q$. Then*

1. *$\mathcal{O}$ is a local ring, i.e. it has a unique maximal ideal $P := \mathcal{O}\backslash\mathcal{O}^\times$.*
2. *Let $0 \neq x \in K$. Then $x \in P \iff x^{-1} \notin \mathcal{O}$.*
3. *The maximal ideal P of $\mathcal{O}$ is principal.*

Proof. See [Sti09], Proposition 1.1.5 and Theorem 1.1.6. □

We now define the notion of places of a function field, which play the role of closed points in the case of projective curves over $\mathbb{F}_q$.

Definition 1.39 (Place)**.** A *place* P of a function field $K/\mathbb{F}_q$ is the maximum ideal of some valuation ring $\mathcal{O}$ of K. From Proposition 1.38 *2*, $\mathcal{O}$ is uniquely determined by P, thus we write

$$\mathcal{O}_P := \mathcal{O} = \left\{x \in K \mid x^{-1} \notin P\right\}.$$

Since P is principal, there exists $t \in \mathcal{O}_P$ such that $P = t\mathcal{O}_P$. Such an element is called a *local parameter* at P (or a *prime element* at P). The set of all places of K is denoted $\mathbb{P}_K$.

Remark 1.40. Let $\mathcal{X}$ be a smooth projective curve over $\mathbb{F}_q$ and $\mathbb{F}_q(\mathcal{X})$ its function field. There is a one–to–one correspondence between the set of places $\mathbb{P}_{\mathbb{F}_q(\mathcal{X})}$ and the set of closed points of $\mathcal{X}$. Thus, every place $P \in \mathbb{P}_{\mathbb{F}_q(\mathcal{X})}$ coincides with the local ring $\mathcal{O}_p$ of some point $p \in \mathcal{X}$. Hence, there is no ambiguity in the notation between local ring of a point and valuation ring of a place.

Proposition 1.41 ([Sti09, Theorem 1.1.6, (b)])**.** *Let $P \in \mathbb{P}_K$ and $t \in K$ be a local parameter at P. Then any $0 \neq x \in K$ has a unique representation of the form $x = t^n u$, with $n \in \mathbb{Z}$ and $u \in \mathcal{O}_P^\times$.*

Let us fix a place $P \in \mathbb{P}_K$, t a local parameter at P and $x \in \mathcal{O}_P\backslash\{0\}$ an element such that $x = t^n u$. Moreover, we denote by $\nu_P(x) := n$ the *valuation* of x at P (which does not depend on the choice of t). Setting $\nu_P(0) = \infty$, the function $\nu_P : K \to \mathbb{Z} \cup \{\infty\}$ is a discrete valuation of $K/\mathbb{F}_q$, *i.e.* it satisfies the following properties:

1. $\nu_P(x) = \infty \iff x = 0$.
2. For all $x, y \in K$, $\nu_P(xy) = \nu_P(x) + \nu_P(y)$.
3. For all $x, y \in K$, $\nu_P(x + y) \geq \min\{\nu_P(x), \nu_P(y)\}$, with equality if $\nu_P(x) \neq \nu_P(y)$.
4. There exists $z \in K$ such that $\nu_P(z) = 1$.
5. For all $a \in \mathbb{F}_q^*$, $\nu_P(a) = 0$.

Let $x \in K$ and $P \in \mathbb{P}_K$. If $\nu_P(x) > 0$, we say that P is a *zero* of x, and a *pole* of x if $\nu_P(x) < 0$.

Definition 1.42 (Residue field and degree)**.** Let $P \in \mathbb{P}_K$.

1. The field $F_P := \mathcal{O}_P/P$ is called the *residue class field* of P. Given $x \in \mathcal{O}_P$, its class in the quotient group F_P is denoted by $x(P)$.

2. The *degree* of P is defined by $\deg(P) := [F_P : K]$. A degree one place is referred to as a rational place.

The degree of a place is always finite (see [Sti09], Proposition 1.1.15). Again, there is a one–to–one correspondence between rational points on a curve $\mathcal{X}$ and degree one places in its function field $\mathbb{F}_q(\mathcal{X})$.

Example 1.43. [Sti09, Section 1.2] The rational function field $\mathbb{F}_q(x)$ over $\mathbb{F}_q$ has exactly $q+1$ places of degree one. According to Example 1.34, they corresponds to the set $\mathbb{P}^1(\mathbb{F}_q)$ of rational points on the projective line, given by

$$\mathbb{P}^1(\mathbb{F}_q) = \{[\alpha : 1] \mid \alpha \in \mathbb{F}_q\} \cup \{P_\infty\},$$

where P_∞ is the unique pole of x.

In what follows, we consider a finite extension L/K, where K is a function field over $\mathbb{F}_q$. We then recall that there exists a link between places in K and those in L: this is the *ramification* theory.

Definition 1.44. Let L/K be a finite extension of function fields over $\mathbb{F}_q$, and $Q \in \mathbb{P}_L, P \in \mathbb{P}_K$ two places. We say that Q *lie over* P (or that Q is an *extension* of P) if $P \subseteq Q$, in which case we write $Q|P$. In particular, $P = Q \cap K$.

Proposition 1.45 ([Sti09, Proposition 3.1.4])**.** *With notation as above, the following assertions are equivalent:*

1. $Q|P$.

2. *There exists an integer* $e := e(Q|P) \geq 1$ *such that for all* $x \in L$,

$$\nu_Q(x) = e(Q|P)\nu_P(x).$$

The finite integer $e(Q|P)$ is called the *ramification index* of $Q|P$. Moreover, the (also finite) integer $f(Q|P) := [F_Q : F_P]$ is called the *inertia degree* of Q over P. A place P is said to be *totally ramified* in L/K if there is only one place Q above P and $e(Q|P) = [L : K]$. Likewise, we say that P *totally splits* in L/K if P admits $[L : K]$ extensions in L, each with ramification index equals to one.

In this thesis, we will several times consider the following situation: let $L/\mathbb{F}_q$ be a function field and $\mathrm{Aut}(L)$ be the group of $\mathbb{F}_q$–automorphisms of L. For any finite subgroup $\mathcal{G} \subseteq \mathrm{Aut}(L)$, we denote by $L^{\mathcal{G}}$ the *fixed field* of L by $\mathcal{G}$, *i.e*

$$L^{\mathcal{G}} = \{x \in L \mid \sigma(x) = x \ , \ \forall \sigma \in \mathcal{G}\}.$$

In this case, the function field extension $L/L^{\mathcal{G}}$ is Galois with Galois group $\mathcal{G}$, and the ramification in the extension behaves with respect to the action of the Galois group $\mathcal{G}$:

Proposition 1.46 ([Sti09, Lemma 3.5.2])**.** *Let* $L/\mathbb{F}_q$ *be a function field and* $\sigma \in \mathrm{Aut}(L)$*. Let also* $Q \in L$ *and* $P \in L^{\langle\sigma\rangle}$ *be such that* $Q|P$*. Then* $\sigma(Q) := \{\sigma(x) \mid x \in Q\}$ *is a place of* L*, and:*

1. $\sigma(Q)|P$.

2. $e(\sigma(Q)|P) = e(Q|P)$.

Remark 1.47. If $\mathcal{X}$ is a smooth irreducible projective curve over $\mathbb{F}_q$ with function field F and $\mathcal{G} \subseteq \mathrm{Aut}(F)$, then the curve associated to the fixed field $F^{\mathcal{G}}$ is called the *quotient curve* of $\mathcal{X}$ by $\mathcal{G}$, denoted by $\mathcal{X}/\mathcal{G}$ (notice that we make the analogy between automorphisms of the curve and those of its function field, see Theorem 1.35).

Example 1.48. [Sti09, Annex A.13] Suppose that $n \mid q-1$ and let $L = \mathbb{F}_q(x,y)$ be a function field defined by the equation

$$y^n = p(x),$$

where $p \in \mathbb{F}_q[x]$ is a square–free polynomial of degree d prime to n. Then $L/\mathbb{F}_q(x)$ is cyclic of order n and its Galois group is given by

$$\mathrm{Gal}(L/\mathbb{F}_q(x)) = \{\sigma : y \mapsto \xi y \mid \xi \in \mu_n^*(\mathbb{F}_q)\}.$$

Such an extension is called a *Kummer extension*. Remark that if $\mathcal{Y}$ is the smooth projective curve associated to L, then the one associated to its fixed field $L^{\mathrm{Gal}(K/\mathbb{F}_q(x))} = \mathbb{F}_q(x)$ is the projective line $\mathbb{P}^1$ over $\mathbb{F}_q$.

1.2.3 Divisors and Riemann–Roch spaces

In this section, we choose to present the results from the algebraic point of view, meaning that we deal with function fields and places rather than curves and points. Further details can be found in [Sti09, Sections 1.4 and 1.5].

From now on, fix a function field K over $\mathbb{F}_q$. As discussed above and following [Sti09, Section 1.4 ff.], we assume that $\mathbb{F}_q$ is algebraically closed in K. Keep in mind that the remaining of Section 1.2 can be applied by replacing K with its associated geometrically irreducible curve $\mathcal{X}$.

Definition 1.49 (Divisors)**.** The *divisor group* of K, denoted by $\mathrm{Div}(K)$, is the free abelian group generated by the places of K. The elements of $\mathrm{Div}(K)$ are called *divisors*. In other words, any divisor $D \in \mathrm{Div}(K)$ is a formal sum

$$D = \sum_{P \in \mathbb{P}_K} n_P P,$$

where $n_P \in \mathbb{Z}$ are all zero but finitely many. The *support* of D is defined by

$$\mathrm{Supp}(D) := \{P \in \mathbb{P}_K \mid n_P \neq 0\}.$$

Given a place $P \in \mathbb{P}_K$, we set $\nu_P(D) := n_P$. The *degree* of the divisor D is defined by

$$\deg(D) := \sum_{P \in \mathrm{Supp}(D)} \nu_P(D) \deg(P).$$

The group $\mathrm{Div}(K)$ is endowed with a partial order in the following way:

$$\sum_{P \in \mathbb{P}_K} n_P P \geq \sum_{P \in \mathbb{P}_K} m_P P \iff \forall Q \in \mathbb{P}_K, n_P \geq m_P.$$

If 0 denotes the zero divisor in $\mathrm{Div}(K)$ (*i.e* all $\nu_P(0) = 0$), then we say that the divisor D is *effective* if $D \geq 0$. It is well–known that any non zero function $x \in K$ has only finitely many poles and zeros in $\mathbb{P}_K$ (see for example [Sti09, Corollary 1.3.4]). Hence, the following definition makes sense.

Definition 1.50 (Principal divisors)**.** Let $0 \neq x \in K$ and denote by $Z(x)$ (resp. $N(x)$) the set of zeros (resp. poles) of x in $\mathbb{P}_K$. We define

$$\begin{aligned}(x)_0^K &:= \sum_{P \in Z(x)} \nu_P(x) P, \text{ the } \textit{zero divisor} \text{ of } x,\\ (x)_\infty^K &:= \sum_{P \in N(x)} (-\nu_P(x)) P, \text{ the } \textit{pole divisor} \text{ of } x \text{ and}\\ (x)^K &= (x)_0^K - (x)_\infty^K \text{ the } \textit{principal divisor} \text{ of } x.\end{aligned}$$

The subset of $\mathrm{Div}(K)$ generated by the principal divisors of K forms a subgroup, denoted by $\mathrm{Princ}(K)$. Given two functions $x, y \in K$, we have

$$(xy)^K = (x)^K + (y)^K.$$

Whenever it is clear from the context, we just write (x) instead of $(x)^K$ to talk about the principal divisor of $x \in K$ (the same holds for $(x)_0$ and $(x)_\infty$). Note that $(x)_0 \geq 0$, $(x)_\infty \geq 0$ and

$$(x) = \sum_{P \in \mathbb{P}_K} \nu_P(x) P.$$

The non zero elements lying in $\mathbb{F}_q$ are characterized by

$$x \in \mathbb{F}_q \iff (x) = 0.$$

Example 1.51. Let $\mathbb{F}_q(x)$ be the rational function field over $\mathbb{F}_q$ and $f \in \mathbb{F}_q[x]$ a polynomial which splits in $\mathbb{F}_q$. Denote by $\{\alpha_1, ..., \alpha_s\}$ its roots in $\mathbb{F}_q$ and by m_i their multiplicity. The divisor of f is then given by

$$(f)^{\mathbb{F}_q(x)} = \sum_{i=1}^{s} m_i P_i - \deg(f) P_\infty,$$

where P_i a degree one place in K, associated to the point $[\alpha_i : 1]$ in the projective line (see Example 1.43 for a description of $\mathbb{P}^1(\mathbb{F}_q)$). The place P_∞ is the unique pole of x.

Theorem 1.52 ([Sti09, Theorem 1.4.11])**.** *All principal divisors of K have degree zero. More precisely, for any $x \in K$, we have*

$$\deg((x)_0) = \deg((x)_\infty) = [K : \mathbb{F}_q(x)].$$

Definition 1.53 (Divisor class group)**.** The factor group

$$\mathrm{Cl}(K) := \mathrm{Div}(K)/\,\mathrm{Princ}(K)$$

is called the *divisor class group* of K. For any $D \in \mathrm{Div}(K)$, we denote by $[D]$ its class in $\mathrm{Cl}(K)$. We say that two divisors D_1 and D_2 are *equivalent*, and we write $D_1 \sim D_2$, if $[D_1] = [D_2]$, *i.e.* $D_2 = D_1 + (x)$ for some $x \in K$.

Remark 1.54. As any principal divisor has degree zero, two equivalent divisors have same degree.

Let $\mathrm{Div}^0(K)$ be the subgroup of $\mathrm{Div}(K)$ made of all degree zero divisors, and

$$\mathrm{Cl}^0(K) := \mathrm{Div}^0(K)/\,\mathrm{Princ}(K)$$

the group of divisor class of degree zero. (which makes sense thanks to Theorem 1.52).

Proposition 1.55 ([Sti09, Proposition 5.1.3])**.** $\mathrm{Cl}^0(K)$ *is a finite group, and its order $h(K) := \#\,\mathrm{Cl}^0(K)$ is called the class number of K.*

For any $r \geq 1$, the number of divisor classes in $\mathrm{Cl}(K)$ of degree r does not depend on r, and is equal to $h(K)$. In Proposition 1.70, we give an estimation of this integer.

In the case of a finite extension of function fields L/K, we can define a specific divisor in L, which keeps track of all the ramification in the extension.

Definition 1.56. Let L/K be a finite extension of function fields. The divisor

$$\mathrm{Diff}(L/K) = \sum_{P \in \mathbb{P}_K} \sum_{Q|P} d(Q|P) \cdot Q. \tag{1.2}$$

is called the *different* of L/K, where $d(Q|P)$ is the *different exponent* of Q over P (see [Sti09, Definition 3.4.3]).

Our next subject of interest will play a fundamental role in the thesis, being key to define the so–called Algebraic Geometry (AG) codes, which will be done in Section 1.3.

Definition 1.57 (Riemann–Roch space)**.** For a divisor $D \in \mathrm{Div}(K)$, we define its *Riemann–Roch* space as the $\mathbb{F}_q$–vector space

$$\mathcal{L}_K(D) := \{x \in K^* \mid (x) \geq -D\} \cup \{0\}.$$

Its dimension $\ell(D) := \dim_{\mathbb{F}_q} \mathcal{L}_K(D)$ is called the *dimension* of the divisor D.

Without ambiguity on the function field, we simply write $\mathcal{L}(D)$ instead of $\mathcal{L}_K(D)$.

Remark 1.58 ([Sti09], Lemma 1.4.6). The Riemann–Roch spaces of two equivalent divisors are isomorphic as $\mathbb{F}_q$–vector spaces.

Example 1.59. Let us consider the pole P_∞ of x in the rational function field $\mathbb{F}_q(x)$. Then for any integer $k > 0$, the space $\mathcal{L}(kP_\infty)$ has dimension $k + 1$, and

$$\mathcal{L}(kP_\infty) = \left\langle 1, x, \cdots, x^k \right\rangle_{\mathbb{F}_q}.$$

These spaces are usually used to build Reed–Solomon codes.

The computation of the dimension $\ell(D)$ of a divisor is very important for us as it gives information on the parameters of AG codes. However, this is usually a difficult task and we are often only able to give a lower bound on it. A first one is given by the following lemma:

Lemma 1.60 ([Sti09, Proposition 1.4.9])**.** *For any $D \in \mathrm{Div}(K)$, we have*

$$\ell(D) \leq \max\{0, \deg(D) + 1\}.$$

In particular, if $\deg(D) < 0$, then $\ell(G) = 0$.

To better handle the dimension of Riemann–Roch spaces, we need to introduce first the notion of differential forms.

1.2.4 Differentials and the Riemann–Roch theorem

The main purpose of this section is to present the Riemann-Roch theorem, which is a powerful tool that gives an explicit formula for the dimension of any divisor. Details for this section can be found in [Sti09, Sections 1.5 and 4.1].

Let K be a function field over $\mathbb{F}_q$. A *derivation* of K is a $\mathbb{F}_q$–linear map $\delta : K \to K$ satisfying the product rule

$$\delta(xy) = x\delta(y) + y\delta(x), \ \forall x, y \in K.$$

We denote by $\mathrm{Der}(K)$ the K–vector space of derivations over K. If we are given a *separating element* $x \in K$ (*i.e.* such that $K/\mathbb{F}_q(x)$ is finite and algebraic), then there exists a unique derivation $\delta_x \in \mathrm{Der}(K)$ such that $\delta_x(x) = 1$.

Lemma 1.61 ([Sti09, Lemma 4.1.6])**.** $\mathrm{Der}(K)$ *is a one–dimensional K–vector space. In particular, for every $\eta \in \mathrm{Der}(K)$, we have $\eta = \eta(x)\delta_x$.*

The last step before defining differentials is to consider the set

$$\mathcal{Z} := \{(u, x) \in K \times K \mid x \text{ is separating}\},$$

on which we define an equivalence relation $\sim$ by

$$(u, x) \sim (v, y) \iff v = u\delta_y(x).$$

Definition 1.62. For a couple $(u, x) \in \mathcal{Z}$, its class under the relation $\sim$ is denoted by udx, and is called a *differential form* (or just *differential*) of K. We simply write dx for the class of $(1, x)$, and the set of all differentials of K is denoted by Ω_K.

The main properties of differentials are put together in the following proposition:

Proposition 1.63 ([Sti09, Propositions 4.1.8 and 1.5.13])**.** *With notation as above, we have:*

1. *$x \in K$ is a separating element $\iff$ $dx \neq 0$.*
2. *Every differential $\omega \in \Omega_K$ can be uniquely written in the form $\omega = hdx$, with $h \in K$. In particular, $\dim_K \Omega_K = 1$.*
3. *If $0 \neq \omega \in \Omega_K$ and if t_P is a prime element at the place $P \in \mathbb{P}_K$, then there exists $u \in K$ such that $\omega = udt_P$.*

Definition 1.64. Let $\omega \in \Omega_K$ and $P \in \mathbb{P}_K$. We define the *valuation* of ω at P by

$$\nu_P(\omega) := \nu_P(u), \text{ if } \omega = udt_P.$$

As we associated a divisor to any function $x \in K$, we can associate to a differential $\omega \in \Omega_K$ the divisor $(\omega)^K$ (or just (ω) if it is clear from the context), defined by

$$(\omega)^K = \sum_{P \in \mathbb{P}_K} \nu_P(\omega) P.$$

Such a divisor is called a *canonical divisor*, and it does not depend on the choice of the prime elements t_P's. If $\omega = hdx$ with $h \in K$ (see Proposition 1.63, *2*), then we have

$$(\omega)^K = (h)^K + (dx)^K,$$

where

$$(dx)^K = -2(x)_\infty^K + \mathrm{Diff}(K/\mathbb{F}_{q^m}(x)). \tag{1.3}$$

Remark 1.65. Since $\dim_K \Omega_K = 1$, we easily see that canonical divisors on K are equivalent.

Definition 1.66. The *genus* $\mathfrak{g}(K)$ (or just $\mathfrak{g}$) of K is defined as the dimension of any canonical divisor, *i.e.*

$$\mathfrak{g}(K) := \ell(W), \text{ for a canonical divisor } W = (\omega)^K \in \mathrm{Div}(K).$$

We are now ready to present the Riemann–Roch theorem.

Theorem 1.67 (Riemann–Roch, [Sti09, Theorem 1.5.15])**.** *Let $\omega \in \Omega_K$ and $W := (\omega)^K$ its divisor. Then for any $D \in \mathrm{Div}(K)$, we have*

$$\ell(A) = \deg(A) + 1 - \mathfrak{g}(K) + \ell(W - A).$$

This result can be precised in the case of canonical divisors:

Corollary 1.68 ([Sti09, Corollary 1.5.16])**.** *For any canonical divisor $W \in \mathrm{Div}(K)$, we have*

$$\deg(W) = 2\mathfrak{g}(K) - 2.$$

Theorem 1.69 ([Sti09, Theorem 1.5.17])**.** *If $D \in \mathrm{Div}(K)$ is a divisor such that $\deg(D) \geq 2\mathfrak{g}(K) - 1$, then*

$$\ell(D) = \deg(D) + 1 - \mathfrak{g}(K).$$

The genus of a function field is a powerful tool that can be used to estimate the class number or the number of rational points:

Proposition 1.70 ([TVN07, Proposition 3.1.22])**.** *Let K be a function field over a finite field $\mathbb{F}$. Then*

$$(\sqrt{|\mathbb{F}|} - 1)^{2\mathfrak{g}(K)} \leq h(K) \leq (\sqrt{|\mathbb{F}|} + 1)^{2\mathfrak{g}(K)}. \tag{1.4}$$

This implies in particular that any rational function field has only one divisor class, since it has genus zero.

Theorem 1.71 (Hasse–Weil Bound)**.** *Let K be a function field over $\mathbb{F}_q$ and $\mathcal{X}$ the corresponding curve with respect to Theorem 1.35. Then the number $\mathcal{X}(\mathbb{F}_q)$ of rational points on $\mathcal{X}$ is bounded by*

$$|\mathcal{X}(\mathbb{F}_q)| \leq q + 1 + 2\mathfrak{g}(K)\sqrt{q}.$$

A proof of this famous theorem can be found in [Sti09, Theorem 5.2.3]. Usually, the genus of a function field is hard to compute. In the case of an extension L/K, we have a formula that links $\mathfrak{g}(L)$ and $\mathfrak{g}(K)$.

Theorem 1.72 (Hurwitz Genus Formula, [Sti09, Theorem 3.4.13])**.** *Let L/K be a finite extension of function fields over $\mathbb{F}_{q^m}$. Then*

$$2\mathfrak{g}(L) - 2 = [L : K] \cdot (2\mathfrak{g}(K) - 2) + \deg\left(\mathrm{Diff}(L/K)\right),$$

where the different $\mathrm{Diff}(L/K)$ is defined in Definition 1.56.

We finish this section by giving an overview of a well–known class of curves, namely $C_{a,b}$ curves.

1.2.5 $C_{a,b}$ curves

As a complement to the upcoming discussion, we refer the reader to [Miu93].

Definition 1.73. Let a, b be coprime integers. A $C_{a,b}$ curve over $\mathbb{F}_q$ is a curve having an irreducible, affine and non singular plane model with equation

$$f_{a,b}(x, y) = \alpha_{0a}y^a + \alpha_{b0}x^b + \sum \alpha_{ij}x^i y^j = 0, \tag{1.5}$$

where $f_{a,b} \in \mathbb{F}_q[X, Y]$ and the sum is taken over all couples $(i, j) \in \{0, \cdots, b\} \times \{0, \cdots, a\}$ such that $ai + bj < ab$.

Let $\mathcal{X}_{a,b}$ be such a curve, defined over $\mathbb{F}_q$. Its genus is given by

$$\mathfrak{g}_{a,b} := \mathfrak{g}(\mathcal{X}_{a,b}) = \frac{(a-1)(b-1)}{2}.$$

The common point of all these curves is their behaviour with respect to the points at infinity. In fact, the condition imposed on the leading monomial of $f_{a,b}$ implies that $C_{a,b}$ curves have only one point at infinity, say P_∞. More importantly, we know a nice basis of the Riemann–Roch space associated to any multiple of this point, *i.e.* for any non negative integer s, we have:

$$\mathcal{L}(sP_\infty) = \text{Span}\left(x^i y^j \mid 0 \leq i, 0 \leq j \leq a-1 \text{ and } ai + bj \leq s\right). \tag{1.6}$$

This particularity will be used several times throughout this thesis. One could think that Equation (1.5) is kinda restrictive, but the class of $C_{a,b}$ is quite general: for example, any Kummer type curve (see Example 1.48) is a particular case of $C_{a,b}$ curve. It is also the case for the so–called Artin–Schreier curves.

1.3 AG codes and their subfield subcodes

In this section, we define algebraic geometry (AG) codes and their subfield subcodes (SSAG). Without specific mention, we refer to [TVN07] and [Sti09].

1.3.1 Algebraic Geometry codes

We consider a finite extension $\mathbb{F}_{q^m}$ of $\mathbb{F}_q$, and we construct codes over $\mathbb{F}_{q^m}$, before considering subfield subcodes over $\mathbb{F}_q$.

In what follows, $\mathcal{X}$ denotes an irreducible and smooth projective curve over $\mathbb{F}_{q^m}$ and K its function field. The genus of $\mathcal{X}$ is the one of its function field, *i.e.* $\mathfrak{g}(\mathcal{X}) := \mathfrak{g}(K)$. Let $\mathcal{P} = \{P_1, \cdots, P_n\}$ be a set of n distinct rational points on $\mathcal{X}$ and $G \in \text{Div}(\mathcal{X})$ be a divisor of degree less than n such that $\text{Supp}(G) \cap \mathcal{P} = \varnothing$. We consider the evaluation map

$$\text{ev}_{\mathcal{P}} : \begin{cases} \mathcal{L}(G) \longrightarrow \mathbb{F}_{q^m}^n \\ \quad f \longmapsto (f(P_1), \cdots, f(P_n)). \end{cases} \tag{1.7}$$

Definition 1.74 (AG code)**.** With previous notation, the *algebraic geometry* (AG) code on $\mathcal{X}$ associated to the support $\mathcal{P}$ and the divisor G is defined by

$$C_{\mathcal{L}}(\mathcal{X}, \mathcal{P}, G) := \{\text{ev}_{\mathcal{P}}(f) \mid f \in \mathcal{L}(G)\} \subseteq \mathbb{F}_{q^m}^n.$$

Let $k := \ell(G)$ and $\{f_1, \cdots, f_k\}$ be a basis of $\mathcal{L}(G)$. Then the matrix

$$\mathbf{M} = \begin{pmatrix} f_1(P_1) & \cdots & f_1(P_n) \\ \vdots & \ddots & \vdots \\ f_k(P_1) & \cdots & f_k(P_n) \end{pmatrix} \in \mathbb{F}_{q^m}^n \tag{1.8}$$

is a generator matrix of $C_{\mathcal{L}}(\mathcal{X}, \mathcal{P}, G)$.

Using the Riemann–Roch theorem (see. Theorem 1.67), we can estimate the dimension and the minimum distance of AG codes.

Theorem 1.75 ([Sti09, Corollary 2.2.3])**.** *If $n > \deg(G)$, $C_{\mathcal{L}}(\mathcal{X}, \mathcal{P}, G)$ is an $[n, k, d]_{q^m}$ code, with*

$$k = \ell(G) \geq \deg(G) + 1 - \mathfrak{g}(\mathcal{X}) \text{ and } d \geq n - \deg(G).$$

Moreover, if $\deg(G) \geq 2\mathfrak{g}(\mathcal{X}) - 1$, then $k = \deg(G) + 1 - \mathfrak{g}(\mathcal{X})$. The Goppa designed distance for the AG code $C_{\mathcal{L}}(\mathcal{X}, \mathcal{P}, G)$ is defined by $d^* := n - \deg(G)$.

1.3.2 Duality

In the previous section, we defined AG codes associated to a support $\mathcal{P}$ and a divisor G. Actually, there exists another kind of AG codes that can be constructed from the same support and divisor, using differential forms introduced in Section 1.2.4.

Definition 1.76 (Special divisor)**.** Let $G \in \mathrm{Div}(\mathcal{X})$. We associate to G the subspace $\Omega_K(G) \subseteq \Omega_K$ of differential forms defined by

$$\Omega_K(G) := \left\{\omega \in \Omega_K^* \mid (\omega)^K \geq G\right\} \cup \{0\}.$$

This space is a vector space over $\mathbb{F}_{q^m}$, whose dimension $i(G) := \dim_{\mathbb{F}_{q^m}} \Omega_K(G)$ is called the *index of speciality* of G. We say that G is *non–special* if $i(G) = 0$; otherwise G is called *special*.

Remark 1.77. Notice that the definition of the space $\Omega_K(G)$ is similar to the one of Riemann–Roch spaces, which is not surprising as these objects play similar roles in terms of AG codes.

Theorem 1.78 ([Sti09, Theorem 1.5.14])**.** *Let $G \in \mathrm{Div}(\mathcal{X})$ be any divisor and $W = (\omega)$ be a canonical divisor. Then the map*

$$\begin{cases} \mathcal{L}(W-G) \to \Omega_K(G) \\ \qquad f \longmapsto f\omega \end{cases}$$

is an isomorphism of $\mathbb{F}_{q^m}$–vector spaces. In particular, $i(G) = \ell(G) - \deg(G) + \mathfrak{g}(K) - 1$.

Definition 1.79 (Residue)**.** Let $\omega \in \Omega_K$, P a degree one place in $K = \mathbb{F}_{q^m}(\mathcal{X})$ and t_P a local parameter at P such that $\omega = f dt_P$, with $f \in K$. The expansion of f into a Laurent series in t_P has the form

$$f = \sum_{i=-r}^{\infty} a_i t_P^i, \text{ with } r \in \mathbb{Z}.$$

The *residue* $\mathrm{Res}_\omega(P)$ of ω at P is then defined by

$$\mathrm{Res}_\omega(P) := a_{-1}.$$

The above definition does not depend on the choice of the local parameter t_P. We now have all the tools to define the other kind of AG codes, namely Ω–AG codes. As in Section 1.3.1, $\mathcal{P} = \{P_1, \cdots, P_n\}$ denotes a set of n distinct rational points on $\mathcal{X}$ and $G \in \mathrm{Div}(\mathcal{X})$ a divisor of degree less than n such that $\mathrm{Supp}(G) \cap \mathcal{P} = \varnothing$. The residue map is defined by

$$\mathrm{Res}_{\mathcal{P}} : \begin{cases} \Omega_K \longrightarrow \mathbb{F}_{q^m}^n \\ \omega \longmapsto (\mathrm{Res}_\omega(P_1), \cdots, \mathrm{Res}_\omega(P_n)). \end{cases}$$

Definition 1.80 (Ω–AG codes)**.** With above notation, the Ω–AG code (or *differential* code) on $\mathcal{X}$ associated with the support $\mathcal{P}$ and the divisor G is defined by

$$C_\Omega(\mathcal{X}, \mathcal{P}, G) := \{\mathrm{Res}_{\mathcal{P}}(\omega) \mid \omega \in \Omega(G - D_{\mathcal{P}})\},$$

where $D_{\mathcal{P}} = \sum_{P \in \mathcal{P}} P$.

Again, the Riemann–Roch theorem allows us to estimate the parameters of Ω–AG codes:

Theorem 1.81 ([Sti09, Theorem 2.2.7])**.** *$C_\Omega(\mathcal{X}, \mathcal{P}, G)$ is an $[n, k, d]_{q^m}$ code, with*

$$k = i(G - D_{\mathcal{P}}) \geq n - \deg(G) + \mathfrak{g}(\mathcal{X}) - 1 \text{ and } d \geq \deg(G) + 2 - 2\mathfrak{g}(\mathcal{X}).$$

In addition, if $\deg(G) \geq 2\mathfrak{g}(\mathcal{X}) - 1$, then $k = n - \deg(G) + \mathfrak{g}(\mathcal{X}) - 1$.

The link between both constructions of AG codes is given by the following theorem:

Theorem 1.82. *The codes $C_{\mathcal{L}}(\mathcal{X}, \mathcal{P}, G)$ and $C_\Omega(\mathcal{X}, \mathcal{P}, G)$ are dual to each other, i.e.*

$$C_{\mathcal{L}}(\mathcal{X}, \mathcal{P}, G)^\perp = C_\Omega(\mathcal{X}, \mathcal{P}, G).$$

A proof of this statement can be found in [Sti09, Theorem 2.2.8], and mainly relies on the residue formula ([Sti09, Corollary 4.3.3]).

The next and final result of this section shows that any Ω–AG code $C_\Omega(\mathcal{X}, \mathcal{P}, G)$ can be represented as a code $C_{\mathcal{L}}(\mathcal{X}, \mathcal{P}, H)$ for some explicit divisor $H \in \mathrm{Div}(\mathcal{X})$ (up to diagonal equivalence). This can be helpful to deal with dual of AG codes without having to rely on differentials.

Proposition 1.83 ([Sti09, Proposition 2.2.10])**.** *Let $C_{\mathcal{L}}(\mathcal{X}, \mathcal{P}, G)$ be an AG code defined on a curve $\mathcal{X}$. Then*

$$C_{\mathcal{L}}(\mathcal{X}, \mathcal{P}, G)^\perp = C_{\mathcal{L}}(\mathcal{X}, \mathcal{P}, G^\perp) \cdot \mathbf{Z}_\omega,$$

with $G^\perp = D_{\mathcal{P}} - G + W$, where $W := (\omega)^K$ is the divisor of some differential $\omega \in \Omega_K$ such that for all $P \in \mathcal{P}$, $\nu_P(\omega) = -1$, and $\mathbf{Z}_\omega$ is the diagonal matrix those coefficients are the $\mathrm{Res}_\omega(P)$, $P \in \mathcal{P}$.

The divisor $G^\perp$ is referred to as the *dual divisor* of G. In [Sti09, Lemma 2.2.9], it is also proven that there exists a differential $\omega \in \Omega_K$ such that $\mathbf{Z}_\omega$ is the identity matrix. Note that the dual divisor is not well–defined, as it depends on the choice of the differential. Throughout this thesis, while using this notation, there will be no ambiguity, as a suitable choice of a differential will be made.

1.3.3 Subfield subcode of AG codes and their parameters

We keep the notation of Section 1.3.1. Here, we define subfield subcode of AG codes (SSAG in short), which will be used a lot in this thesis as they are good candidates to replace classical Goppa codes in the McEliece cryptosystem [McE78]. We also recall known results about their parameters.

Definition 1.84 (SSAG codes)**.** Let $C_{\mathcal{L}}(\mathcal{X}, \mathcal{P}, G)$ be an AG code over $\mathbb{F}_{q^m}$ as in Definition 1.74. We define its subfield subcode over $\mathbb{F}_q$, denoted by $\mathsf{SSAG}_q(\mathcal{X}, \mathcal{P}, G)$, as follows:

$$\mathsf{SSAG}_q(\mathcal{X}, \mathcal{P}, G) := C_{\mathcal{L}}(\mathcal{X}, \mathcal{P}, G)|_{\mathbb{F}_q} = C_{\mathcal{L}}(\mathcal{X}, \mathcal{P}, G) \cap \mathbb{F}_q^n.$$

For SSAG codes, there is an obvious upper bound on the dimension, which is

$$\dim_{\mathbb{F}_q} \mathsf{SSAG}_q(\mathcal{X}, \mathcal{P}, G) \leq \dim_{\mathbb{F}_{q^m}} C_{\mathcal{L}}(\mathcal{X}, \mathcal{P}, G),$$

coming from the fact that any basis of $C_{\mathcal{L}}(\mathcal{X}, \mathcal{P}, G)$ over $\mathbb{F}_{q^m}$ remains free when restricted to the subfield $\mathbb{F}_q$. In general, it is hard to find the true dimension of an SSAG code, but a trivial estimate can be derived from Delsarte's theorem (Theorem 1.12):

$$\dim_{\mathbb{F}_q} \mathsf{SSAG}_q(\mathcal{X}, \mathcal{P}, G) \geq n - m \dim_{\mathbb{F}_{q^m}} C_{\Omega}(\mathcal{X}, \mathcal{P}, G). \tag{1.9}$$

As for the minimum distance, it is at least the one of the AG code $C_{\mathcal{L}}(\mathcal{X}, \mathcal{P}, G)$, hence bounded from below by the designed distance $d^* = n - \deg(G)$. Additionally, the structure of AG codes may provide sharper bounds on the dimension of subfield subcodes of Ω–AG codes and trace codes of AG codes.

Theorem 1.85 ([Sti09, Theorem 9.1.6])**.** *With above notation, let $G_1 \in \mathcal{X}$ be a divisor such that*

$$G \geq qG_1 \text{ and } G \geq G_1. \tag{1.10}$$

Then

$$\dim_{\mathbb{F}_q} \mathrm{Tr}_{\mathbb{F}_{q^m}/\mathbb{F}_q}(C_{\mathcal{L}}(\mathcal{X}, \mathcal{P}, G)) \leq \begin{cases} m(\ell(G) - \ell(G_1)) + 1 & \text{if } G_1 \geq 0, \\ m(\ell(G) - \ell(G_1)) & \text{if } G_1 \not\geq 0, \end{cases}$$

and

$$\dim_{\mathbb{F}_q} C_{\Omega}(\mathcal{X}, \mathcal{P}, G)|_{\mathbb{F}_q} \geq \begin{cases} n - 1 - m(\ell(G) - \ell(G_1)) & \text{if } G_1 \geq 0, \\ n - m(\ell(G) - \ell(G_1)) & \text{if } G_1 \not\geq 0. \end{cases}$$

Remark 1.86. The biggest divisor (with respect to the degree) satisfying both conditions in Equation (1.10) is given by

$$\left\lceil \frac{G}{q} \right\rceil := \sum_{P \in \mathrm{Supp}(G^+)} \left\lfloor \frac{\nu_P(G^+)}{q} \right\rfloor P + \sum_{P \in \mathrm{Supp}(G^-)} \nu_P(G^-)P, \tag{1.11}$$

where G^+ and G^- are effective divisors such that $G = G^+ - G^-$.

With further hypotheses on G and $\left\lceil \frac{G}{q} \right\rceil$, [LC16, Theorem 1] gives an exact formula for the dimension of such codes. Regarding the parameters of subfield subcodes of differential codes, Wirtz [Wir88] improved the bound on their the minimum distance.

Theorem 1.87 ([Wir88, Theorem 2])**.** *With the same notation as in Theorem 1.85, assume that* $\deg(G_1) \geq 2\mathfrak{g} - 1$. *Set* $U := \{P \in \mathrm{Supp}(G) \mid \nu_P(G) \geq 0 \text{ and } \nu_P(G) = q - 1 \mod q\}$ *and* $G_U = \sum_{P \in U} P$. *Then*

$$\dim_{\mathbb{F}_q} C_{\Omega}(\mathcal{X}, \mathcal{P}, G)|_{\mathbb{F}_q} = \dim_{\mathbb{F}_q} C_{\Omega}(\mathcal{X}, \mathcal{P}, G + G_U)|_{\mathbb{F}_q},$$

hence the minimum distance of $C_{\Omega}(\mathcal{X}, \mathcal{P}, G)|_{\mathbb{F}_q}$ satisfies

$$d\left(C_{\Omega}(\mathcal{X}, \mathcal{P}, G)|_{\mathbb{F}_q}\right) \geq \deg G + \deg G_U - 2\mathfrak{g} + 2.$$

The next section is dedicated to a well–known class of AG codes, namely *Generalized Reed–Solomon* codes.

1.3.4 The family of Generalized Reed–Solomon codes

In the discussion below, we recall the definitions and some properties of Generalized Reed–Solomon (GRS) codes and their subfield subcodes. In the following, the dimension of a GRS is denoted by r, and our codes are still defined over the finite field $\mathbb{F}_{q^m}$.

Definition 1.88. Let $\mathbf{x} = (x_1, \ldots, x_n) \in \mathbb{F}_{q^m}^n$ be a vector with pairwise distinct entries and $\mathbf{y} = (y_1, \ldots, y_n) \in \mathbb{F}_{q^m}^n$ be a vector with nonzero entries. The $[n, r]_{q^m}$ generalized Reed–Solomon code with support $\mathbf{x}$ and multiplier $\mathbf{y}$ is defined by

$$\mathsf{GRS}_r(\mathbf{x}, \mathbf{y}) = \{(y_1 f(x_1), \ldots, y_n f(x_n)) \mid P \in \mathbb{F}_{q^m}[T],\ \deg f < r\}.$$

It is clear that the Reed–Solomon code RS_r introduced in Example 1.6 is a particular case of GRS code: in fact, taking $\mathbf{x}_\beta = \left\{\beta, \beta^2, \ldots, \beta^{q^m-1}\right\}$ where β is a generator of $\mathbb{F}_{q^m}^*$, we have

$$\mathsf{RS}_r = \mathsf{GRS}_r(\mathbf{x}_\beta, \mathbf{1}),$$

where $\mathbf{1}$ stands for the unit vector.

It is possible to represent any GRS code as an AG code defined over the projective line: in fact, let $\mathsf{GRS}_r(\mathbf{x}, \mathbf{y})$ be a GRS code as above. On the rational function field $\mathbb{F}_{q^m}(x)$, denote by P_i the zero of $x - x_i$, and consider the set of degree one places $\mathcal{P} = \{P_1, \ldots, P_n\} \in \mathbb{P}_{\mathbb{F}_{q^m}(x)}$. Using Lagrange's interpolation, we can find a polynomial $h \in \mathbb{F}_{q^m}[T]$ such that

$$h(P_i) = y_i \text{ for all } i \in \{1, \ldots, n\} \text{ and } \deg h < n.$$

Then

$$\mathsf{GRS}_r(\mathbf{x}, \mathbf{y}) = C_{\mathcal{L}}(\mathbb{P}^1, \mathcal{P}, (r-1)P_\infty + (h)^{\mathbb{F}_{q^m}(x)}),$$

where P_∞ is the pole of x. From Theorem 1.82, Proposition 1.83 and the above discussion, we know that the dual of any GRS code is also a GRS code. More precisely, we have:

Proposition 1.89. *Let $\mathbf{x}, \mathbf{y}$ be a support and multiplier of length n and $r \leq n$. Then*

$$\mathsf{GRS}_r(\mathbf{x}, \mathbf{y})^\perp = \mathsf{GRS}_r(\mathbf{x}, \mathbf{y}^\perp),$$

with

$$\mathbf{y}^\perp := \left(\frac{1}{p'_{\mathbf{x}}(x_1)y_1}, \ldots, \frac{1}{p'_{\mathbf{x}}(x_n)y_n}\right),$$

where $p'_{\mathbf{x}}$ is the derivative of the polynomial $p_{\mathbf{x}}(T) := \prod_{i=1}^{n}(T - x_i) \in \mathbb{F}_{q^m}[T]$.

Up to the dual operation, the subfield subcode of a GRS code is referred to as an alternant code.

Definition 1.90. With above notation, we define the alternant code over $\mathbb{F}_q$ associated with the support $\mathbf{x}$ and multiplier $\mathbf{y}$ as

$$\mathcal{A}_{r,q}(\mathbf{x}, \mathbf{y}) := \mathsf{GRS}_r(\mathbf{x}, \mathbf{y})^\perp|\mathbb{F}_q.$$

The integer r is called the *order* of the alternant code.

From Delsarte's theorem and by duality, we also have

$$\mathcal{A}_{r,q}^\perp(\mathbf{x}, \mathbf{y}) = \mathrm{Tr}_{\mathbb{F}_{q^m}/\mathbb{F}_q}(\mathsf{GRS}_r(\mathbf{x}, \mathbf{y})).$$

There exists a subclass of alternant codes which is particularly attractive for cryptographic purposes:

Definition 1.91. Let $\mathbf{x} \in \mathbb{F}_{q^m}^n$ be a support vector and $g \in \mathbb{F}_{q^m}[T]$ a univariate polynomial of degree r such that $g(x_i) \neq 0$ for every $i \in \{1, \ldots, n\}$. Then the *Goppa code* of order r with support $\mathbf{x}$ and *Goppa polynomial* g is defined by

$$\Gamma_r(\mathbf{x}, g) = \mathsf{GRS}_r(\mathbf{x}, \mathbf{y})^\perp|_{\mathbb{F}_q},$$

where for any $i \in \{1, \cdots, n\}, y_i := \frac{1}{g(x_i)}$.

The class of binary Goppa codes was the one considered by McEliece in his first proposal [McE78]. They are interesting because when the Goppa polynomial is square–free, there exists a polynomial time algorithm that can correct up to r errors, where r is the order of the Goppa code (a discussion about decoding AG codes can be found in the next section). In Chapter 4, we will consider a generalization of these codes, called Goppa–like AG–codes, and study their behaviour under squaring their dual.

1.3.5 Decoding AG codes

In order to use AG codes in a cryptographic context, we need to know efficient decoding algorithms for this family of codes. Several of them are known, we refer the reader to the survey of Couvreur and Randriambololona [CR20] for more details. An algorithm for general AG codes, called the *basic algorithm* [SV90, JLJ+89], can correct up to half the designed distance minus some defect proportional to the genus of the curve, *i.e.* up to $\lfloor \frac{d^*-1-g}{2} \rfloor$. For AG codes on planes curves, the *modified algorithm* [SV90] can correct at most $\lfloor \frac{d^*-1}{2} - \frac{\mathfrak{g}}{4} \rfloor$ errors. Both these algorithms run in $\mathcal{O}(n^3)$ operations in the base field. Since then, a lot of work have been spend trying to remove the term due to the genus, focusing on specific codes:

In the case of one–point codes, Feng and Rao [FR93], and later Sakata and al. [SJM+95], proposed an algorithm which corrects $\lfloor \frac{d^*-1}{2} \rfloor$ in complexity $\mathcal{O}(n^{\frac{7}{3}})$.

For the case of codes on maximal curves, Pellikaan [Pel89] gives an algorithm which correct up to half the designed distance in $\mathcal{O}(n^4)$ operations over the base field. This particular case is interesting in this thesis as we are sometimes dealing with the Hermitian curve, which is known to be maximal.

All the improvements of the *basic algorithm* since the work of Feng and Rao get rid of the algebraic geometry point of view, introducing the notion of *error correcting pairs*, and allow to correct up to half the designed distance. For recent improvements in this area, see [CP20].

Independently, Sudan [Sud97] showed that, at the cost of possibly returning a list of codewords instead of a single one, it was possible to correct errors on Reed–Solomon codes beyond half the designed distance. A version of the so–called *list decoding algorithm*, valid for general AG codes, is proposed in [GS99]. Recently, a efficient version has been proposed by Beelen, Rosenkilde, and Solomatov [BRS22].

Chapter 2

Code–based cryptography

2.1 The McEliece encryption scheme and its security

2.1.1 Description of the scheme

The McEliece encryption scheme [McE78] is the first encryption scheme based on error–correcting codes. It is a public key cryptosystem whose pair of keys are constructing from a certain family $\mathcal{F}$ of structured codes over a finite field $\mathbb{F}$, for which an efficient decoding algorithm is known. The public key is then a random looking generator matrix of some code in $\mathcal{F}$, while the secret key is the corresponding decoding algorithm. For short, the McEliece scheme can be presented as follows.

The McEliece encryption scheme

Key generation:

Input: The parameters $n, k, t \in \mathbb{N}$ and a finite field $\mathbb{F}$.

1. Select a family $\mathcal{F}$ of linear codes over $\mathbb{F}$ with an efficient decoding algorithm $\mathcal{D}$.
2. Choose a $[n, k]$ code $\mathcal{C} \in \mathcal{F}$ correcting t errors. Let $\mathbf{M}$ be a $k \times n$ generator matrix of $\mathcal{C}$ and denote by $\mathcal{D}_{\mathcal{C}}$ an efficient decoding algorithm for $\mathcal{C}$.

Output: The public key $\mathbf{pk} = (\mathbf{M}, t)$ and the secret key $\mathbf{sk} = \mathcal{D}_{\mathcal{C}}$.

Encryption:

Input: A message $\mathbf{m} \in \mathbb{F}^k$.

1. Select a random error vector $\mathbf{e} \in \mathbb{F}^n$ such that $w_H(\mathbf{e}) \leq t$.
2. Compute $\mathbf{y} := \mathbf{mM} + \mathbf{e}$.

Output: The cypher text $\mathbf{y}$.

Decryption:

Input: A cypher text $\mathbf{y} = \mathbf{c} + \mathbf{e}$, where $\mathbf{c} \in \mathcal{C}$.

1. Compute $\mathbf{c} = \mathcal{D}_{\mathcal{C}}(\mathbf{y})$.
2. Recover the initial message $\mathbf{m}$ from the knowledge of $\mathbf{c} = \mathbf{mM}$ and $\mathbf{M}$ by Gaussian elimination.

Output: The plaintext $\mathbf{m}$.

There exists two kinds of attacks against the McEliece cryptosystem: *message recovery attack* and *key recovery attack*. The first one essentially consists in recovering a plaintext $\mathbf{m}$ from the knowledge of the cypher text $\mathbf{y}$ and the public key. Once done, the attacker knows the plaintext

but not the private key $\mathcal{D}_\mathcal{C}$, meaning that in order to recover several messages, the attack must be performed several times, hence the total cost becomes the sum of the cost of the attack on each cypher text. The second kind of attack, *i.e.* key recovery attacks, consists in retrieving the decoding algorithm $\mathcal{D}_\mathcal{C}$ of $\mathcal{C}$ from the knowledge of one of its generator matrix. In this case, once the attack is performed, the cost of recovering any message becomes the cost of the decoding algorithm, which works in polynomial time in the parameters n and k of the code.

In both cases, given a security parameter λ for the scheme, we say that there exists an efficient attack on the scheme if there exists a polynomial time algorithm (in n and k) which can recover either a plain message from a ciphertext or the secret key in less than 2^λ binary operations. The standard value for the security parameter is $\lambda = 128$, but a more realistic one would be around 80.

The security of the McEliece encryption scheme hence relies on both the choice of the family of codes $\mathcal{F}$ and the hardness of solving the *syndrome decoding problem* (SD). The SD problem does not depend on $\mathcal{F}$ and is related to the security of the message itself (*i.e.* message recovery attack). More precisely, recovering a plaintext $\mathbf{m}$ from the knowledge of its cipher text and a generator matrix of the code reduces to solve an instance of the SD problem. This will be discussed in Section 2.1.2. The choice of $\mathcal{F}$ influences the security of the secret key, which can lead to key recovery attack. In fact, we will see in Section 2.1.3 that for some family of codes which are *too structured*, there exist polynomial time algorithms that recover the secret key from the public one.

2.1.2 Message recovery attack and Information Set Decoding (ISD)

The general idea of security behind McEliece cryptosystem's relies on the hardness of decoding a linear code, a problem which is referred to as the *Worst–Case Syndrome Decoding Problem*, which can be stated as follows in its searching version:

Problem 2. (Search) Worst–Case Syndrome Decoding Problem: *Let $\mathbf{H}$ be a parity check matrix of a $[n, k]$ code $\mathcal{C}$ over a finite field $\mathbb{F}$. Let t be an integer and $\mathbf{s} \in \mathbb{F}^{n-k}$ be a uniformly random vector (called the* syndrome*). Then the (Worst–Case) Search Syndrome Decoding problem is to find a vector $\mathbf{e} \in \mathbb{F}^n$ of Hamming weight $\leq t$ such that $\mathbf{He}^T = 0$.*

This problem differs from the slightly easier *Average–Case Syndrome Decoding Problem*, in which $\mathbf{H}$ is also sampled uniformly at random. The decisional version of the worst–case scenario was proven to be NP–complete in [BMvT78], and is related to the generic decoding problem of a linear code in the following way: let $\mathcal{C}$ be a $[n, k]$ code over $\mathbb{F}$, $\mathbf{H}$ one of its parity check matrix and t its correction capability. Consider a noisy vector $\mathbf{y} = \mathbf{c} + \mathbf{e}$, with $\mathbf{c} \in \mathcal{C}$ and $\mathbf{e} \in \mathbb{F}^n$ an error vector of Hamming weight $\leq t$. Decoding the cipher text $\mathbf{y}$ consists in finding $\mathbf{c}$ without a priori knowledge of any decoding algorithm, and can be realized as follows. From the knowledge of $\mathbf{H}$ and the cipher text, we compute the syndrome $\mathbf{s} := \mathbf{Hy}^T$. Note that

$$\mathbf{s} := \mathbf{Hy}^T = \mathbf{Hc}^T + \mathbf{He}^T = \mathbf{He}^T,$$

meaning that the syndrome does only depend on the error vector. If we are able to solve Problem 2 with parameters $(\mathbf{H}, \mathbf{s}, t)$, then we can decipher $\mathbf{c} = \mathbf{y} - \mathbf{e}$.

To secure the scheme, we need to chose the initial parameters such that the cost of solving Problem 2 is high enough. Apart from the brute force search among all possible error vectors of weight t, all known algorithms for solving the SD problem are generalization of the so–called *Information Set Decoding* (ISD), introduced by Prange in [Pra62]. The idea is to find a set of positions of the noisy vector which does not contain any errors, and such that the corresponding submatrix of the public generator matrix is invertible. Since its introduction in 1962, a lot of improvements of the ISD as been realized. We do not enter into further details here, we refer the reader to the survey of Peters [Pet10] or more recently to the one of [WGR22].

2.1.3 Key recovery attack

The complexity of a key recovery attack highly relies on the choice of the family of codes, and thus is hard to estimate in a general case. It consists in recovering the secret key (*i.e.* a decoding algorithm for the chosen code) knowing only a random looking generator matrix. Usually, this problem can be reduced to the following: given a generator matrix of a $[n, k]$ code $\mathcal{C}$, can we find the inherent structure that defines it, that is to say the secret elements that allow to build an efficient decoding algorithm. For example, this problem can be easy to solve if the family of code is not large enough, in which case a brute force search among all possible codes can be realized in less binary operations than imposed by the security level.

Historically, McEliece [McE78] proposed to use binary classical Goppa codes. For his choices of cryptographic parameters, we do not know any efficient algorithm that can recover the corresponding support and Goppa polynomial; reason why this family of codes is still considered as secure at the moment. Note that the distinguisher given in [MT21] only works for high rate codes, which is out of range of McEliece initial parameters. A recent set of parameters has been proposed to the NIST's post–quantum cryptography standardization project [BCC$^+$22].

The major drawback is that these parameters impose to have huge key sizes, which makes it impracticable. Since 1978, several other families have been investigated to mitigate this problem, while keeping a good security level. In particular, a few proposals use structured codes (*e.g.* quasi–cyclic or quasi–dyadic codes). However, for some families, the key recovery problem is not hard. In the case of Generalized Reed-Solomon (GRS) codes, Sidelnikov and Shestakov proposed in [SS92] a polynomial time algorithm that recovers the secret elements of the codes, by seeking for minimum weight codewords. As a natural geometric generalization, AG codes have been proposed [JM96]: for codes defined over genus ≤ 2 curves, an attack was found by Faure and Minder [FM08]. This attack cannot be extended to any AG code as its complexity is exponential in the genus. Later, Couvreur, Marquez–Corbella and Pellikaan [CMCP17] broke any McEliece scheme based on AG code for arbitrary genus. Their attack (called attack by filtration) does not recover the secret structure of the code, but allows to build directly an efficient decoding algorithm.

In this thesis, we are mainly interested in SSAG codes, as there exists no key recovery attack on them yet. In this case, the key security reduces to find the curve, support and divisor from which the code is defined. More precisely, in SSAG code–based McEliece cryptosystem, the idea is to check if the knowledge of a generator matrix of $\mathsf{SSAG}_q(\mathcal{X}, \mathcal{P}, G)$ allows to find the triple $(\mathcal{X}, \mathcal{P}, G)$. We will focus on two kinds of cryptosystems:

1. Quasi–cyclic SSAG–based McEliece cryptosystems (see Chapter 3), in which case we show that the key security reduces to the security of a subcode of the public code: the *invariant code*.

2. Systems based on one–point Goppa–like AG codes defined over a $C_{a,b}$ curve, which are defined in Chapter 4. In this case, we show that using square codes considerations allows to build a distinguisher on the public code. More precisely, given a random looking matrix, we are able to determine if it comes from a one–point Goppa–like code or not, which means that the secret structure is not hidden enough. This work generalizes the distinguisher proposed in [MT21] in the case of alternant and classical Goppa codes. However, we point out that it seems difficult to turn the distinguisher into an efficient structural attack.

2.2 IOP of Proximity to a linear code

In this section, we give some context in the domain of proximity test to an AG code, by recalling some definition and known properties about proofs systems. This will be useful in Chapter 5, where we define Interactive Oracle Proofs of Proximity to AG codes (AG–IOPP system). The case of proximity tests to Reed–Solomon codes, known as the FRI protocol, is detailed in Section 2.2.3.

2.2.1 Interactive oracle proof (IOPs)

We start by defining a specific proof system that has quite recently emerged in cryptography: Interactive Oracle Proofs (IOPs in short) (see [BCS16] for further explanations). This model has demonstrated to be particularly promising for the design of proof systems in the past few years.

IOPs are proof systems that naturally combines interactive proofs (IPs) and probabilistically checkable proofs (PCPs), and generalize interactive PCP protocols (which consist of a PCP followed by an IP). Hence, we start by recalling these alternative models of proof systems.

Interactive proofs (IPs). Interactive proofs were introduced by Goldwasser, Micali and Rackoff [GMR89]: it consists in a r–round interactive proof between a probabilistic polynomial–time verifier and an all-powerful prover. During this interaction, r messages are exchanged, and the protocol ends with the response of the verifier, which either accepts or rejects the proof proposed by the prover. An IP is said to be public–coin if all the verifier's messages are chosen uniformly and independently at random (such a system is also called an Arthur–Merlin game [Bab85]). Since 1992, it is well–known that any IOP can be turned into an Arthur–Merlin game via the Fiat–Shamir transform [Sha92].

Probabilistically checkable proofs (PCPs). Probabilistically checkable proofs were introduced by [ALM+98, BFLS91, AS92]: roughly speaking, in a PCP system, a probabilistic polynomial–time verifier has oracle access to a proof string of length at most 2^u that uses at most u bits of randomness, and queries at most q locations of the proof before accepting or rejecting it.

Interactive Oracle Proofs (IOPs). An Interactive Oracle Proof (IOP) is a "multi–round PCP" in which the verifier has oracle access to the prover's messages, and may probabilistically query them (rather than having to read them in full). In more details, during each of the r rounds, the verifier sends a challenge c to the prover, which he reads in full, before replying with a message f. The verifier can then query to f as an oracle string. After the r rounds, the verifier either accepts or rejects the proof.

The efficiency in the IOP model is expressed in terms of the *proof length* (total number of bits in all the prover's messages), the *query complexity* (total number of locations queried by the verifier across all prover's messages) and the *round complexity* r (total number of rounds).

2.2.2 Proximity testing to an evaluation code

To construct a proof system for a non–deterministic binary relation $\mathcal{R}$, arithmetization techniques (introduced in [LFKN90]) transform any instance–witness (x, w) into a word that belongs to a certain error–correcting code $\mathcal{C}$ if $(x, w) \in \mathcal{R}$, and is very far from $\mathcal{C}$ otherwise. This motivates a new proof system, namely public–coin IOP of proximity (IOPP) to a linear code $\mathcal{C}$.

To specify our definition, we consider a finite field $\mathbb{F}$ and some evaluation code $\mathcal{C} \subseteq \mathbb{F}^S$ with evaluation domain S of size n over the alphabet $\mathbb{F}$. An IOPP (P,V) for the code $\mathcal{C}$ is a pair of randomized algorithms, where both P (the prover) and V (the verifier) receive as explicit input the specification of the code $\mathcal{C} \subseteq \mathbb{F}^S$. We define the input size to be the integer $n = |S|$. Furthermore, a purported codeword $f : S \to \mathbb{F}$ is given as explicit input to P and as an oracle to V. The two parties interact over at most $r = r(n)$ rounds and, during this conversation, P seeks to convince V that f belongs to the code $\mathcal{C}$. More precisely, at each round, V sends a message chosen uniformly and independently at random, and P answers with an oracle. Verifier's queries to the prover's message are generated by public randomness and performed after the end of the interaction with the prover. Thus, this proof system is a public–coin protocol, as defined above.

We define the *Hamming relative distance* between f and the code $\mathcal{C}$ by

$$\Delta(f, \mathcal{C}) := \min \{d_H(f, \mathbf{c}) \mid \mathbf{c} \in \mathcal{C}\}.$$

The output of V after interacting with P is denoted by $\langle \mathrm{P} \leftrightarrow \mathrm{V} \rangle \in \{\text{accept}, \text{reject}\}$. Finally, the notation V^f means that f is given as an oracle input to V.

Definition 2.1. A pair of randomized algorithms (P,V) is an IOPP system for the code $\mathcal{C} \subseteq \mathbb{F}^n$ with soundness error $s : (0, 1] \to [0, 1]$ if the following conditions hold:

1) **Perfect completeness:** If $f \in \mathcal{C}$, then

$$\Pr\left[\left\langle \mathrm{P}(\mathcal{C}, f) \leftrightarrow \mathrm{V}^f(\mathcal{C}) \right\rangle = \text{accept}\right] = 1,$$

i.e. if f is in fact a codeword of $\mathcal{C}$, then the verifier accepts the proof with probability one.

2) **Soundness:** For any function $f : S \to \mathbb{F}$ such that $\delta := \Delta(f, \mathcal{C}) > 0$ and any malicious prover P^*, we have

$$\Pr\left[\left\langle \mathrm{P}^* \leftrightarrow \mathrm{V}^f(\mathcal{C}) \right\rangle = \text{accept}\right] \leq s(\delta),$$

i.e. if $f \notin \mathcal{C}$, a cheating prover can only convince the verifier with low probability.

The efficiency of an IOPP system is expressed in terms of several parameters: the sum of lengths of prover's messages defines the proof length $\ell(n)$, expressed in number of symbols in the alphabet $\mathbb{F}$. The query complexity $q(n)$ is the total number of queries made by the verifier to both the purported codeword f and the oracle sent by the prover during the interaction. The prover complexity $t_p(n)$ is the time needed to generate prover's messages (which does not include the input f) and the verifier complexity $t_v(n)$ is the time spent by the verifier to make his decision when queries and query-answers are given as inputs.

2.2.3 The FRI protocol

The IOPP construction described in the previous section has been applied in the case where $\mathcal{C}$ is a RS code, and is referred to as the Fast Reed–Solomon IOPP (FRI in short, see [BKS18] for a complete presentation). As we aim to consider an IOPP for general AG codes, we explain how the FRI protocol works below.

Let k be a positive integer and $\rho \in (0,1)$ such that $\rho := 2^{-k}$. The FRI protocol allows to check proximity to a Reed–Solomon code of length $n = |\mathcal{P}|$, say

$$\mathsf{RS}(\mathcal{P}, \rho) := \{f : \mathcal{P} \to \mathbb{F} \mid \deg(f) < \rho n\},$$

by testing proximity to a smaller code $\mathsf{RS}(\mathcal{P}', \rho)$ with $|\mathcal{P}'| < n = |\mathcal{P}|$. To do so, the protocol considers a family of linear maps $\mathbb{F}^{\mathcal{P}} \to \mathbb{F}^{\mathcal{P}'}$ which randomly *fold* any function in $\mathbb{F}^{\mathcal{P}}$ into a function in $\mathbb{F}^{\mathcal{P}'}$. We present below in a simplified way three ingredients that enable the protocol to work.

1. Splitting of polynomials into even and odd part. Given an univariate polynomial $f \in \mathbb{F}[x]$ such that $\deg(f) < \rho n$, there exists two polynomials g, h of degree both $< \frac{1}{2}\rho n$ such that

$$f(x) = g(x^2) + xh(x^2). \tag{2.1}$$

   One may view such a decomposition as the result of the splitting of the space of polynomials of degree less than ρn into two copies of the space of polynomials of degree less than $\rho n/2$.

2. Randomized folding. Choose $\mathcal{P}$ to be a multiplicative group of order 2^r generated by $\omega \in \mathbb{F}$. Then, define $\mathcal{P}' = \langle \omega^2 \rangle = \{x^2 \mid x \in \mathcal{P}\}$. Set $\pi : \mathbb{F} \to \mathbb{F}$ to be the map defined by $\pi(x) = x^2$ and observe that $\pi(\mathcal{P}) = \mathcal{P}'$. Moreover, $|\mathcal{P}'| = |\mathcal{P}|/2$. The structure of the evaluation domain allows to reduce the proximity problem by half the size at each round of iteration. Based on the decomposition (2.1), we define the folding operator $\mathbf{Fold}[\cdot, z] : \mathbb{F}^{\mathcal{P}} \to \mathbb{F}^{\mathcal{P}'}$ for any $z \in \mathbb{F}$ as follows:

$$\mathbf{Fold}[f, z] := g(x) + zh(x).$$

   If $\deg(f) < \rho n$, both functions $g : \mathcal{P}' \to \mathbb{F}$ and $h : \mathcal{P}' \to \mathbb{F}$ belong to the smaller code $\mathsf{RS}(\mathcal{P}', \rho)$. Then for any random challenge $z \in \mathbb{F}$, the operator $\mathbf{Fold}[\cdot, z]$ maps $\mathsf{RS}(\mathcal{P}, \rho)$ into $\mathsf{RS}(\mathcal{P}', \rho)$.

3. Distance preservation after folding. Except with small probability over z, if $\Delta(f, \mathsf{RS}(\mathcal{P}, \rho)) \geq \delta$, then

$$\Delta\left(\mathbf{Fold}[f, z], \mathsf{RS}(\mathcal{P}', \rho)\right) \geq (1 - o(1))\delta.$$

The FRI protocol goes as follow: the verifier sends a random challenge $z \in \mathbb{F}$ and the prover answers with an oracle function $f' : \mathcal{P}' \to \mathbb{F}$, which is expected to be equal to the folded function $\mathbf{Fold}[f, z] : \mathcal{P}' \to \mathbb{F}$. At the next round, f' becomes the function to be *folded*, and the process is repeated for r rounds. Each round reduces the problem by half (*i.e.* the size of the evaluation domain), eventually leading to a function $f^{(r)}$ evaluated over a small enough set of points. This induces a sequence of Reed–Solomon codes of strictly decreasing length, while the code rate remains unchanged (and so does the relative minimum distance). The final test consists in testing if $f^{(r)}$ belongs to the last RS code.

Perfect completeness of the protocol follows from 2. Prover and verifier efficiencies come from the possibility of determining any value of $\mathbf{Fold}[f, z]$ at any point $y \in \mathcal{P}'$ with exactly two values of f, namely on the set $\pi^{-1}(\{y\})$ (*i.e.* the square roots of y). Hence, a consistency test between f and f' only requires two queries to f and one to f'.

Soundness of the protocol mainly relies on item 3. It is proved using results about distance preservation under random linear combinations. Based on that, we can deduce that if the folded function $\mathbf{Fold}[f, z]$ is close to $\mathcal{C}$ for enough values of z, then it remains true for both g and h (defined in Equation (2.1)). Details about this can be found in [BBHR18, BKS18, BGKS20, BCI+20].

Remark 2.2. Note that 3; above holds because both polynomials g and h appearing in the decomposition (2.1) have exactly the same degree, arising from the crucial fact that the FRI protocol only considers RS codes of dimension a power of two, meaning that the considered polynomials have degree at most an odd bound.

Let us sketch what could happens when f is expected to have degree at most an even integer, say $2d$. According to Equation (2.1), we have $\deg(g) \leq d$ and $\deg(h) \leq d-1$. Therefore, if $\deg(f) \leq 2d$, then $g + zh$ corresponds to a polynomial of degree $\leq d$. However, knowing that $g + zh$ has degree $\leq d$ with high probability on $z \in \mathbb{F}$ only tells us that both g and h have degree $\leq d$, which is not enough to deduce that $\deg(f) \leq 2d$ and not $2d+1$. Even worse, it is worth noting that words corresponding

to a degree $2d+1$ polynomial are among the farthest words from the RS code of degree $\leq 2d$. In the univariate case, the obstacle can be overcome by supposing not only $\deg(g), \deg(h) \leq d$ but also $\deg(\nu h) \leq d$ for a degree 1 polynomial function ν, called *balancing function*. This implies $\deg(h) < d$ and hence $\deg(f) \leq 2d$. This will be really important in order to generalize this protocol to AG codes since we will have to deal with the same issue.

Chapter 3

Structural attack against quasi–cyclic SSAG codes

After several attempts in trying to replace the family of binary Goppa codes by AG codes, the natural generalization of Reed–Solomon codes, in the McEliece encryption scheme, an attack for AG codes defined on any genus curve is now known [CMCP17]. However, the scheme based on SSAG codes is still unbroken, hence considering structured (*e.g.* quasi-cyclic) SSAG codes in order to reduce key sizes is still promising. In the present chapter, we focus on this family of codes and make some progress in the direction of structural attacks. More precisely, we show that the security of the public quasi-cyclic SSAG code reduces to the security of its invariant code (see Definition 1.15). Since the invariant code can be derived from the public data and has smaller parameters, cautions should be made while considering such cryptosystems. This work can be seen as an extension of Chapter 5 of Barelli's PhD memoir [Bar18a], as her method is generalized in order to be applied in more general setting.

For the whole chapter, we consider a Galois cover $\pi : \mathcal{Y} \to \mathcal{X}$ of smooth and irreducible projective curves over the finite field $\mathbb{F}_{q^m}$. This corresponds (via Theorem 1.35) to a Galois extension of function fields L/K, where $L = \mathbb{F}_{q^m}(\mathcal{Y})$ and $K = \mathbb{F}_{q^m}(\mathcal{X})$. Note that our AG and SSAG codes will first be defined over the curve $\mathcal{Y}$, while the corresponding invariant subcodes lie on the quotient curve $\mathcal{X}$.

The chapter is organized as follows. In Section 3.1, we define structured AG codes and give some properties of their invariant code. Next, we present the general idea to recover the equation of a Galois cover of curves in Section 3.2. Section 3.3 is dedicated to concrete instances of our attack: after giving properties required for the quotient curve, we focus on Kummer coverings and elementary abelian p-extensions. We finish the Chapter by discussing how our attack could be generalized to a solvable Galois cover in Section 3.4.

3.1 Preliminaries

3.1.1 Structured AG codes

By *structured codes*, we mean codes with a non trivial permutation group (Definition 1.14), coming from the underlying geometry. We now explain how to obtain such codes, starting from the geometry of the curve: to fix notation, consider an AG code

$$\mathcal{C} := C_{\mathcal{L}}(\mathcal{Y}, \mathcal{Q}, G)$$

defined on the curve $\mathcal{Y}$, where $\mathcal{Q}$ is a set of n distinct rational points which does not intersect the support of some divisor $G \in \mathrm{Div}(\mathcal{Y})$ and assume as usual that $\deg(G) < n$. Let $\mathrm{Aut}(\mathcal{Y}/\mathbb{F}_{q^m})$ be the automorphism group of $\mathcal{Y}$ over $\mathbb{F}_{q^m}$, acting on the left. Thanks to Theorem 1.35, we can identify $\mathrm{Aut}(L/\mathbb{F}_{q^m})$ with $\mathrm{Aut}(\mathcal{Y}/\mathbb{F}_{q^m})$, acting on L on the right: for $\sigma \in \mathrm{Aut}(L/\mathbb{F}_{q^m}) \simeq \mathrm{Aut}(\mathcal{Y}/\mathbb{F}_{q^m})$ and $f \in L$, we set $f^\sigma := \sigma^* f = f \circ \sigma$. Now consider a finite subgroup $\Sigma \subseteq \mathrm{Aut}(\mathcal{Y}/\mathbb{F}_{q^m})$, and assume that $\mathcal{Q}$ and G are invariant under Σ; observe that this means precisely that $\mathcal{Q}$ is a union, and G a sum, of orbits under Σ. Since G is invariant, Σ also acts on the Riemann–Roch space $\mathcal{L}(G)$. As $n > \deg(G)$, the isomorphism

$$\mathrm{ev}_{\mathcal{Q}} : \mathcal{L}(G) \to \mathcal{C}$$

sends this action to the AG code. Each $\sigma \in \Sigma$ acts by

$$\begin{aligned}\mathbf{c} = (f(Q_1), \ldots, f(Q_n)) \mapsto \mathbf{c}^\sigma := \mathrm{ev}_{\mathcal{Q}}(f^\sigma) &= (\sigma^* f(Q_1), \ldots, \sigma^* f(Q_n)) \\ &= (f(\sigma^{-1}(Q_1)), \ldots, f(\sigma^{-1}(Q_n)))\end{aligned}$$

As $\mathcal{Q}$ is also Σ–invariant, the set $\left\{\sigma^{-1}(Q_1), \ldots, \sigma^{-1}(Q_n)\right\}$ is a permutation of $\mathcal{Q}$, and the above action on $\mathcal{C}$ permutes the corresponding coordinates. We denote this permutation by $\widetilde{\sigma}$, meaning that for every $1 \leq i \leq n$, we have $Q_{\widetilde{\sigma}(i)} = \sigma^{-1}(Q_i)$. As σ ranges in Σ, these $\widetilde{\sigma}$ form a subgroup $\widetilde{\Sigma}$ of the permutation group $\mathrm{Perm}(\mathcal{C})$, acting on the right.

It is clear that any permutation automorphism of a linear code stabilizes its subfield subcodes. Hence, given a subgroup $\Sigma \subseteq \mathrm{Aut}(\mathcal{Y}/\mathbb{F}_{q^m})$, and assuming $\mathcal{Q}$ and G to be invariant under Σ as above, we also get a subgroup $\widetilde{\Sigma}_{\mathrm{SSAG}}$ of $\mathrm{Perm}(\mathsf{SSAG}_q(\mathcal{Y}, \mathcal{Q}, G))$, acting on the right. Note that $\widetilde{\Sigma}_{\mathrm{SSAG}}$ is nothing but $\widetilde{\Sigma}$ restricted to the subfield subcode. Later on we will occasionally abuse notation and write σ for $\widetilde{\sigma}$, and Σ for $\widetilde{\Sigma}$ or $\widetilde{\Sigma}_{\mathrm{SSAG}}$, whenever it is clear from the context.

Since we know how to construct AG and SSAG codes that are invariant under some automorphism subgroup, we focus on the properties of the corresponding invariant code.

3.1.2 The invariant code

Fix an AG code

$$\mathcal{C} := C_{\mathcal{L}}(\mathcal{Y}, \mathcal{Q}, G)$$

as in the previous section, and let $\Sigma \subseteq \mathrm{Aut}(\mathcal{Y}/\mathbb{F}_{q^m})$ be an automorphism subgroup. In the discussion below, we show that if $\mathcal{C}$ is Σ–invariant, its (punctured) invariant subcode $\mathcal{C}^\Sigma$ (see. Definition 1.15) is also an AG code, defined on the quotient curve $\mathcal{Y}/\Sigma$, whose function field is the fixed field $K := L^\Sigma$. We start with the following lemma:

Lemma 3.1. *With above notation, assume that $\mathcal{Q}$ and G are invariant under an automorphism $\sigma \in \mathrm{Aut}(\mathcal{Y}/\mathbb{F}_{q^m})$. If a codeword $\mathbf{c} = \mathrm{ev}_{\mathcal{Q}}(f) \in C_{\mathcal{L}}(\mathcal{Y}, \mathcal{Q}, G)$ is σ–invariant, i.e if $\mathbf{c}^\sigma = \mathbf{c}$, then $f^\sigma = f$.*

Proof. Write $\mathcal{Q} = \{Q_1, ..., Q_n\}$. We have

$$0 = \mathbf{c}^\sigma - \mathbf{c} = \mathrm{ev}_{\mathcal{Q}}(f^\sigma - f),$$

hence $f^\sigma - f$ admits at least n zeroes $Q_1, \ldots, Q_n$. Since G is σ–invariant, $f^\sigma \in \mathcal{L}(G)$, thus $f^\sigma - f \in \mathcal{L}(G)$. As $n > \deg(G)$, this imposes $f^\sigma - f = 0$. □

Our next definition will play a crucial while considering invariant codes.

Definition 3.2. Consider an extension L/K of function fields, corresponding to a morphism of curves $\pi : \mathcal{Y} \to \mathcal{X}$. Given a divisor $G \in \mathrm{Div}(L)$, we define its *pushforward* $\widetilde{G}$ as the largest divisor in K such that

$$\pi^* \widetilde{G} \leq G.$$

Remark 3.3. We warn the reader that our definition of *pushforward* differs from the one usually used in the literature, and denoted by $\pi_* G$. We will encounter the latter terminology later, in Chapter 5. Since our definition notion will be widely used thought this chapter, we made the choice to give it a name, which fits well as it is in some sense a "dual" operation of *pullback*.

The properties of pushforwards are put together in the upcoming result.

Lemma 3.4. *With notation as above:*

(i) $\mathcal{L}_L(G) \cap K = \mathcal{L}_K(\widetilde{G})$.

(ii) if we write

$$G = \sum_{S \in \mathbb{P}_L} t_S S$$

for integers t_S almost all of which are 0, then

$$\widetilde{G} = \sum_{R \in \mathbb{P}_K} \left(\min_{S|R} \left\lfloor \frac{t_S}{e(S|R)} \right\rfloor \right) R.$$

(iii) we have

$$\mathrm{Supp}(\widetilde{G}) \subseteq \pi(\mathrm{Supp}(G)).$$

(iv) for $G_1, G_2 \in \mathrm{Div}(L)$, we have

$$\widetilde{(G_1 + G_2)} \geq \widetilde{G_1} + \widetilde{G_2}$$

(v) if $A \in \mathrm{Div}(K)$, then

$$\widetilde{\pi^* A} = A.$$

Proof. *(i)* Given $h \in K$ we have $(h)^L = \pi^*(h)^K$, and thus we have $h \in \mathcal{L}_L(G)$ iff $\pi^*(h)^K \geq -G$. But by Definition 3.2 this means precisely $(h)^K \geq -\widetilde{G}$, *i.e.* $h \in \mathcal{L}_K(\widetilde{G})$.

(ii) If $\widetilde{G} = \sum_{R \in \mathbb{P}_K} x_R R$, then

$$\pi^* \widetilde{G} = \sum_{R \in \mathbb{P}_K} \sum_{S|R} x_R e(S|R) S,$$

and so $\pi^* \widetilde{G} \leq G$ if and only if

$$x_R \leq \frac{t_S}{e(S|R)}$$

for all $R \in \mathbb{P}_K$ and all $S|R$. Then $\widetilde{G}$ is maximal when all x_R are maximal under this condition, which means precisely

$$x_R = \min_{S|R} \left\lfloor \frac{t_S}{e(S|R)} \right\rfloor.$$

The remaining results are consequences of *(ii)*. □

When L/K is a Galois extension, the above lemma can be precised when talking about invariant divisor:

Proposition 3.5. *Let L/K be a Galois extension of function fields, with Galois group Σ, and let $G \in \mathrm{Div}(L)$ be a Σ–invariant divisor. Then:*

(i) $\mathcal{L}_L(G)^\Sigma = \mathcal{L}_K(\widetilde{G})$

(ii) if we write G as a sum of orbits under Σ, i.e.

$$G = \sum_{i=1}^{s} t_i \sum_{Q \in \mathrm{Orb}_\Sigma(S_i)} Q$$

for places $S_1, ..., S_s \in \mathbb{P}_L$ nonconjugate under Σ and integers $t_i \in \mathbb{Z} \backslash \{0\}$, then

$$\widetilde{G} = \sum_{i=1}^{s} \left\lfloor \frac{t_i}{e(S_i|R_i)} \right\rfloor R_i,$$

where $R_i = S_i \cap K \in \mathbb{P}_K$.

Proof. *(i)* Since $\mathcal{L}_L(G)^\Sigma = \mathcal{L}_L(G) \cap L^\Sigma = \mathcal{L}_L(G) \cap K$, this is a special case of (i) in the previous Lemma.

(ii) Observing that G is Σ–invariant and that $e(Q|R_i) = e(S_i|R_i)$ for all $Q \in \mathrm{Orb}_\Sigma(S_i)$, this is again a special case of (ii) in the previous Lemma.

□

Theorem 3.6. *Let $\mathcal{C} := C_\mathcal{L}(\mathcal{Y}, \mathcal{Q}, G)$ be an AG code defined on a curve $\mathcal{Y}$ over $\mathbb{F}_{q^m}$, with $\mathcal{Q}$ and G invariant under the action of $\Sigma \subseteq \mathrm{Aut}(\mathcal{Y}/\mathbb{F}_{q^m})$. Then its invariant code under Σ is also an AG code, defined on the quotient curve $\mathcal{Y}/\Sigma$. In particular, we have*

$$\mathcal{C}^\Sigma = C_\mathcal{L}(\mathcal{Y}/\Sigma, \mathcal{P}, \widetilde{G}),$$

where $\widetilde{G}$ is given by Proposition 3.5, (ii) and

$$\mathcal{P} = \{Q \cap L^\Sigma \ , Q \in \mathcal{Q}\}.$$

Proof. Consequence of Lemma 3.1 and Proposition 3.5. □

Remark 3.7. The invariant and subfield subcode operations commute: more precisely, for any linear code $\mathcal{C} \subseteq \mathbb{F}_{q^m}^n$ and Σ a subgroup of its permutation group, we have

$$(\mathcal{C} \cap \mathbb{F}_q^n)^\Sigma = \{c \in \mathcal{C} \mid c \in \mathbb{F}_q^n \text{ and } \sigma(c) = c \ , \ \forall \sigma \in \Sigma\} = \mathcal{C}^\Sigma \cap \mathbb{F}_q^n.$$

Corollary 3.8. *With notation of Theorem 3.6, we have*

$$\mathsf{SSAG}_q\left(\mathcal{Y}, \mathcal{Q}, G\right)^\Sigma = \mathsf{SSAG}_q\left(\mathcal{Y}/\Sigma, \mathcal{P}, \widetilde{G}\right).$$

Proof. Consequence of Theorem 3.6 and Remark 3.7. □

3.2 Finding the equation of a Galois cover

While considering a structured SSAG code $\mathsf{SSAG}_q\left(\mathcal{Y}, \mathcal{Q}, G\right)$ as public key for the McEliece cryptosystem, showing that the secret structure of the public code can be recovered from the invariant code consists in assuming that the geometric structure of the invariant code is known, then use it to find the secret data. More precisely, the crucial step is to recover an equation for the curve $\mathcal{Y}$. In this section, we provide the general idea to do so.

3.2.1 Setting

Let us consider a Galois cover

$$\pi : \mathcal{Y} \longrightarrow \mathcal{X}$$

between curves over $\mathbb{F}_{q^m}$, with Galois group Σ. It corresponds to a Galois extension of function fields L/K, where $K = \mathbb{F}_{q^m}(\mathcal{X})$, $L = \mathbb{F}_{q^m}(\mathcal{Y})$ and such that $\Sigma = \mathrm{Gal}(L/K)$ (*i.e.* $\mathcal{X} = \mathcal{Y}/\Sigma$ is the corresponding quotient curve). In particular, there exists a primitive element $y \in L$ such that $L = K(y)$ and

$$H(y) = 0 \ , \text{ where } H \in K[T] \text{ irreducible polynomial.}$$

By an equation of $\mathcal{Y}$ over $\mathcal{X}$, we mean the minimal polynomial H of such a y over K. Denote by $\ell = [L : K] = |\Sigma|$ the degree of the extension, and suppose we are given a set of r rational points on $\mathcal{X}$, say $\mathcal{P} = \{P_1, ..., P_r\}$, that totally split in the cover $\mathcal{Y} \to \mathcal{X}$. For any $1 \leq i \leq r$, we then have

$$\pi^* P_i = Q_{i,1} + ... + Q_{i,\ell},$$

where $Q_{i,j} | P_i$. In particular, it means that for all $1 \leq i \leq r$,

$$\mathrm{Orb}_\Sigma(Q_{i,1}) = \{Q_{i,1}, ..., Q_{i,\ell}\}.$$

We denote by $\mathcal{Q} = \{Q_{i,j} \mid 1 \leq i \leq r \text{ and } 1 \leq j \leq \ell\}$ the set of all extensions of the P_i's in L.

Let $G \in \mathrm{Div}(L)$ be a Σ–invariant divisor such that $\mathrm{Supp}(G) \cap \mathcal{Q} = \varnothing$ and $\deg(G) < n := \ell r$. Denote by $\widetilde{G} \in \mathrm{Div}(K)$ its pushforward (with respect to Definition 3.2), which satisfies $\mathrm{Supp}(\widetilde{G}) \cap \mathcal{P} = \varnothing$. The AG code $C_\mathcal{L}(\mathcal{Y}, \mathcal{Q}, G)$ then admits $C_\mathcal{L}(\mathcal{X}, \mathcal{P}, \widetilde{G})$ as its invariant subcode under the action of Σ, and likewise the SSAG–code

$$\mathcal{C} := \mathsf{SSAG}_q\left(\mathcal{Y}, \mathcal{Q}, G\right)$$

admits $\mathcal{C}^\Sigma = \mathsf{SSAG}_q\left(\mathcal{X}, \mathcal{P}, \widetilde{G}\right)$ as its invariant subcode. Hence, considering $\mathcal{C}$ as public code in a McEliece encryption scheme, we aim to recover its geometric structure from the knowledge of $\mathcal{C}^\Sigma$. To do so, we make the following assumptions:

A1. We know a generator matrix $\mathbf{M} \in \mathcal{M}_{k,n}(\mathbb{F}_{q^m})$ of the SSAG code $\mathcal{C}$, where $k := \dim_{\mathbb{F}_q} \mathcal{C}$.

A2. We know the permutation subgroup $\widetilde{\Sigma}_{\mathrm{SSAG}} \subseteq \mathrm{Perm}(\mathcal{C})$ induced by Σ.

A3. We know the underlying geometric structure of the invariant subcode $\mathcal{C}^\Sigma$, *i.e.* we know the quotient curve $\mathcal{X}$ or equivalently its function field K, the finite set of places $\mathcal{P}$ in K, and the invariant divisor $\widetilde{G} \in \mathrm{Div}(K)$.

As our aim is to recover an equation of $\mathcal{Y}$ over $\mathcal{X}$, we describe below the general principles of our method that allows to do so, at least under a further assumption that will be given later. Observe first that, as a consequence of A2 and A3, we know the degree ℓ of the cover, and for any place $P_i \in \mathcal{P}$, we know the coordinate positions of $\mathcal{C}$ corresponding to the ℓ extensions $Q_{i,j}$ of P_i in $\mathcal{Q}$. As justified by the following proposition, the main ingredient of our method will be to find the evaluation vector $\mathbf{y} = (y(Q_{i,j}))_{i,j}$ of length $n = \ell r$, for some primitive element $y \in L$ satisfying certain degree conditions.

Proposition 3.9. *Let y be a primitive element of L over K (i.e. $L = K(y)$), and*

$$H(T) = T^\ell + h_{\ell-1}T^{\ell-1} + \cdots + h_1 T + h_0 \in K[T]$$

be its minimal polynomial. For each $k \in \{0, \ldots, \ell-1\}$, assume that either:

- *we know a priori some divisor $D_k \in \mathrm{Div}(K)$, of degree $\deg(D_k) < r$, such that $h_k \in \mathcal{L}_K(D_k)$; or:*
- *the support of $(h_k)_\infty^K$ is unknown, but we know that it has degree $\deg(h_k)_\infty^K < r/2$.*

Then the polynomial $H(T)$ is entirely determined by the evaluation vector $\mathbf{y} = (y(Q_{i,j}))_{i,j}$.

Proof. By Galois theory, the roots of H are $y^{\sigma_1}, \ldots, y^{\sigma_\ell}$, where $\Sigma = \{\sigma_1, \ldots, \sigma_\ell\}$; and so for each k we have

$$h_k = (-1)^{\ell-k} s_{\ell-k}(y^{\sigma_1}, \ldots, y^{\sigma_\ell})$$

where $s_{\ell-k}$ stands for the degree $\ell-k$ elementary symmetric polynomial. Evaluating at $Q_{i,j}$, we deduce

$$h_k(P_i) = (-1)^{\ell-k} s_{\ell-k}(y(Q_{i,1}), \ldots, y(Q_{i,\ell})), \qquad \text{for all } i \in \{1, \ldots, r\}. \tag{3.1}$$

Now it suffices to show that Equation (3.1) entirely determines h_k.

In the case that $h_k \in \mathcal{L}_K(D_k)$ for a known D_k of degree $\deg(D_k) < r$, let $h'_k \in \mathcal{L}_K(D_k)$ be another solution of (3.1). Then $h_k - h'_k \in \mathcal{L}_K(D_k)$, but $h_k - h'_k$ has at least r zeroes, hence $h_k - h'_k = 0$.

If we only know that $\deg(h_k)_\infty^K < r/2$, let h'_k be another solution of (3.1) with $\deg(h'_k)_\infty^K < r/2$, and set $D_k = (h_k)_\infty^K + (h'_k)_\infty^K$, so $\deg(D_k) < r$. Then again we have $h_k - h'_k \in \mathcal{L}_K(D_k)$, but $h_k - h'_k$ has at least r zeroes, hence $h_k - h'_k = 0$. □

Note that finding H from $\mathbf{y}$ reduces to an interpolation problem: find the h_k knowing their values $h_k(P_i)$. Ultimately, this boils down to solving a linear system. Proposition 3.9 asserts that the system will have a unique solution provided the cover $\mathcal{Y}$ over $\mathcal{X}$ can be defined by an equation H whose coefficients h_k are small enough compared to the length n of $\mathcal{C}$. However, the algorithmic complexity of writing down this linear system depends on the geometry: in case $h_k \in \mathcal{L}_K(D_k)$ for a known D_k, we face a generalization to $\mathcal{X}$ of the Lagrange interpolation problem. In case we only have control on the degree of $(h_k)_\infty^K$, we are in front of a generalized rational function reconstruction problem. The study of these interpolation problems on arbitrary curves is still an area of active research. Efficient algorithms are known for curves of genus 0, and ad hoc methods could be devised for specific instances.

3.2.2 Finding the evaluation vector

Now we focus on the task of finding a suitable evaluation vector $\mathbf{y}$. For this we make a fourth assumption:

A4. We know an effective divisor $B \in \mathrm{Div}(K)$ such that the Riemann–Roch space $\mathcal{L}_L(\pi^* B)$ contains a primitive element y for L over K.

Moreover we will request that this $B \geq 0$ is *not too large*, in a sense that will appear clearer later.

Remark 3.10. We observe that the assumption A4 is closely related to the conditions in Proposition 3.9. Indeed, if $y \in \mathcal{L}_L(\pi^* B)$, then also $y^\sigma \in \mathcal{L}_L(\pi^* B)$ for each conjugate y^σ, from which it follows $h_k = (-1)^{\ell-k} s_{\ell-k}(y^{\sigma_1}, \ldots, y^{\sigma_\ell}) \in \mathcal{L}_K((\ell-k)B)$, *i.e.* we can take $D_k = (\ell-k)B$. Conversely, if we know the D_k, then we can find a B using the theory of Newton polygons.
However, it might be that in some instances, we have additional information either on B or on the $(h_k)_\infty^K$. Regarding Proposition 3.9, we also need to be able to compute Riemann–Roch spaces on $\mathcal{X}$, especially those associated to the $D'_k s$, reason why we keep assumption A4 and the conditions in the proposition separate, in order to allow more flexibility.

For instance, suppose we have some a priori information that allows us to know $(y)_\infty^L$ perfectly. It is then easily checked that we can take $B = -\widetilde{(-(y)_\infty^L)}$, and actually this will be the optimal (*i.e.* the smallest) choice. More precisely, while considering applications in Section 3.3, the divisor $(y)_\infty^L$ will be a multiple of the unique point at infinity in the extension, hence all the Riemann-Roch spaces on K we need to compute will be known.

Notation 3.11. For the rest of the chapter, we adopt the following notation: given any set S of rational points on a curve $\mathcal{Y}$ with function field L, we denote by D_S the divisor in $\mathrm{Div}(L)$ which is defined as the sum of all places in S, *i.e.*

$$D_S := \sum_{P \in S} P.$$

Now, we make use of the specific structure of algebraic geometry codes: first, as subfield subcode, we have $\mathsf{SSAG}_q(\mathcal{Y},\mathcal{Q},G) \subseteq C_{\mathcal{L}}(\mathcal{Y},\mathcal{Q},G)$, hence

$$C_{\mathcal{L}}(\mathcal{Y},\mathcal{Q},G)^{\perp} \subseteq \mathsf{SSAG}_q(\mathcal{Y},\mathcal{Q},G)^{\perp} \otimes \mathbb{F}_{q^m}.$$

Then, from Proposition 1.83, there exists a differential $\omega \in \Omega_L$ such that

$$C_{\mathcal{L}}(\mathcal{Y},\mathcal{Q},G)^{\perp} = C_{\mathcal{L}}\left(\mathcal{Y},\mathcal{Q},G^{\perp}\right) \cdot \mathbf{Z}_{\omega}, \tag{3.2}$$

where $G^{\perp} = D_{\mathcal{Q}} - G + (\omega)^L$, $\nu_{Q_{i,j}}(\omega) = -1$ for all $Q_{i,j} \in \mathcal{Q}$ and $\mathbf{Z}_{\omega}$ is the diagonal matrix of length $n = \ell r$ whose coefficient are $(\mathrm{Res}_{\omega}(Q))_{Q\in\mathcal{Q}}$. For all $g \in \mathcal{L}_L(G^{\perp})$, we then have

$$\mathbf{M} \cdot \mathbf{Z}_{\omega} \cdot \mathrm{ev}_{\mathcal{Q}}(g)^T = 0, \tag{3.3}$$

where $\mathbf{M}$ is a generator matrix of $\mathsf{SSAG}_q(\mathcal{Y},\mathcal{Q},G)$, seen as a matrix over $\mathbb{F}_{q^m}$ by scalar extension.

At this step, we could be tempted to use a differential $\omega \in \Omega_L$ such that $\mathbf{Z}_{\omega}$ is the identity matrix (see. Theorem 1.82) as it would clearly simplify Equation (3.3), but we actually have no reason to know this differential explicitly, since L is unknown. More precisely, the trickiest part is to find some explicit differential (*i.e.* in terms of known data) whose valuation at any place in the unknown support $\mathcal{Q}$ equals -1. Among these differentials, if we manage to find one such that the divisor $G^{\perp}$ is *not to complicated*, we can use Equation (3.3) to find $\mathbf{y}$.

Lemma 3.12. *With above notation, set $\mathcal{F} := \mathcal{L}_K(\widetilde{G^{\perp}} - B)$ where $B \in Div(K)$ is defined in Assumption A4. Then*

$$\mathcal{F} \subseteq \mathcal{L}_K(\widetilde{G^{\perp}})$$

and for all $y \in \mathcal{L}_L(\pi^*B)$ we have

$$\pi^*\mathcal{F} \cdot y \subseteq \mathcal{L}_L(G^{\perp}).$$

Proof. The first assertion comes from the fact that B is effective. As for the second, observe that for $g \in \mathcal{F}$ and $y \in \mathcal{L}_L(\pi^*B)$ we have

$$(\pi^*g \cdot y)^L = \pi^*(g)^K + (y)^L \geq \pi^*(-(\widetilde{G^{\perp}} - B)) - \pi^*B = -\pi^*\widetilde{G^{\perp}} \geq -G^{\perp}.$$

□

Let $g_1, ..., g_s$ be a basis of $\mathcal{F}$ over $\mathbb{F}_{q^m}$. For any $1 \leq k \leq s$, consider the evaluation vector

$$\mathbf{u}_k := (\pi^*g_k(Q_{1,1}), \ldots, \pi^*g_k(Q_{1,\ell}), \ldots, \pi^*g_k(Q_{r,1}), \ldots, \pi^*g_k(Q_{r,\ell}))$$

and denote by

$$\mathbf{D}_k := \mathrm{Diag}(\mathbf{u}_k)$$

the corresponding diagonal matrix of length n. Equation (3.3) can then be rewritten as a linear system:

Proposition 3.13. *We consider the vector $\mathbf{z}_{\omega} := \mathrm{Res}_{\omega}(Q_{i,j}))_{i,j}$, where $\omega \in \Omega_L$ satisfies Equation (3.3). For any $y \in \mathcal{L}_L(\pi^*B)$ with associated evaluation vector $\mathbf{y} = (y(Q_{i,j}))_{i,j}$, we have*

$$\begin{cases} \mathbf{M} \cdot \mathbf{D}_1 \cdot (\mathbf{z}_{\omega} \star \mathbf{y})^T = 0 \\ \qquad\vdots \\ \mathbf{M} \cdot \mathbf{D}_s \cdot (\mathbf{z}_{\omega} \star \mathbf{y})^T = 0, \end{cases} \tag{3.4}$$

which gives a set of ks equations of which $\mathbf{z}_{\omega} \star \mathbf{y}$ is solution, where $k = \dim_{\mathbb{F}_q} \mathcal{C}$.

Proof. For all $k \in \{1, \ldots, s\}$, Lemma 3.12 gives

$$\pi^*g_k \cdot y \in \mathcal{L}_L(G^{\perp}).$$

Evaluating at the places $Q_{i,j}$'s yields

$$\mathbf{u}_k \star \mathbf{y} \in C_{\mathcal{L}}(\mathcal{Y},\mathcal{Q},G^{\perp}).$$

Since $\mathbf{Z}_{\omega} \cdot C_{\mathcal{L}}(\mathcal{Y},\mathcal{Q},G^{\perp}) \subseteq \mathsf{SSAG}_q(\mathcal{Y},\mathcal{Q},G)^{\perp}$, we finally get

$$\mathbf{M} \cdot \mathbf{D}_k \cdot (\mathbf{z}_{\omega} \star \mathbf{y})^T = 0,$$

which gives the desired result. □

Remark 3.14. Depending on the quotient curve $\mathcal{X}$, it may be hard to find the exact divisor $\widetilde{G^\perp}$, especially if its support is hard to explicit. For this reason, we might in concrete instances use another divisor $D \in \mathrm{Div}(K)$ such that

- $D \leq \widetilde{G^\perp}$;
- $\widetilde{G^\perp} - D$ has small degree and
- D can be explicitly computed from our hypotheses.

Note that Lemma 3.12 (and thus the system (3.4)) still holds by replacing $\widetilde{G^\perp}$ with such a D.

We point out that several computations realized on Magma [BCP97] show that the system (3.4) usually does not have a unique solution. Thankfully, we can add other equations that are only satisfied by the vector we are searching, making use of the action of the automorphism acting on $\mathcal{Q}$, leaving it invariant. We will come back at this step later in concrete instances. Moreover, depending on the context, the residue vector $\mathbf{z}_\omega$ might be known from our hypothesis and choice of differential. Otherwise, solving the system (3.4) only allows us to recover the product vector $\mathbf{z}_\omega \star \mathbf{y}$.

Isolating the desired vector. If $\mathbf{z}_\omega$ is unknown, we have to isolate $\mathbf{y}$ from the product $\mathbf{z}_\omega \star \mathbf{y}$. To do so, we can build another linear system by changing slightly the space $\mathcal{F}$ defined in Lemma 3.12, so that $\mathbf{z}_\omega \star \mathbf{y}^2$ is a solution of it.

Lemma 3.15. *Let* $B = -(\widetilde{-(y^2)_\infty^L})$ *and set* $\mathcal{F}' := \mathcal{L}_K(\widetilde{G^\perp} - B')$. *Then*

$$\mathcal{F}' \subseteq \mathcal{L}_K(\widetilde{G^\perp})$$

and

$$\pi^*\mathcal{F} \cdot y^2 \subseteq \mathcal{L}_L(G^\perp).$$

Proof. Similar to the proof of Lemma 3.12. □

Repeating the same process as above (using this time a basis of the space $\mathcal{F}'$), we can recover $\mathbf{z}_\omega \star \mathbf{y}^2$. As any component of $\mathbf{z}_\omega$ is non zero, we conclude by computing

$$\mathbf{y} = (\mathbf{z}_\omega \star \mathbf{y}^2) \star (\mathbf{z}_\omega \star \mathbf{y})^{-1}.$$

Finishing the attack. To complete the security reduction, we make use of Proposition 3.9 to recover a defining equation of the cover, hence finding a projection morphism π and the minimal polynomial $H \in K[T]$ of a primitive element y of L over K. From this, we can also rebuild the divisor $G := \pi^*\widetilde{G}$ using Kummer's theorem or its corollary ([Sti09, Theorem 3.3.7 and Corollary 3.3.8]). Having now recovered the whole structure of the public SSAG code, this proves that under our assumptions, the security of the scheme reduces to the security of the invariant code.

Recap. Let us end up this somewhat formal section by summing up what is left to be precised in concrete instances, in preparation for the next section.

- How to chose the divisor B ? As discussed in Remark 3.10, we will only consider extensions where the divisor $(y)_\infty^L$ is perfectly known, in which case we can take $B = -(\widetilde{-(y)_\infty^L})$. This is in some sense the easiest case but it still provides some interesting results.
- How can we make a smart choice for the differential $\omega \in \Omega_L$ satisfying Equation (3.3) ? The ideal candidate is a differential satisfying

$$\forall Q_{i,j} \in \mathcal{Q}\ ,\ \nu_{Q_{i,j}}(\omega) = -1, \tag{3.5}$$

  while having a divisor which is as simple as possible (*i.e.* supported by only a few points). Since Ω_L is a one dimensional vector space over L (see Proposition 1.63, *2*), it implies a suitable choice of some function $h \in L$ such that $\omega = h dx$ (where x is a separating element of L over $\mathbb{F}_{q^m}$). In concrete instances, this can be done by studying the different divisor of $L/\mathbb{F}_{q^m}(x)$.
- How to ensure that $\mathbf{z}_\omega \star \mathbf{y}$ is the unique solution of the System (3.4) ? In Sections 3.3.2 and 3.3.3, we focus on extensions L/K which are cyclic generated by σ, in which cases the action on the public SSAG code is uniquely determined by the image $\sigma(y)$ of the primitive element y under σ. In particular, this means that each orbit of the unknown vector $\mathbf{y}$ has a specific structure (if $\sigma(y)$ is known, the full orbit can be recovered from only one entry). Our plan is to use this additional structure to add more equations to the system we have to solve, hopping to reduce the number of solutions.

3.3 Applications

In this section, we present our attack in concrete instances of a cyclic extension of function fields, which can be either a Kummer extension or an elementary abelian p–extension, depending whether the degree of the cover is prime to the characteristic of the base field or not. First, we give sufficient conditions on the quotient curve $\mathcal{X}$ to make the attack possible.

3.3.1 About the quotient curve

Below, we define a general class of curves that satisfy two properties, needed to enable our attack whenever the quotient curve $\mathcal{X}$ is in this class.

Definition 3.16. We denote by $\mathcal{B}$ the set of pairs $(\mathcal{X}, P_\infty)$, where $\mathcal{X}$ is a smooth and irreducible projective curve over $\mathbb{F}_{q^m}$ equipped with a point $P_\infty \in \mathcal{X}(\mathbb{F}_{q^m})$ such that:

1. There exists a morphism $\mathcal{X} \to \mathbb{P}^1$ which is totally ramified at infinity, and hence P_∞ is the unique point at infinity in $\mathcal{X}$.
2. There exists a function $h \in K := \mathbb{F}_{q^m}(\mathcal{X})$ such that
$$(h \cdot dx)^K = (2\mathfrak{g}(\mathcal{X}) - 2) \cdot P_\infty, \tag{3.6}$$
where x is a separating element of K over $\mathbb{F}_{q^m}$.

If α is a primitive element of K over the rational function field $\mathbb{F}_{q^m}(x)$, then the first property ensures that the divisor $(\alpha)_\infty^K$ is a multiple of P_∞, which will helps in choosing the divisor B. The second will be useful to find a good differential that satisfies Equation (3.2).
A fairly large class of curves that satisfy both conditions of Definition 3.16 is the class of $C_{a,b}$ curves (see Section 1.2.5): in fact,

- it has a unique point at infinity P_∞, since a and b are relatively prime,
- for any canonical divisor W on a $C_{a,b}$ curve, $W \sim (2\mathfrak{g}_{a,b} - 2)P_\infty$ (see [BESP10, Proposition 4.1]).

Note also that under technical assumptions, Kummer and Artin–Schreier curves can be seen as $C_{a,b}$ curves. In the upcoming sections, we apply our attack in the case where L/K is a Kummer extension (Section 3.3.2) or an elementary abelian p–extension (Section 3.3.3) of a curve $\mathcal{X}$ which is in the class $\mathcal{B}$.

3.3.2 Kummer covering

Setting. Let $(\mathcal{X}, P_\infty) \in \mathcal{B}$ be a curve satisfying both properties of Definition 3.16, $K = \mathbb{F}_{q^m}(\mathcal{X})$ and denote by $a = [K : \mathbb{F}_{q^m}(x)]$. If ∞ is the point at infinity in the rational function field, we have $P_\infty|\infty$ and $e(P_\infty|\infty) = a$. Given an integer ℓ (not necessarily a prime) such that $\gcd(\ell, \mathrm{char}(\mathbb{F}_{q^m})) = 1$ and $\ell \mid q^m - 1$, we consider the function fields extension $L = K(y)$, with

$$y^\ell = f,$$

where $f \in K$ is a function that satisfies the following properties:

K1. P_∞ is the only pole of f, and $d := -\nu_{P_\infty}(f)$ is prime to ℓ. Hence, $(f)_\infty^K = dP_\infty$;

K2. $\forall P \in \mathrm{Supp}\left((f)_0^K\right),\ \nu_P(f) = 1$.

L/K is a Kummer extension, cyclic of order $\ell = [L : K]$ and whose Galois group is given by

$$\mathrm{Gal}(L/K) = \{\sigma : y \mapsto \xi \cdot y \mid \xi \in \mu_\ell^*(\mathbb{F}_{q^m})\}.$$

We denote by $\mathcal{Y}$ the curve associated to L and $\pi : \mathcal{Y} \to \mathcal{X}$ the corresponding morphism.

Assuming $\ell \mid q^m - 1$ implies that the primitive ℓ–th roots of unity are indeed in $\mathbb{F}_{q^m}$. Let us discuss the assumptions made on the function f:

- first, assuming $\gcd(d, \ell) = 1$ ensures that P_∞ is totally ramified in L/K, which will be mandatory later on.

- Supposing that f as only one pole, which is P_∞, is not too much of a restriction since we can always get back to this case: in fact, if f has multiple poles, the Strong Approximation Theorem [Sti09, Theorem 1.6.5] guarantees that there exists a function $\alpha \in K^\times$, with P_∞ as only pole, and such that
$$(f\alpha)_\infty^K = (d + d')P_\infty,$$
where $d' := -\nu_{P_\infty}(\alpha) > 0$. Now, from Kummer's theory, we get the same extension by replacing the equation $y^\ell = f$ with $z^\ell = f'$, where $f' = f\alpha^\ell$ (since $\alpha^\ell \in (K^\times)^\ell$). If we do so, we have $(f')_\infty^K = (d + d'\ell)P_\infty$, which is prime to ℓ since d is. Hence, $f' = f\alpha$ satisfies K1. Note however that multiplying the equation by α^ℓ would create new zeroes, but we can also ensure that the zeroes of α differ from those of f.

- Finally, we make the assumption K2 for 2 reasons: it allows to describe simply both the ramification in L/K and the structure of the different divisor $\mathrm{Diff}(L/K)$. More precisely, it implies that the divisor $(\frac{y}{f})^L + \mathrm{Diff}(L/K)$ is only supported by the unique point at infinity in L. In Example 3.21, we show that the latter property does not hold if f has a multiple zero.

Remark 3.17. Notice that K2 exactly means that f as only simple zeroes. If $K = \mathbb{F}_{q^m}(x)$, then f is a univariate polynomial in x and it corresponds to suppose it is square–free, which is usually assumed while considering Kummer extensions of the rational function field.

In what follow, we describe in details the attack presented in Section 3.2 to recover the evaluation vector $\mathbf{y} = (y(Q_{i,j}))_{i,j}$. For our hypotheses, we take the same as in Section 3.2.1: more precisely, with notation of Proposition 3.9, we have $H(T) = T^\ell - f \in K[T]$. Moreover, we assume that K1 and K2 hold, but the function $f \in K$ is unknown.

The choice of ω. We start by studying the different of $L/\mathbb{F}_{q^m}(x)$. First, by transitivity of the different [Sti09, Corollary 3.4.12], we have

$$\mathrm{Diff}(L/\mathbb{F}_{q^m}(x)) = \pi^* \left(\mathrm{Diff}(K/\mathbb{F}_{q^m}(x))\right) + \mathrm{Diff}(L/K). \tag{3.7}$$

Since L/K is a Kummer extension, which has been extensively studied, we know exactly which places ramify.

Proposition 3.18. *With the above notation, let $P \in \mathbb{P}_K$ and $Q \in \mathbb{P}_L$ such that $Q|P$. Then*

$$e(Q|P) = \frac{\ell}{\gcd(\ell, \nu_P(f))} \text{ and } d(Q|P) = e(Q|P) - 1.$$

Proof. See [Sti09, Proposition 3.7.3]. □

Corollary 3.19. *Let $\mathcal{Z}(f)$ be the set of zeroes of f in $\mathbb{P}_K$. Under K1 and K2, we have*

$$(f)^K = \sum_{P \in \mathcal{Z}(f)} P - dP_\infty,$$

and thus

$$\mathrm{Diff}(L/K) = (\ell - 1)\left(\sum_{P \in \mathcal{Z}(f), Q|P} Q + Q_\infty\right),$$

where $Q_\infty|P_\infty$ is the unique extension of P_∞ in L.

Proof. Proposition 3.18 gives that any place $P \in \mathbb{P}_K \setminus \mathrm{Supp}(f)^K$ does not ramify. Now, for any $P \in \mathrm{Supp}(f)^K$, we have $\gcd(\ell, \nu_P(f)) = 1$ (using K1 for P_∞ and K2 for the zeroes of f), hence any such P totally ramifies in L/K. The result follows from the definition of the different (see Definition (1.56)). □

Proposition 3.20. *With notation as above, $(\mathcal{Y}, Q_\infty) \in \mathcal{B}$. Moreover, there exists $h \in K$ such that the differential*

$$\omega_0 := h \cdot \frac{y}{f} \cdot dx \in \Omega_L$$

satisfies

$$(\omega_0)^L = (2\mathfrak{g}(L) - 2) \cdot Q_\infty.$$

Proof. We have already seen that Q_∞ is the unique point at infinity in $\mathcal{Y}$. It remains to prove the second point of Definition 3.16: since $(\mathcal{X}, P_\infty) \in \mathcal{B}$, there exists $h \in K$ such that

$$(h \cdot dx)^K = (2\mathfrak{g}(K) - 2)P_\infty.$$

Next, notice that both K1 and K2 imply

$$(f)^L = \pi^*(f)^K = \ell \left(\sum_{P \in \mathcal{Z}(f), Q|P} Q - dQ_\infty \right),$$

since any place in $\mathrm{Supp}\left((f)^K\right)$ is totally ramified. From the defining equation $y^\ell = f$, we then get

$$\left(\frac{y}{f}\right)^L = (\ell - 1) \left(dQ_\infty - \sum_{P \in \mathcal{Z}(f), Q|P} Q \right).$$

Using Corollary 3.19 yields

$$\left(\frac{y}{f}\right)^L + \mathrm{Diff}(L/K) = (\ell - 1)(d+1) \cdot Q_\infty = \deg\left(\mathrm{Diff}(L/K)\right) \cdot Q_\infty, \tag{3.8}$$

the last equality coming from the linearity of the degree map and the fact that any principal divisor has degree zero. By definition of h and Equation (1.3), we have

$$(h)^K = (2\mathfrak{g}(K) - 2)P_\infty - (dx)^K = (2\mathfrak{g}(K) - 2)P_\infty + 2(x)^K_\infty - \mathrm{Diff}(K/\mathbb{F}_{q^m}(x)).$$

Now we take the pullback:

$$(h)^L = \ell(2\mathfrak{g}(K) - 2)Q_\infty + 2(x)^L_\infty - \pi^*\left(\mathrm{Diff}(K/\mathbb{F}_{q^m}(x))\right). \tag{3.9}$$

Adding both Equations (3.8) and (3.9), the transitivity of the different (3.7) finally gives

$$\begin{aligned}(\omega_0)^L &= (h)^L + \left(\frac{y}{f}\right)^L + (dx)^L \\ &= \left[\ell(2\mathfrak{g}(K) - 2) + \deg(\mathrm{Diff}(L/K))\right] Q_\infty \\ &= (2\mathfrak{g}(L) - 2)Q_\infty,\end{aligned}$$

the last equality coming from Corollary 1.68. □

In the following counter–example, we show in a simple case why the assumption K2 is important.

Counter-example 3.21. Let $\mathbb{F} := \mathbb{F}_{121}$. Denote by $K = \mathbb{F}(x)$ the rational function field and consider the Kummer extension $L = K(y)$, with

$$y^5 = f(x) := (x - \alpha)^3(x - \beta) \quad \text{and } \alpha \neq \beta \in \mathbb{F}.$$

Let P_α and P_β be the zeroes of $x - \alpha$ and $x - \beta$ in K, respectively. We easily check that $(f)^K_\infty = 4P_\infty$, where P_∞ is the pole of x. Remark that P_α, P_β and P_∞ are totally ramified, hence K1 hold but clearly K2 does not, as P_α is a zero of order 3. We have

$$\mathrm{Diff}(L/K) = 4(Q_\alpha + Q_\beta + Q_\infty),$$

where Q_α, Q_β and Q_∞ are the extensions of P_α, P_β and P_∞ respectively. A quick computation shows that $(\frac{y}{f})^L + \mathrm{Diff}(L/K)$ is not only supported by Q_∞, *i.e.*

$$\left(\frac{y}{f}\right)^L + \mathrm{Diff}(L/K) = 20Q_\infty - 8Q_\alpha.$$

With above notation, we can take $h = 1$ (K is the rational function field). Hence, the differential ω_0 defined in Proposition 3.20 satisfies

$$(\omega_0)^L = 10Q_\infty - 8P_\alpha.$$

Consequently, the formula for ω_0 does not work here. Even worse, we can check on Magma [BCP97] that there exist no function $z \in L$ such that

$$(zdx)^L = (2\mathfrak{g}(L) - 2)Q_\infty,$$

which entails $(\mathcal{Y}, Q_\infty) \notin \mathcal{B}$.

At this step, we now have access to a differential ω_0 which is rather simple since it is only supported by Q_∞. Of course, we still need to deal with the conditions of Equation (3.5), which is the point of the upcoming discussion.

Lemma 3.22. *Consider the sets*

$$\mathcal{R} := \left\{ R \in \mathbb{P}_{\mathbb{F}_{q^m}(x)} \mid \exists P \in \mathcal{P} \text{ such that } P|R \right\}, \ \mathcal{R}_K := \{ P \in \mathbb{P}_K \mid P|R, \text{ for some } R \in \mathcal{R} \},$$

and the function

$$\iota_{\mathcal{P}} := \prod_{R \in \mathcal{R}} (x - x(R)) \in \mathbb{F}_{q^m}(x).$$

Setting $r^* := \#\mathcal{R} \leq r$, where $r = \frac{n}{\ell}$ is the number of orbits in the unknown support $\mathcal{Q}$, we have

$$(\iota_{\mathcal{P}})^L = \pi^* D_{\mathcal{R}_K} - (\ell a r^*) \cdot Q_\infty,$$

where $D_{\mathcal{R}_K}$ is the divisor associated to $\mathcal{R}_K$ with respect to Notation 3.11.

Proof. By definition, we have

$$(\iota_{\mathcal{P}})^{\mathbb{F}_{q^m}(x)} = D_{\mathcal{R}} - r^* \infty,$$

where ∞ stands for the point at infinity in $\mathbb{F}_{q^m}(x)$. The result follows from the definition of $D_{\mathcal{R}_K}$ and the fact that Q_∞ is the only extension of ∞, with $e(Q_\infty|\infty) = \ell a$. □

Notice that $\mathcal{P} \subseteq \mathcal{R}_K$, the equality being attained if $K/\mathbb{F}_{q^m}(x)$ is Galois and if $\mathcal{P}$ is Galois–invariant, which is not necessarily the case (this justify the introduction of the integer r^*).

Proposition 3.23. *The differential $\omega := \iota_{\mathcal{P}}^{-1} \omega_0 \in \Omega_L$ satisfies*

$$(\omega)^L = (2\mathfrak{g}(L) - 2 + \ell a r^*) \cdot Q_\infty - \pi^* D_{\mathcal{R}_K}.$$

Moreover, for every $Q_{i,j} \in \mathcal{Q}$, we have $\nu_{Q_{i,j}}(\omega) = -1$.

Proof. Consequence of Proposition 3.20 and Lemma 3.22. The result about the valuation of ω at each $Q_{i,j}$ comes from the fact that $\nu_{Q_{i,j}}(\pi^* D_{\mathcal{R}_K}) = 1$. □

Corollary 3.24. *With above notation, let $A := D_{\mathcal{R}_K} - D_{\mathcal{P}} \in \mathrm{Div}(K)$. Then the dual divisor $G^\perp$ of G, as defined in Proposition 1.83, is given by*

$$G^\perp := D_{\mathcal{Q}} - G + (\omega)^L = (2\mathfrak{g}(L) - 2 + \ell a r^*) \cdot Q_\infty - G - \pi^* A.$$

Proof. Consequence of Proposition 3.23 and the fact that $\pi^* A = \pi^* D_{\mathcal{R}_K} - D_{\mathcal{Q}}$. □

The shape of the divisor $G^\perp$ being now known, it remains to study its pushforward $\widetilde{G^\perp}$ in order to apply Lemma 3.12.

Study of $\widetilde{G^\perp}$. To recover the evaluation vector $\mathbf{y}$, we need to have as many equations as possible in the system (3.4), meaning that we want $\mathcal{F}$ (defined in Lemma 3.12) to be as big as possible (the same holds for $\mathcal{F}'$ (see Lemma 3.15) if it is needed). It might be hard to compute directly the divisor $\widetilde{G^\perp}$, but we show below that we still can perform the attack by considering an alternative divisor $D \in \mathrm{Div}(K)$ such that $D \leq \widetilde{G^\perp}$ and $\widetilde{G^\perp} - D$ has small degree.

Proposition 3.25. *Following notation 3.11, we consider the divisors*

$$D := \left(\left\lfloor \frac{2\mathfrak{g}(L) - 2 + \ell a r^*}{\ell} \right\rfloor - 1 \right) \cdot P_\infty - \widetilde{G} - A - D_{\mathrm{Supp}(\widetilde{G})} - D_{\mathrm{Supp}(\widetilde{G}) \cap \mathrm{Supp}(A)} \quad \text{and} \quad B := -(\widetilde{-(y)^L_\infty}),$$

both lying in $\mathrm{Div}(K)$. *Then*

1. $B = \left\lceil \frac{d}{\ell} \right\rceil \cdot P_\infty$.
2. $D \leq \widetilde{G^\perp}$ *and* $\mathcal{L}_K(D - B) \subseteq \mathcal{L}_K(\widetilde{G^\perp})$.
3. *For all $y \in \mathcal{L}_L(\pi^* B)$ and $g \in \mathcal{L}_K(D - B)$, we have*

$$\pi^* g \cdot y \in \mathcal{L}_L(G^\perp).$$

4. The divisor $D - B \in \mathrm{Div}(K)$ can be computed from our hypotheses.

Proof. 1. Since $(f)_\infty^K = dP_\infty$, the defining equation of $\mathcal{Y}$ gives $\ell(y)_\infty^L = \pi^*(dP_\infty) = \ell d Q_\infty$. Thus, $(y)_\infty^L = dQ_\infty$, and hence

$$B = -(\widetilde{-(y)_\infty^L}) = -(\widetilde{-dQ_\infty}) = -\left\lfloor \frac{-d}{\ell} \right\rfloor P_\infty = \left\lceil \frac{d}{\ell} \right\rceil P_\infty.$$

2. First, recall that L/K is a Galois extension, hence for all $P \in \mathbb{P}_K$ and $Q \in \mathbb{P}_L$ with $Q|P$, the integer $e(Q|P) := e(P)$ does not depend on Q [Sti09, Corollary 3.7.2]. Keeping above notation, we know by definition that

$$A = \sum_{P \in \mathrm{Supp}(A)} P, \quad \text{and thus} \quad \pi^* A = \sum_{P \in \mathrm{Supp}(A), Q|P} e(Q|P) \cdot Q.$$

Using Corollary 3.24, we can write

$$G^\perp = (2\mathfrak{g}(L) - 2 + \ell ar^* - \nu_{Q_\infty}(G)) \cdot Q_\infty + \sum_{Q_\infty \neq Q \in \mathbb{P}_L} (-\nu_Q(G) - \nu_Q(\pi^* A))\, Q,$$

since we know that $Q_\infty \notin \mathrm{Supp}(\pi^* A)$ (but we might have $Q_\infty \in \mathrm{Supp}(G)$). By definition of the pushforward (cf. Definition 3.2) and its properties (cf. Lemma 3.4), we have

$$\widetilde{G^\perp} = \left\lfloor \frac{2\mathfrak{g}(L) - 2 + \ell ar^* - \nu_{Q_\infty}(G)}{\ell} \right\rfloor P_\infty + \sum_{P_\infty \neq P \in \mathbb{P}_K} \left(\underbrace{\min_{Q|P} \left\lfloor \frac{-\nu_Q(G) - \nu_Q(\pi^* A)}{e(Q|P)} \right\rfloor}_{:= \nu_P(\widetilde{G^\perp})} \right) P, \tag{3.10}$$

using the fact that $Q_\infty | P_\infty$ is totally ramified in L/K. By distinguishing cases, we can estimate the valuations $\nu_P(\widetilde{G^\perp})$ defined above for any P (at several steps of the computations, we will use the fact that for any $x, y \in \mathbb{R}$, we have $\lfloor x - y \rfloor \geq \lfloor x \rfloor - \lfloor y \rfloor - 1$):

(a) If $P = P_\infty$, we have

$$\nu_{P_\infty}(\widetilde{G^\perp}) \geq \left\lfloor \frac{2\mathfrak{g}(L) - 2 + \ell ar^*}{\ell} \right\rfloor - \nu_{P_\infty}(\widetilde{G}) - 1.$$

(b) If $P \in \mathrm{Supp}(\widetilde{G})$ and $P \notin \mathrm{Supp}(A)$, then

$$\nu_P(\widetilde{G^\perp}) = \min_{Q|P} \left\lfloor \frac{-\nu_Q(G)}{e(Q|P)} \right\rfloor \geq - \left\lfloor \frac{\nu_Q(G)}{e(P)} \right\rfloor - 1 = -\nu_P(\widetilde{G}) - 1.$$

(c) If $P \notin \mathrm{Supp}(\widetilde{G})$ and $P \in \mathrm{Supp}(A)$, we have by definition

$$\nu_P(\widetilde{G^\perp}) = \min_{Q|P} \left\lfloor \frac{-\nu_Q(\pi^* A)}{e(Q|P)} \right\rfloor = -1 = -\nu_P(A).$$

(d) In the case $P \in \mathrm{Supp}(\widetilde{G}) \cup \mathrm{Supp}(A)$, we have

$$\begin{aligned} \nu_P(\widetilde{G^\perp}) = \min_{Q|P} \left\lfloor \frac{-\nu_Q(G) - \nu_Q(\pi^* A)}{e(Q|P)} \right\rfloor &\geq \left\lfloor \frac{-\nu_Q(G)}{e(P)} \right\rfloor - \left\lfloor \frac{\nu_Q(\pi^* A)}{e(P)} \right\rfloor - 1 \\ &\geq - \left\lfloor \frac{\nu_Q(G)}{e(P)} \right\rfloor - 1 - \nu_P(A) - 1 \\ &\geq -\nu_P(\widetilde{G}) - \nu_P(A) - 2. \end{aligned}$$

(e) For any other P, we have $\nu_P(\widetilde{G^\perp}) = 0$.

Using these estimations in Equation (3.10) yields

$$\widetilde{G^\perp} \geq \left(\left\lfloor \frac{2\mathfrak{g}(L) - 2 + \ell ar^*}{\ell} \right\rfloor - 1 \right) \cdot P_\infty - \widetilde{G} - A - D_{\mathrm{Supp}(\widetilde{G})} - D_{\mathrm{Supp}(\widetilde{G}) \cap \mathrm{Supp}(A)} := D.$$

The inclusion $\mathcal{L}_K(D - B) \subseteq \mathcal{L}_K(\widetilde{G^\perp})$ follows from the fact that B is an effective divisor.

3. For $y \in \mathcal{L}_L(\pi^* B)$ and $g \in \mathcal{L}_K(D - B)$, we have

$$\begin{aligned}(\pi^* g \cdot y)^L &= \pi^*(g)^K + (y)^L \\ &\geq \pi^*(-(D-B)) - \pi^* B \\ &\geq -\pi^* D \\ &\geq -\pi^* \widetilde{G^\perp} \\ &\geq -G^\perp,\end{aligned}$$

since $D \leq \widetilde{G^\perp}$ from 2. This proves that $\pi^* g \cdot y \in \mathcal{L}_L(G^\perp)$.

4. The integer ℓ is known from A2; and A3 allows us to know explicitly a, r^* and both divisors $\widetilde{G}$ and A (in particular, we know their supports). Finally $\mathfrak{g}(L)$ can be computed from Hurwitz' formula (see Theorem 1.72).

□

The above proposition implies that we can construct D explicitly and consider $\mathcal{F} := \mathcal{L}_K(D - B)$ instead of $\mathcal{L}_K(\widetilde{G^\perp} - B)$ (as done in Lemma 3.12) to build the linear system (3.4). Notice that our choice of D is not too far from the true value of $\widetilde{G^\perp}$: in fact, the imprecision arises while bounding from below the valuations $\nu_P(\widetilde{G^\perp})$, using a classic inequality of the floor function, i.e.

$$\forall x, y \in \mathbb{R}, \lfloor x - y \rfloor \geq \lfloor x \rfloor - \lfloor y \rfloor - 1.$$

Obviously, we cannot compute directly the left parts since we do not know the valuations occurring in the decomposition of G and $\pi^* A$ (they are secret). By splitting these floor parts into two, we end up with known valuations (i.e. these of $\widetilde{G}$ and A). Built this way, the divisor $D - \widetilde{G^\perp}$ has a small enough degree, and can be used to build our linear system.

Adding geometric progression. To increase our chances to find a unique solution, we can add other equations to our system, using the action of the automorphism acting on $\mathcal{Q}$. In fact, since L/K is a Kummer extension, the automorphism σ acting on the code $\mathcal{C} := \mathsf{SSAG}_q(\mathcal{Y}, \mathcal{Q}, G)$ is completely determined by the choice of a primitive ℓ–th root unity ξ. Thus, the evaluation vector $\mathbf{y} = (y(Q_{i,j}))_{i,j}$ satisfies a geometric progression on each orbit of size ℓ (since we assumed by construction that $\mathcal{Q}$ is ordered by orbits of size ℓ). Let us clarify what we mean by *geometric progression*: for any $\xi \in \mu_\ell^*(\mathbb{F}_{q^m})$, we consider the following block matrix:

$$\mathbf{E}(\xi) = \begin{pmatrix} \mathbf{B}(\xi) & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \mathbf{B}(\xi) \end{pmatrix} \in \mathcal{M}_n(\mathbb{F}_{q^m}), \text{ where } \mathbf{B}(\xi) = \begin{pmatrix} \xi & -1 & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ 0 & & \ddots & \ddots & -1 \\ -1 & 0 & \cdots & 0 & \xi \end{pmatrix}$$

has size $\ell \times \ell$. If ξ is the root of unity that defines σ, then

$$\mathbf{E}(\xi) \cdot \mathbf{y}^T = 0.$$

The next lemma shows that the residue vector $\mathbf{z}_\omega$ also satisfy this geometric progression.

Lemma 3.26. *With the choice of differential made in Proposition 3.23, we have*

$$\forall Q_{i,j} \in \mathcal{Q},\ \mathrm{Res}_\omega(Q_{i,j}) = \frac{h(Q_{i,j}) y(Q_{i,j})}{f(Q_{i,j}) \prod\limits_{R \in \mathcal{R}, x(R) \neq x(Q_{i,j})} (x(Q_{i,j}) - x(R))}.$$

Proof. Clear since $\omega := \dfrac{h \cdot y}{f \cdot \prod\limits_{R \in \mathcal{R}} (x - x(R))} dx$. □

Since both f, h and x lie in K, they are invariant under the action of σ, which implies that the vector $\mathbf{z}_\omega$ satisfies the same geometric progression as $\mathbf{y}$, *i.e.* $\mathbf{E}(\xi) \cdot \mathbf{z}_\omega^T = 0$. Consequently, for any $i \in \{1, \cdots, r\}$ and $(Q_{i,1}, \cdots, Q_{i,\ell}) \in \mathcal{Q}$, we have

$$\mathrm{Res}_\omega(Q_{i,j+1}) \cdot y(Q_{i,j+1}) = \xi^2 \cdot \mathrm{Res}_\omega(Q_{i,j}) \cdot y(Q_{i,j}),$$

where $j \in \{1, \cdots, \ell\} \bmod \ell$. Adding this to our system yields

$$\begin{cases} \mathbf{E}(\xi^2) \cdot (\mathbf{z}_\omega \star \mathbf{y})^T &= 0 \\ \mathbf{M} \cdot \mathbf{D}_1 \cdot (\mathbf{z}_\omega \star \mathbf{y})^T &= 0 \\ \vdots & \\ \mathbf{M} \cdot \mathbf{D}_s \cdot (\mathbf{z}_\omega \star \mathbf{y})^T &= 0, \end{cases} \tag{3.11}$$

which is a collection of $n + ks$ equations for n unknowns. The new system (3.11) is over–constrained but we know that at least $\mathbf{z}_\omega \star \mathbf{y}$ is solution, provided that we guessed the good root of unity ξ. Since $\varphi(\ell) = |\mu_\ell^*(\mathbb{F}_{q^m})| = O(n)$, we can test all possible values of ξ until eventually we find the correct one, at the cost of at most a linear factor (in the length n of the code). Actually, it is difficult to give sufficient conditions for the system to have rank one, but it is very reasonable to hope it has a unique solution (up to scalar multiplication) when we picked the good root of unity, which always happened in our computing experiments. Additionally, picking a wrong ξ tends to give a system without any solution.

From now on, assume that we recovered the product vector $\mathbf{z}_\omega \star \mathbf{y}$ as well as the good root of unity $\xi \in \mu_\ell^*(\mathbb{F}_{q^m})$. Following the idea of Lemma 3.15, we build another linear system to recover $\mathbf{z}_\omega \star \mathbf{y}^{\mathbf{2}}$, obtained by replacing $\mathbf{E}(\xi^2)$ with $\mathbf{E}(\xi^3)$ in (3.11), *i.e.*

$$\begin{cases} \mathbf{E}(\xi^3) \cdot (\mathbf{z}_\omega \star \mathbf{y}^{\mathbf{2}})^T &= 0 \\ \mathbf{M} \cdot \mathbf{D}_1 \cdot (\mathbf{z}_\omega \star \mathbf{y}^{\mathbf{2}})^T &= 0 \\ \vdots & \\ \mathbf{M} \cdot \mathbf{D}_s \cdot (\mathbf{z}_\omega \star \mathbf{y}^{\mathbf{2}})^T &= 0. \end{cases} \tag{3.12}$$

Finally, we compute $\mathbf{y} = (\mathbf{z}_\omega \star \mathbf{y}^2) \star (\mathbf{z}_\omega \star \mathbf{y})^{-1}$. A plane model of $\mathcal{Y}$ can then be found by using an appropriate interpolation method (possible thanks to Proposition, 3.9).

A formal algorithm describing the attack can be found in Appendix A, and a MAGMA [BCP97] implementation can be found on a GitHub repository at https://github.com/Reiikar/attack_QC_SSAG_codes.

3.3.3 Elementary abelian p–extension

Setting. Let $(\mathcal{X}, P_\infty) \in \mathcal{B}$, $K = \mathbb{F}_{q^m}(\mathcal{X})$, $a := [K : \mathbb{F}_{q^m}(x)]$ and denote by p the characteristic of $\mathbb{F}_{q^m}$. Given a integer $u \geq 1$ such that $\mathbb{F}_{p^u} \subseteq \mathbb{F}_{q^m}$, we consider an elementary abelian p–extension L of K with degree p^u. From [GS91, Proposition 1.1], there exists $y \in L$ such that $L = K(y)$, with

$$y^{p^u} - y = f, \quad f \in K. \tag{3.13}$$

Similarly to Section 3.3.2, we need to assume that f satisfies K1, *i.e.*

$$(f)_\infty^K = dP_\infty \quad \text{and} \quad \gcd(d, \ell) = 1.$$

Remark 3.27. In this case, the hypothesis K2 is no longer required, as it does not impact the ramification in the extension: in any case, P_∞ is the only place that ramifies (see Proposition 3.29).

As we assumed $\mathbb{F}_{p^u} \subseteq \mathbb{F}_{q^m}$, the polynomial $T^{p^u} - T \in \mathbb{F}_{q^m}[T]$ has all its roots in $\mathbb{F}_{q^m}$. Hence, the extension L/K is Galois of order $p^u = [L : K]$ and

$$\mathrm{Gal}(L/K) = \{\sigma : y \mapsto y + \beta \mid \beta \in \mathbb{F}_{p^u}\}.$$

As usual, we denote by $\pi : \mathcal{Y} \to \mathcal{X}$ the corresponding morphism.

Remark 3.28. In the case $u = 1$, L/K is nothing but an Artin–Schreier extension (see [Sti09, Proposition 3.7.8]), which is cyclic of prime degree p (*i.e.* $\mathrm{Gal}(L/K) \simeq \mathbb{F}_p$).

In order to recover the evaluation vector

$$\mathbf{y} = (y(Q_{i,j}))_{i,j},$$

we take the same hypotheses as in Section 3.2.1, and assume that $L = K(y)$ with y as in Equation (3.13), *i.e* its minimal polynomial over K equals $H(T) = T^{p^u} - T - f \in K[T]$. Moreover, the function f is unknown, but we know it satisfies K1.

The choice of ω. As generalization of Artin–Schreier extensions, elementary abelian p–extensions are well–studied. In particular, the following proposition gives a description of the ramification in such extensions.

Proposition 3.29. *With above notation, P_∞ is the only place which ramifies in L/K. It has a unique extension $Q_\infty \in \mathbb{P}_L$, i.e. it is totally ramified. Moreover, its different exponent is given by*

$$d(Q_\infty|P_\infty) = (p^u - 1)(d+1).$$

Proof. Let $P \in \mathbb{P}_K$. Then from [Sti09, Proposition 3.7.10], we have:

1. either there exists $z \in K$ such that $\nu_P(f - (z^{p^u} - z)) \geq 0$, in which case we set $m_P := -1$;
2. or else, for some $z \in K$, we have $\nu_P(f - (z^{p^u} - z)) = -m < 0$ and $m \not\equiv 0[p]$. In this case, set $m_P := m$ (uniquely determined by f and P).

The integer m_P is well–defined for any place P, and [Sti09, Proposition 3.7.10 *(c)* and *(d)*] give that P is ramified if and only if $m_P > 0$, in which case it totally ramifies. In our setting, any $P \in \mathbb{P}_K \setminus \{P_\infty\}$ has $m_P = -1$, hence is unramified (we assumed P_∞ to be the only pole of f). From K1, the integer $m_{P_\infty} := d$ is prime to p, thus P_∞ is totally ramified. The formula for the different exponent is also given in [Sti09, Proposition 3.7.10 *(d)*]. □

Corollary 3.30. *The different of L/K is given by*

$$\mathrm{Diff}(L/K) = (p^u - 1)(d+1)Q_\infty.$$

Proof. Immediate consequence of the definition of the different (see Definition 1.56) and Proposition 3.29. □

Similarly to Kummer setting, we now show that $(\mathcal{Y}, Q_\infty) \in \mathcal{B}$ and provide a differential satisfying condition 2 of Definition 3.16.

Proposition 3.31. *With notation as above, $(\mathcal{Y}, Q_\infty) \in \mathcal{B}$. Moreover, there exists $h \in K$ such that the differential*

$$\omega_0 := h \cdot dx \in \Omega_L$$

satisfies

$$(\omega_0)^L = (2\mathfrak{g}(L) - 2) \cdot Q_\infty.$$

Proof. We already saw that Q_∞ is the unique point at infinity in $\mathcal{Y}$. From Definition 3.16, there exists $h \in K$ such that $(hdx)^K = (2\mathfrak{g}(K) - 2)P_\infty$. Applying Equation (1.3) yields

$$(2\mathfrak{g}(K) - 2)P_\infty = (h)^K - 2(x)^K_\infty + \mathrm{Diff}(K/\mathbb{F}_{q^m}(x)).$$

Taking the pullback, we get

$$(h)^L = p^u(2\mathfrak{g}(K) - 2)Q_\infty + 2(x)^L_\infty - \pi^* \left(\mathrm{Diff}(K/\mathbb{F}_{q^m}(x))\right).$$

Using Equation (1.3) again, this time applied to $(dx)^L$ gives

$$(dx)^L = -2(x)^L_\infty + \pi^* \left(\mathrm{Diff}(K/\mathbb{F}_{q^m}(x))\right) + \mathrm{Diff}(L/K).$$

Consequently, summing up the last two equalities and using Corollary 3.30 yields

$$\begin{aligned}(\omega_0)^L &= (h)^L + (dx)^L \\ &= p^u(2\mathfrak{g}(K) - 2)Q_\infty + (p^u - 1)(d+1)Q_\infty \\ &= (2\mathfrak{g}(L) - 2)Q_\infty,\end{aligned}$$

the last equality coming from Corollary 1.68. □

Keeping notation of Lemma 3.22, we proceed as in Section 3.3.2 to build the differential

$$\omega := \iota_{\mathcal{P}}^{-1} h \cdot dx \in \Omega_L \tag{3.14}$$

such that

$$\nu_{Q_{i,j}}(\omega) = -1 \quad \text{for all } Q_{i,j} \in \mathcal{Q}$$

and

$$(\omega)^L = (2\mathfrak{g}(L) - 2 + p^u a r^*) \cdot Q_\infty - \pi^* D_{\mathcal{R}_K}.$$

The divisors $G^\perp$ and $\widetilde{G^\perp}$. Setting $A := D_{\mathcal{R}_K} - D_{\mathcal{P}}$, we deduce from the above discussion that

$$G^\perp := D_{\mathcal{Q}} - G + (\omega)^L = (2\mathfrak{g}(L) - 2 + p^u a r^*) \cdot Q_\infty - G - \pi^* A.$$

As in the Kummer case, we build our linear system by using a divisor $D \in \mathrm{Div}(K)$ which is slightly smaller than $\widetilde{G^\perp}$:

Proposition 3.32. *Let*

$$D := \left(\left\lfloor \frac{2\mathfrak{g}(L) - 2 + p^u a r^*}{\ell} \right\rfloor - 1 \right) \cdot P_\infty - \widetilde{G} - A - D_{\mathrm{Supp}(\widetilde{G})} - D_{\mathrm{Supp}(\widetilde{G}) \cap \mathrm{Supp}(A)} \quad \text{and} \quad B := -(\widetilde{-(y)^L_\infty}),$$

both lying in $\mathrm{Div}(K)$. *Then*

1. $B = \left\lceil \frac{d}{p^u} \right\rceil \cdot P_\infty$.
2. $D \leq \widetilde{G^\perp}$ *and* $\mathcal{L}_K(D - B) \subseteq \mathcal{L}_K(\widetilde{G^\perp})$.
3. *For all* $y \in \mathcal{L}_L(\pi^* B)$ *and* $g \in \mathcal{L}_K(D - B)$, *we have*
$$\pi^* g \cdot y \in \mathcal{L}_L(G^\perp).$$
4. *The divisor* $D - B \in \mathrm{Div}(K)$ *can be computed from our hypotheses.*

Proof. Similar to Proposition 3.25. □

Adding arithmetic progression. One of the main difference with Section 3.3.2 is of course the action of the Galois group of L/K. By assumption, the code $\mathcal{C} = \mathsf{SSAG}_q(\mathcal{Y}, \mathcal{Q}, G)$ is invariant under the action of some order p^u automorphism $\sigma : y \mapsto y + \beta$, with $\beta \in \mathbb{F}_{p^u}$. As a consequence, the vector $\mathbf{y} = (y(Q_{i,j}))_{i,j}$ satisfies an arithmetic progression on each orbit of size p^u. More precisely, consider the following block matrices:

$$\mathbf{C} = \begin{pmatrix} \mathbf{B} & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \mathbf{B} \end{pmatrix} \in \mathcal{M}_n(\mathbb{F}_{q^m}), \text{ where } \mathbf{B} = \begin{pmatrix} -1 & 1 & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & 0 \\ 0 & & & \ddots & 1 \\ 1 & 0 & \cdots & 0 & -1 \end{pmatrix} \in \mathcal{M}_{p^u}(\mathbb{F}_{q^m}).$$

If $\beta \in \mathbb{F}_{p^u}$ is the element that defines the automorphism acting on $\mathcal{C}$, then

$$C \cdot \mathbf{y}^T = \boldsymbol{\beta},$$

where $\boldsymbol{\beta} = (\beta, \ldots, \beta)^T$ is a column vector of size $n = rp^u$.
Our choice of differential (Equation (3.13)) implies

$$\mathrm{Res}_\omega(Q_{i,j}) = \frac{h(Q_{i,j})}{\prod\limits_{R \in \mathcal{R}, x(R) \neq x(Q_{i,j})} (x(Q_{i,j}) - x(R))}, \quad \forall\, Q_{i,j} \in \mathcal{Q}. \tag{3.15}$$

It is then clear that the residue vector does not satisfy the arithmetic progression under each orbit: in fact, from Equation (3.15), the residues of ω are equal on each given orbit (since both h and $\iota_{\mathcal{P}}$ lie in K). The latter property is helpful as it means that $\mathbf{z}_\omega$ is known under our hypotheses. The final linear system we have to solve is then

$$\begin{cases} \mathbf{C} \cdot \mathbf{y}^T & = \boldsymbol{\beta} \\ \mathbf{M} \cdot \mathbf{D}_1 \cdot \mathbf{Z}_\omega \cdot \mathbf{y}^T & = 0 \\ \vdots & \\ \mathbf{M} \cdot \mathbf{D}_s \cdot \mathbf{Z}_\omega \cdot \mathbf{y}^T & = 0, \end{cases} \tag{3.16}$$

where $\mathbf{Z}_\omega$ is the diagonal matrix of length n corresponding to $\mathbf{z}_\omega$. Again, the system (3.16) is a collection of $n + ks$ equations with n unknowns, which has to be solved at most $p^u = 0(n)$ times, until we guessed the good value of β. Again, this would at most increase the complexity by a linear factor.

3.4 Generalization to solvable Galois cover

In this last section, we discuss how the attack presented in Section 3.2 could be generalized to any solvable Galois cover of curves. In Sections 3.3.2 and 3.3.3, we instantiate the attack in both Kummer and abelian p–extensions, which are interesting since they characterize in a sense all cyclic extensions: in fact, given an extension L/K of function fields over $\mathbb{F}_{q^m}$ with degree $a = [L : K]$, then

- if $\gcd(a, p) = 1$ and $a \mid q^m - 1$, then L/K is a Kummer extension of degree a (see [Sti09, Annex A.13]);
- if $a = p^u$ and $\mathbb{F}_{p^u} \subseteq \mathbb{F}_{q^m}$, then L/K is an elementary abelian p–extension of degree p^u (see [GS91, Proposition 1.1]).

Let us explain how we could deal with the solvable Galois case: let $(\mathcal{X}, P_\infty)$, and consider a cover $\pi : \mathcal{Y} \to \mathcal{X}$ (corresponding to an extension L/K of function fields) of curves over $\mathbb{F}_{q^m}$, such that $\mathcal{G} := \mathrm{Gal}(L/K)$ is solvable. By definition, there exists a sequence of normal subgroups

$$\{Id\} := \mathcal{G}_0 \lhd \mathcal{G}_1 \lhd \cdots \lhd \mathcal{G}_s := \mathcal{G}, \tag{3.17}$$

such that any quotient $\mathcal{G}_{i+1}/\mathcal{G}_i$ in Equation (3.17) is cyclic of degree $n_i := m_i p^{r_i} \in \mathbb{N}$, where m_i is prime to p. For any $i \in \{0, \ldots, s\}$, we denote by $L_i := L^{\mathcal{G}_i}$ the fixed field by $\mathcal{G}_i$. From Galois theory, the sequence (3.17) leads to a tower of function fields

$$K := L_s \subseteq L_{s-1} \subseteq \cdots \subseteq L_0 := L \tag{3.18}$$

such that for every $0 \leq i \leq s-1$, the extension L_i/L_{i+1} is cyclic, with Galois group $\mathcal{G}_{i+1}/\mathcal{G}_i$ and degree n_i. Equivalently, this corresponds to a tower of $\mathbb{F}_{q^m}$–curves

$$\mathcal{Y} := \mathcal{X}_0 \longrightarrow \mathcal{X}_1 \longrightarrow \cdots \longrightarrow \mathcal{X}_s := \mathcal{X} \tag{3.19}$$

such that each curve $\mathcal{X}_i$ is equipped with the action of the cyclic group $\mathcal{G}_{i+1}/\mathcal{G}_i$, and $\mathcal{X}_{i+1}$ is the corresponding quotient curve.

Now, assume that we want to show that the security of a public $\mathcal{G}$–invariant SSAG code

$$\mathcal{C}_0 := \mathsf{SSAG}_q\left(\mathcal{X}_0, \mathcal{Q}_0, G_0\right)$$

can be reduced to the security of its invariant subcode

$$\mathcal{C}_s := C_0^{\mathcal{G}} = \mathsf{SSAG}_q\left(\mathcal{X}_s, \mathcal{Q}_s, G_s\right).$$

As both $\mathcal{Q}_0$ and G_0 are assumed to be globally $\mathcal{G}$–invariant, they are also invariant under the action of each subgroup $\mathcal{G}_i$ appearing in the sequence (3.17). Consequently, there exists a sequence of invariant subcodes of $\mathcal{C}_0$, say $(\mathcal{C}_i)_{i=1}^s$such that for all $0 \leq i \leq s-1$,

$$\mathcal{C}_i := \mathcal{C}_{i+1}^{\mathcal{G}_{i+1}/\mathcal{G}_i}.$$

As the invariant subcode of an SSAG code is still an SSAG code (Corollary 3.8), each $\mathcal{C}_i$ is itself an SSAG code, *i.e.*

$$\mathcal{C}_i = \mathsf{SSAG}_q\left(\mathcal{X}_i, \mathcal{Q}_i, G_i\right),$$

for some (invariant) support $\mathcal{Q}_i$ and divisor G_i. Keeping the assumptions A1,A2 and A3 of Section 3.2.1, we want to show that we can recover the structure of $\mathcal{C}_0$ from whose of its (full) invariant code $\mathcal{C}_s$. To do so, we could proceed the following way: from the knowledge of $\mathcal{C}_s$ and since $\mathcal{X}_{s-1} \to \mathcal{X}_s$ is Galois with degree $n_s = m_s p^{r_s}$, we may apply successively the attack in Kummer case with degree m_i (Section 3.3.2) and abelian p–extension case with degree p^{r_i} (Section 3.3.3). Hopefully, we end up with an equation of the intermediary curve $\mathcal{X}_{s-1}$, from which we manage to build the code $\mathcal{C}_{s-1}$. Repeating this process, we could ride up the sequence (3.19) until recovering the secret structure of $\mathcal{C}_0$.

However, this method might work only if at each step, we can efficiently verify that our extensions satisfy conditions K1 and K2 (Kummer case) or K1 (abelian p–extension case). Actually, there is no reason for it to be true, and it is not reasonable to assume from the beginning that these conditions are true for all intermediary extensions.

As future work, it could be promising to find the largest class of cyclic covers $\mathcal{Y} \to \mathcal{X}$ such that $(\mathcal{Y}, Q_\infty) \in \mathcal{B}$, whenever $(\mathcal{X}, P_\infty) \in \mathcal{B}$. Finding a characterization of such covers would give a step further to this generalization.

Chapter 4

Goppa–like SSAG codes distinguisher

In this Chapter, in introduce a new family of codes that can be used in a McEliece cryptosystem, called *Goppa–like AG codes*. These codes generalize classical Goppa codes and can be constructed from any curve of genus $\mathfrak{g} \geq 0$. Focusing on codes from $C_{a,b}$ curves, we study the behaviour of the dimension of the square of their dual to determine their resistance to distinguisher attacks similar to the one for alternant and Goppa codes developed by Mora and Tillich [MT21]. In this paper, the authors managed to get a sharp upper bound by performing Euclidean division by powers of the multiplicator g in the case of classical Goppa codes. Considering one–point Goppa–like AG codes defined on a $C_{a,b}$ curve, we prove that performing division algorithms via Gröbner bases enables us to obtain similar results. Even better, computations tend to show that the bound we obtain on the dimension is sharp whenever the code seems random, hence generalizing the distinguisher proposed in [MT21]. The counterpart is that our distinguisher suffers the same problem, *i.e.* we can only distinguish high rate codes.

The chapter is organized as follows. In Section 4.1, we give a first upper bound on the dimension of the square of the dual of an SSAG code, using an old result on Riemann–Roch spaces due to Mumford [Mum70]. Section 4.2 is dedicated to Goppa–like AG codes: after motivating their definition, we then bound from above the dimension of the square of their dual, as done in [MT21] in the case of alternant codes. In Section 4.3, we refine the bound in the case of Goppa–like codes from $C_{a,b}$ curves, associated to one–point divisors. Our results are then analyzed in Section 4.4, where we discuss our bound in the case of codes from elliptic curves and the Hermitian curve. In the latter case, we study the effectiveness of our codes applied to a McEliece's cryptosystem.

Throughout the whole chapter, we fix a finite field $\mathbb{F}_{q^m}$ of characteristic $p > 0$, where q is a power of p and $m \geq 1$. To make the notation less cumbersome, we write $\mathrm{Tr}(\cdot)$ instead of $\mathrm{Tr}_{\mathbb{F}_{q^m}/\mathbb{F}_q}(\cdot)$ (see Section 1.1.2 for a definition of the Trace operator).

4.1 First estimation of the dimension of the square of the trace of an SSAG code

In [MT21], the authors benefited from the fact that the square of GRS codes is abnormally small, *i.e* the inequality in Equation (1.1) is always strict. More precisely, if $\mathsf{GRS}_r(\mathbf{x}, \mathbf{y}) \subseteq \mathbb{F}_{q^m}^n$ is a dimensional $r \leq \frac{n+1}{2}$ code, then

$$\dim_{\mathbb{F}_{q^m}} \mathsf{GRS}_r(\mathbf{x}, \mathbf{y})^{\star 2} = 2r - 1.$$

This fact is then used to get a first estimation of the square of the dual of an alternant code, via Proposition 1.19. Below, we show that AG codes, as generalization of GRS ones, benefit from the same structure with respect to the Schür product. For the remaining of this section, let $\mathcal{X}$ be a smooth and irreducible projective curve over $\mathbb{F}_{q^m}$ with genus $\mathfrak{g} := \mathfrak{g}(\mathcal{X})$.

Proposition 4.1 ([Mum70, Theorem 6])**.** *Let F, G be two divisors on $\mathcal{X}$ such that* $\deg(G) \geq 2\mathfrak{g} + 1$ *and* $\deg(F) \geq 2\mathfrak{g}$*. Then*

$$\mathcal{L}(F) \cdot \mathcal{L}(G) = \mathcal{L}(F + G),$$

where $\mathcal{L}(F) \cdot \mathcal{L}(G) := \mathrm{Span}\,(f \cdot g : (f, g) \in \mathcal{L}(F) \times \mathcal{L}(G))$.

As a consequence, given an AG code $C_{\mathcal{L}}(\mathcal{X}, \mathcal{P}, G)$, we have

$$C_{\mathcal{L}}(\mathcal{X}, \mathcal{P}, G)^{\star 2} \subseteq C_{\mathcal{L}}(\mathcal{X}, \mathcal{P}, 2G),$$

with equality if $\deg(G) \geq 2\mathfrak{g}+1$. If $\deg(G) \geq \mathfrak{g}$, applying the Riemann–Roch theorem to the divisors G and $2G$ yields

$$\dim_{\mathbb{F}_{q^m}} C_{\mathcal{L}}(\mathcal{X},\mathcal{P},G)^{\star 2} \leq \dim_{\mathbb{F}_{q^m}} C_{\mathcal{L}}(\mathcal{X},\mathcal{P},2G) = 2\deg(G)+1-\mathfrak{g}, \tag{4.1}$$

which is much smaller than the expected dimension given in Equation (1.1). As the dual of an AG code is also an AG code (see Proposition 1.83), Proposition 1.19 can also give information on the subfield subcode of the AG code $\mathcal{C} := C_{\mathcal{L}}(\mathcal{X},\mathcal{P},G)$ (using the correspondence of Delsarte's theorem 1.12), *i.e*

$$\left(\mathsf{SSAG}_q\left(\mathcal{X},\mathcal{P},G^{\perp}\right)^{\perp}\right)^{\star 2} \subseteq \sum_{i=0}^{\lfloor \frac{m}{2} \rfloor} \mathrm{Tr}\left(\mathcal{C}\star\mathcal{C}^{q^i}\right). \tag{4.2}$$

Corollary 4.2. *Let $\mathcal{C} := C_{\mathcal{L}}(\mathcal{X},\mathcal{P},G)$ be a k–dimensional AG code on $\mathcal{X}$ associated with a degree $s \geq \mathfrak{g}$ divisor. Then*

$$\dim_{\mathbb{F}_q}\left(\mathsf{SSAG}_q\left(\mathcal{X},\mathcal{P},G^{\perp}\right)^{\perp}\right)^{\star 2} \leq \binom{mk+1}{2} - \frac{m}{2}(k(k-1)-2s).$$

Proof. From Equation (4.1), we have $\dim_{\mathbb{F}_{q^m}}(\mathcal{C})^{\star 2} \leq 2s+1-\mathfrak{g} \leq k+s$. Thus, Corollary 1.20 yields

$$\begin{aligned}\dim_{\mathbb{F}_q} \mathrm{Tr}(\mathcal{C})^{\star 2} \leq m(k+s) + \binom{m}{2}k^2 &= (2k+2s+mk^2-k^2)\frac{m}{2} \\ &= (k(mk+1)-k^2+k+2s)\frac{m}{2} \\ &= \binom{mk+1}{2} - \frac{m}{2}(k(k-1)-2s).\end{aligned}$$

□

According to the above corollary, the dimension of the square of the dual of an SSAG code is less than the expected value for random mk–dimensional codes (which is $\binom{mk+1}{2}$), due to the algebraic structure of AG codes. However, this bound does not fully benefit from this rich structure, notably the following property.

Lemma 4.3. *Let $\mathcal{C} := C_{\mathcal{L}}(\mathcal{X},\mathcal{P},G)$ be a k–dimensional AG code on $\mathcal{X}$. For every $i \geq 0$, we have*

$$\mathcal{C}\star\mathcal{C}^{q^i} \subseteq C_{\mathcal{L}}(\mathcal{X},\mathcal{P},(q^i+1)G)$$

Proof. Fix $i \geq 0$ and let $f_1, f_2 \in \mathcal{L}(G)$. Then the product $f_1 f_2^{q^i}$ belong to $\mathcal{L}((q^i+1)G)$ as

$$(f_1 f_2^{q^i}) + (q^i+1)G = ((f_1)+G) + q^i\left((f_2)+G\right) \geq 0.$$

This proves the inclusion of spaces

$$\mathcal{L}(G)\cdot\mathcal{L}(G)^{q^i} \subseteq \mathcal{L}((q^i+1)G),$$

hence the inclusion of the associated codes. □

Remark 4.4. The property above for $i=0$ follows from Proposition 4.1. To the best of our knowledge, there is no sufficient criterion for the equality for any $i \geq 1$ to hold in the literature. Given a basis $\{f_1,\ldots,f_k\}$ of the Riemann–Roch space $\mathcal{L}(G)$, the vector space $\mathcal{L}(G)\cdot\mathcal{L}(G)^{q^i}$ is spanned by the set $\left\{f_u f_v^{q^i} \mid 1 \leq u \leq v \leq k\right\}$. From our experiments, it may happen that the cardinality of this family is larger than $\ell((q^i+1)G)$ without the equality holding, which means that these generators may be linearly dependent in $\mathcal{L}((q^i+1)G)$ and do not form a basis of $\mathcal{L}(G)\cdot\mathcal{L}(G)^{q^i}$.

Thanks to Lemma 4.3, it will be possible to better handle the terms $\mathrm{Tr}\left(\mathcal{C}\star\mathcal{C}^{q^i}\right)$ in Equation (4.2). In the next section, we improve the bound of Corollary 4.2 in some specific cases.

4.2 Goppa–like AG codes

4.2.1 Definition, parameters and context in the literature

The codes and their parameters. Let D be an effective divisor of positive degree s on a smooth and irreducible projective curve $\mathcal{X}$ over $\mathbb{F}_{q^m}$. Denote by $K = \mathbb{F}_{q^m}(\mathcal{X})$ the function field of $\mathcal{X}$ and take a rational function $g \in K$ such that $g \notin \mathcal{L}(D)$. Given a set of rational points $\mathcal{P} \subseteq \mathcal{X}(\mathbb{F}_{q^m})$ such that $\mathcal{P} \cap \mathrm{Supp}(g) = \varnothing$ and $\mathcal{P} \cap \mathrm{Supp}(D) = \varnothing$, we consider the AG code

$$\mathcal{C} := \mathcal{C}_{\mathcal{L}}(\mathcal{X}, \mathcal{P}, D + (g)) = \left\{ \mathrm{ev}_{\mathcal{P}}(fg^{-1}) \mid f \in \mathcal{L}(D) \right\}.$$

From now on, we also set $G := D + (g)$.

Definition 4.5. The *Goppa–like* AG code associated to $\mathcal{C}$ is defined as the subfield subcode of its dual code, *i.e.*

$$\Gamma(\mathcal{P}, D, g) := \mathcal{C}^{\perp}|_{\mathbb{F}_q}.$$

Such a code has length $n = \#\mathcal{P}$. As stated in [JM96, Theorem 1], if $2\mathfrak{g} - 2 < \deg D < n$, its dimension satisfies

$$\dim_{\mathbb{F}_q} \Gamma(\mathcal{P}, D, g) \geq n - m \dim_{\mathbb{F}_{q^m}} \mathcal{C}_{\mathcal{L}}(\mathcal{X}, \mathcal{P}, D + (g)) = n - m(\deg(D) - \mathfrak{g} + 1).$$

As for its minimum distance, it is bounded from below by $\deg(D) - 2\mathfrak{g} + 2$.

Remark 4.6. These estimations of the dimension and the minimum distance may be improved by Theorem 1.85 and 1.87. Regarding the dimension, it is worth noting that, since $g \notin \mathcal{L}(D)$, the divisor $G = D + (g)$ is not effective. Hence, any divisor G_1 satisfying the conditions of Equation (1.10) is also non–effective, which means that

$$\dim_{\mathbb{F}_q} \Gamma(\mathcal{P}, D, g) \geq n - m\left(\ell(G) - \ell(G_1)\right).$$

Without additional conditions on the divisor D and the function g, the divisor G_U for G defined in Theorem 1.87 is zero. Generally, we cannot expect for a better bound for the minimum distance.

Why the terminology *Goppa–like*? In [JM96], Janwa and Moreno define *Goppa codes* on smooth and irreducible projective curves. Compared to their definition, Definition 4.5 introduces a function g which defines a multiplicator for the AG code over $\mathbb{F}_{q^m}$ that is algebraically related to the support $\mathcal{P}$.

Introducing this function g facilitates the use of SSAG as public keys for McEliece cryptosystems. Given an error correcting capability t, we can fix a divisor D whose degree satisfies $\deg(D) \geq 2t + 2\mathfrak{g} + 1$. Then the family of codes in which the public key is picked can be defined by running a family of functions g outside $\mathcal{L}(D)$.

The terminology *Goppa–like AG codes* instead of simply Goppa codes is motivated by the fact that we want to emphasize the use of a different curve than the projective line $\mathbb{P}^1$, like we differentiate AG codes from Reed–Solomon codes. In our definition, the rational function g plays the role of the Goppa polynomial. As described in [Sti09, Example 9.1.8], Goppa codes are nothing but Goppa–like AG codes from the projective line $\mathcal{X} = \mathbb{P}^1$. In fact, given $r \geq 0$, recall that the Generalized Reed–Solomon (GRS) code of dimension r, support $\mathbf{x} \in \mathbb{F}_{q^m}^n$ and multiplier $\mathbf{y} \in (\mathbb{F}_{q^m}^*)^n$ is defined as (see Section 1.3.4)

$$\mathsf{GRS}_r(\mathbf{x}, \mathbf{y}) = \{(y_1 f(x_1), y_2 f(x_2), \ldots, y_n f(x_n)) \mid f \in \mathbb{F}_{q^m}[X] \text{ such that } \deg(f) < r\}.$$

Take a univariate polynomial g of degree r such that $g(x_i) \neq 0$ for every $i \in \{1, \ldots, n\}$. Then the Goppa code of order r and support $\mathbf{x} \in \mathbb{F}_{q^m}^n$ is defined as

$$\Gamma_r(\mathbf{x}, g) = \mathsf{GRS}_r(\mathbf{x}, \mathbf{y})^{\perp}|_{\mathbb{F}_q}$$

where $\mathbf{y} = (g(x_1)^{-1}, g(x_2)^{-1}, \ldots, g(x_n)^{-1})$. As usual, represent the $\mathbb{F}_{q^m}$–points of $\mathbb{P}^1$ by the couples $\mathbb{P}^1(\mathbb{F}_{q^m}) = \{[1 : x] \mid x \in \mathbb{F}_{q^m}\} \cup \{P_\infty\}$ for $P_\infty = [0 : 1]$. Take $\mathcal{P} = \{[1 : x_1], [1 : x_2], \ldots, [1 : x_n]\}$ and $D = (r - 1)P_\infty$. Finally, the polynomial g can be seen as a function on $\mathbb{P}^1$ which lies in $\mathcal{L}(rP_\infty)$ but not in $\mathcal{L}((r - 1)P_\infty)$. Then both constructions match: $\Gamma_r(\mathbf{x}, g) = \Gamma(\mathcal{P}, D, g)$.

Relation with Cartier codes. Cartier codes [Cou14] are also defined as a geometric realization of Goppa codes, since well–known properties of Goppa codes naturally extend to them. For instance, Theorem 1.87 holds for Cartier code without assumption on the degree of the divisor.

The link with Goppa–like AG codes is the following: by definition, a Cartier code is a subcode of the subfield subcode of a differential code (see [Cou14, Proposition 4.3]), which actually means that for the good choice of divisor, a Cartier code is a subcode of the corresponding Goppa–like AG code. Moreover, [Cou14, Theorem 5.1] provides a sufficient condition for both constructions to be equal. More precisely, let us consider a Goppa–like AG code $\Gamma(\mathcal{P}, D, g)$ and $G = D + (g)$. Then the Cartier code $\mathrm{Car}_q(\mathcal{P}, G)$ (see [Cou14, Definition 4.2]) satisfies $\mathrm{Car}_q(\mathcal{P}, G) \subseteq \Gamma(\mathcal{P}, D, g)$, and

$$\dim_{\mathbb{F}_q} \left(\Gamma(\mathcal{P}, D, g)/\mathrm{Car}_q(\mathcal{P}, G)\right) \leq m \cdot i(G_1),$$

where G_1 is any divisor such that

$$G \geq qG_1 \text{ and } G \geq G_1. \tag{1.10}$$

Above, $i(G_1)$ stands for the index of speciality of G_1 (see [Sti09, Definition 1.6.10]). By the Riemann–Roch theorem, if $\deg(G_1) > 2\mathfrak{g} - 2$, then $i(G_1) = 0$. Thus, using Remark 1.86, the Cartier code $\mathrm{Car}_q(\mathcal{P}, G)$ coincides with the Goppa–like AG code $\Gamma(\mathcal{P}, D, g)$ whenever $\deg \left\lfloor \frac{G}{q} \right\rfloor > 2\mathfrak{g} - 2$.

Example 4.7. Let $q = 3$ and $m = 4$. We consider the elliptic curve defined over $\mathbb{F}_{q^m}$ by the equation

$$E : y^2 = x^3 + 2x + 1.$$

As usual, we denote by P_∞ the unique point at infinity on E. We take D as a one–point divisor supported by P_∞, *i.e* $D = sP_\infty$. For a fixed degree $s = 7$, we consider two choices of function g:

1. First, let $g_1 = x^4 \notin \mathcal{L}(D)$ (since $\nu_{P_\infty}(g_1) = -8$). In this case, g has 2 zeros (say P_1 and P_2), which are those of x. More precisely, we have

   $$(g_1) = 4(P_1 + P_2) - 8P_\infty.$$

   Hence, if we set $G = D + (g_1)$, the divisor $\left\lfloor \frac{G}{q} \right\rfloor = P_1 + P_2 - P_\infty$ has degree $1 > 2g(E) - 2 = 0$, meaning that the codes $\Gamma(\mathcal{P}, D, g_1)$ and $\mathrm{Car}_3(\mathcal{P}, G)$ are equal over $\mathbb{F}_3$.

2. Second, take $g_2 = x^4 + xy$. Again, we have $g_2 \notin \mathcal{L}(D)$ since it has same valuation at P_∞ as g_1. This time, we can verify that g_2 has 8 distincts rational zeros, say $R_1, \ldots, R_8$, *i.e*

   $$(g_2) = R_1 + \cdots + R_8 - 8P_\infty.$$

   Consequently, we easily check that $\deg\left(\left\lfloor \frac{G}{q} \right\rfloor\right) = -1 \leq 2g(E) - 2$. Some computations realized on Magma show that the codes $\Gamma(\mathcal{P}, D, g_2)$ and $\mathrm{Car}_3(\mathcal{P}, G)$ are not equal in this case.

Magma results are summarized in Table 4.1.

n	$s = \deg(D)$	Choice of g	$\deg\left(\left\lfloor \frac{G}{q} \right\rfloor\right)$	$\dim_{\mathbb{F}_3}(\Gamma(\mathcal{P}, D, g))$	$\dim_{\mathbb{F}_3}(\mathrm{Car}_3(\mathcal{P}, G))$
88	7	$g = x^4$	1	64	64
82	7	$g = x^4 + xy$	-1	50	54

Table 4.1: Comparison of Cartier and Goppa–like constructions

4.2.2 On the dimension of the square of the dual of a Goppa–like AG code

In this section, we aim to generalize the properties found by the authors of [MT21] (Section 6), in the context of Goppa–like AG codes. To do so, we consider the AG code

$$\mathcal{C} := \mathcal{C}_{\mathcal{L}}(\mathcal{X}, \mathcal{P}, D + (g))$$

as in Definition 4.5. Since $\Gamma(\mathcal{P}, D, g)$ is the subfield subcode of $\mathcal{C}^\perp$, putting it into Equation (4.2) yields

$$\left(\Gamma(\mathcal{P}, D, g)^\perp\right)^{\star 2} \subseteq \sum_{i=0}^{\lfloor m/2 \rfloor} \mathrm{Tr}\left(\mathcal{C} \star \mathcal{C}^{q^i}\right). \tag{4.3}$$

Below, we discuss how to improve the upper bound given in Corollary 4.2, which is valid for all subfield subcodes of AG codes. The idea is to use the specific algebraic structure of our code inherited from the choice of its divisor. In fact, notice that $\mathcal{C}$ is monomially equivalent to $C_{\mathcal{L}}(\mathcal{X}, \mathcal{P}, D)$: more precisely, if $C_{\mathcal{L}}(\mathcal{X}, \mathcal{P}, D)$ is generated by $\mathbf{M}$, then a generator matrix of $\mathcal{C}$ is obtained by multiplying $\mathbf{M}$ by the diagonal matrix whose coefficients are $g^{-1}(P)$, for $P \in \mathcal{P}$. A direct consequence of this is the following lemma.

Lemma 4.8. *Suppose $s = \deg(D) \geq \mathfrak{g}$. Then for all $i \geq 0$, we have*

$$\dim_{\mathbb{F}_q} \mathrm{Tr}\Big(\mathcal{C} \star \mathcal{C}^{q^i}\Big) \leq m\left(s\left(q^i+1\right)+1-\mathfrak{g}\right).$$

Proof. From Lemma 4.3, we deduce that

$$\dim_{\mathbb{F}_{q^m}} \mathcal{C} \star \mathcal{C}^{q^i} \leq \dim_{\mathbb{F}_{q^m}} \mathcal{L}((q^i+1)G) = \dim_{\mathbb{F}_{q^m}} \mathcal{L}((q^i+1)D) = s(q^i+1)+1-\mathfrak{g},$$

the last equality coming from the Riemann–Roch theorem (since $\deg(q^i+1)D = (q^i+1)s \geq 2\mathfrak{g}-1$). The result follows from the usual upper bound on the dimension of the trace of a code. □

Remark 4.9. At first glance, it seems that we could have benefited from Theorem 1.85 to get a sharper bound in the previous lemma. Indeed, for every $i \geq 1$, we have

$$\left[\frac{(q^i+1)G}{q}\right] = q^{i-1}G^+ - (q^i+1)G^-,$$

writing $G = G^+ - G^-$ with G^+, $G^- \geq 0$. However, in the context of Goppa–like codes, we have $G = D + (g)$ where $g \notin \mathcal{L}(D)$, hence $G^- \neq 0$. Without further hypotheses on the divisor D and the function g, the degree of the divisor $\left[\frac{(q^i+1)G}{q}\right]$ may be too low to bound the dimension of its Riemmann–Roch space from below via the Riemann–Roch theorem.

This simple lemma yields an upper bound on the dimension of the square of the dual of Goppa–like codes.

Proposition 4.10. *Let $\mathcal{C} := \mathcal{C}_{\mathcal{L}}(\mathcal{X}, \mathcal{P}, D+(g))$ be an AG code as above and assume $s \geq \mathfrak{g}$. Set $k := \dim_{\mathbb{F}_{q^m}} \mathcal{C}$ and $e := \min\left(\left\lfloor \frac{m}{2} \right\rfloor, \left\lfloor \log_q\left(\frac{k^2}{s}\right)\right\rfloor\right)$. Then*

$$\dim_{\mathbb{F}_q}(\Gamma(\mathcal{P}, D, g)^{\perp})^{\star 2} \leq \binom{mk+1}{2} - \frac{m}{2}\left(k(k-1)(2e+1) - 2s\left(\frac{q^{e+1}-1}{q-1}\right)\right).$$

Proof. For any $e_0 \in \left\{0, \ldots, \left\lfloor \frac{m}{2} \right\rfloor\right\}$, Equation (4.3) above implies

$$\begin{aligned}
\dim_{\mathbb{F}_q}(\Gamma(\mathcal{P}, D, g)^{\perp})^{\star 2} &\leq \sum_{i=0}^{\lfloor m/2 \rfloor} \dim_{\mathbb{F}_q} \mathrm{Tr}\Big(\mathcal{C} \star \mathcal{C}^{q^i}\Big) \\
&\leq \sum_{i=0}^{e_0} m(s(q^i+1)+1-\mathfrak{g}) + \sum_{i=e_0+1}^{\lfloor m/2 \rfloor} \mathrm{Tr}\Big(\mathcal{C} \star \mathcal{C}^{q^i}\Big) \qquad \text{(by Lemma 4.8)} \\
&\leq \sum_{i=0}^{e_0} m(sq^i+k) + \left(\frac{m-1}{2} - e_0\right) mk^2 \qquad \text{(by the Riemann–Roch theorem)} \\
&\leq \frac{m}{2}\left(2k(e_0+1) + 2s\left(\frac{q^{e_0+1}-1}{q-1}\right) + k^2(m-1) - 2e_0k^2\right) \\
&\leq \binom{mk+1}{2} - \frac{m}{2}\left(k(k-1)(2e_0+1) - 2s\left(\frac{q^{e_0+1}-1}{q-1}\right)\right).
\end{aligned}$$

Notice that at the third line, we can replace $\lfloor \frac{m}{2} \rfloor$ with $\frac{m-1}{2}$ while bounding the second part of the sum thanks to the even case in Proposition 1.19.
To get the best bound, we maximize the expression

$$\frac{m}{2}\left(k(k-1)(2e_0+1) - 2s\left(\frac{q^{e_0+1}-1}{q-1}\right)\right)$$

with respect to e_0. Removing the constant parts, this is equivalent to find the maximum of the function

$$F(e_0) = e_0k^2 - s\frac{q^{e_0+1}}{q-1}$$

over $\left\{0, \dots, \lfloor \frac{m}{2} \rfloor\right\}$ in the discrete domain of non–negative integers. We compute the discrete derivative:

$$\begin{aligned}\Delta F(e_0) = F(e_0+1) - F(e_0) &= (e_0+1)k^2 - s\frac{q^{e_0+2}}{q-1} - \left(e_0 k^2 - s\frac{q^{e_0+1}}{q-1}\right) \\ &= k^2 - sq^{e_0+1}.\end{aligned}$$

This function is decreasing with e_0, and the smallest value for which $\Delta F(e_0) \leq 0$ corresponds to its maximum. It is the smallest value of e_0 such that $k^2 \leq sq^{e_0+1}$, *i.e.*

$$e_0 = e := \left\lfloor \log_q\left(\frac{k^2}{s}\right)\right\rfloor.$$

□

4.2.3 Sharpness of the bound

Definition 4.5 of a Goppa–like AG code $\Gamma(\mathcal{P}, D, g) := \mathcal{C}^{\perp}|_{\mathbb{F}_q}$ requires very few hypotheses. Besides the conditions on the supports of D and (g), which guarantee that the code is well–defined, we only ask for the function g not to belong to the Riemann–Roch space $\mathcal{L}(D)$. This hypothesis is enforced to make sure that the dimension of $\mathrm{Tr}(\mathcal{C})^{\star 2}$ is not abnormally small compared to the expected value given in Corollary 1.20, and thus to make Goppa–like AG codes resistant to a distinguisher based on the square of their dual. Let us discuss a bit more this assumption on g:

First, if the function g lied in $\mathcal{L}(D)$ (or more generally if the vector of the evaluations $(g(P))_{P\in\mathcal{P}}$ belonged to $C_{\mathcal{L}}(\mathcal{X}, \mathcal{P}, D)$), then the code $\mathcal{C} = \mathcal{C}_{\mathcal{L}}(\mathcal{X}, \mathcal{P}, D + (g))$ would contain the evaluation of the constant function $1 = \frac{g}{g}$, *i.e.* the unit vector $(1, \dots, 1)$. In this case, the vector $(1, \dots, 1)$ would belong to $\mathcal{C}^{q^i}$ for every $i \in \{0, \dots, \lfloor m/2 \rfloor\}$ and each term in the sum on the right–hand side would contain a copy of $\mathrm{Tr}(\mathcal{C})$. This non–trivial intersection between the codes $\mathrm{Tr}\left(\mathcal{C} \star \mathcal{C}^{q^i}\right)$ would contribute with a negative term in the above bound.

Secondly, if g belonged to $\mathcal{L}(D)$, then $D + (g)$ would be effective. This would imply the inclusion $\mathcal{L}((q^i+1)D) \subset \mathcal{L}((q^{i+1}+1)D)$ for every $i \geq 0$. Therefore, in the proof of Proposition 4.10, when bounding from above the dimension of the sum by the sum of the dimensions of the trace codes, we would have no chance to get a sharp bound.

Unfortunately, the condition $g \notin \mathcal{L}(D)$ does not guarantee that the bound given in Proposition 4.10 is reached. In the following proposition, we detail one situation in which we cannot hope for equality.

Proposition 4.11. *Using the same notation as above, set* $\mathcal{C}_1 = C_{\mathcal{L}}\left(\mathcal{X}, \mathcal{P}, \left[\frac{D+(g)}{q}\right]\right)$ *(see Equation* (1.11) *for the definition of* $\left[\frac{G}{q}\right]$*, given* G*). If* $\dim \mathcal{C}_1 \geq 1$*, then the bound given in Proposition 4.10 is not reached.*

Proof. Any non–zero codeword $\mathbf{c} \in \mathcal{C}_1 \subset \mathcal{C}$ satisfies $\mathbf{c}^q \in \mathcal{C}$. As $\mathbf{c}$ lies in $\mathbb{F}_{q^m}^n$, we have $\mathbf{c}^{q^m} = \mathbf{c} \in \mathcal{C}^{q^{m-1}}$. Therefore, we have $\mathcal{C}_1 \subseteq \mathcal{C} \cap \mathcal{C}^{q^{m-1}}$, and for every $i \in \left\{1, \dots, \lfloor \frac{m}{2} \rfloor\right\}$, we have $\mathcal{C}_1^{q^i} \subseteq \mathcal{C}^{q^i} \cap \mathcal{C}^{q^{i-1}}$. Then

$$\mathrm{Tr}\left(\mathcal{C} \star \mathcal{C}_1^{q^i}\right) \subseteq \mathrm{Tr}\left(\mathcal{C} \star \mathcal{C}^{q^i}\right) \cap \mathrm{Tr}\left(\mathcal{C} \star \mathcal{C}^{q^{i-1}}\right).$$

As a result, each pair of consecutive terms in the sum $\sum_{i=0}^{\lfloor m/2 \rfloor} \mathrm{Tr}\left(\mathcal{C} \star \mathcal{C}^{q^i}\right)$ has a non–trivial intersection. However, using [Tia19, Theorem 2], equality with the upper bound only occurs if

$$\bigcap_{0 \leq j \leq \lfloor m/2 \rfloor} \left(\sum_{\substack{0 \leq i \leq \lfloor m/2 \rfloor \\ i \neq j}} \mathrm{Tr}\left(\mathcal{C} \star \mathcal{C}^{q^i}\right) \right) = \{0\}.$$

□

Remark 4.12. As noted in Remark 4.9, when picking the function g at random outside $\mathcal{L}(D)$, the code $\mathcal{C}_1$ is likely to be reduced to zero.

As recalled in Section 4.2.1, Goppa–like AG codes coincide with Cartier code as soon as

$$\deg\left(\left[\frac{D+(g)}{q}\right]\right) > 2\mathfrak{g}-2.$$

In this case, the code $\mathcal{C}_1$ has dimension at least $\mathfrak{g}$. This means that when the Goppa–like code is also a Cartier code, the dimension of the square of its dual is very unlikely to meet the bound given in Proposition 4.10.

Remark 4.13. When considering classical Goppa codes (see. Definition 1.91) for the McEliece cryptosystem, it is common to ask for the degree r Goppa polynomial g to have only simple roots. In this case, we have $\left[\frac{(r-1)P_\infty+(g)}{q}\right] = -P_\infty$ and the code $\mathcal{C}_1$ defined in Proposition 4.11 is always zero. The situation above thus never occurs.

4.3 One–point Goppa–like AG code on $C_{a,b}$–curves

The bound given in section 4.2.2 can be improved even more by considering more structured codes, *i.e.* one–point Goppa–like AG codes on $C_{a,b}$ curves.

4.3.1 The point at infinity and weighted degree

Let a, b be coprime positive integers and fix a $C_{a,b}$ curve $\mathcal{X}_{a,b}$. As defined in Section 1.2.5, this means that $\mathcal{X}_{a,b}$ have an irreducible, affine and non–singular plane model with equation

$$f_{a,b}(x,y) = \alpha_{0a}y^a + \alpha_{b0}x^b + \sum \alpha_{ij}x^iy^j = 0, \tag{4.4}$$

where $f_{a,b} \in \mathbb{F}_{q^m}[X,Y]$ with $\alpha_{0a}, \alpha_{b0} \neq 0$, and the sum is taken over all couples $(i,j) \in \{0,\cdots,b\} \times \{0,\cdots,a\}$ such that $ai+bj < ab$. Among notable properties, recall that $\mathcal{X}_{a,b}$ has a unique point at infinity. From now on, this point is denoted by P_∞, and we write $\mathfrak{g}(\mathcal{X}_{a,b}) := \mathfrak{g}_{a,b} = \frac{(a-1)(b-1)}{2}$ the genus of $\mathcal{X}_{a,b}$.

We will consider codes obtained by evaluating functions on $\mathcal{X}_{a,b}$ which are regular everywhere, except maybe at P_∞.
These functions then belong to the coordinate ring of the affine curve $\mathcal{X}_{a,b} \setminus \{P_\infty\}$, which we denote by $\mathcal{S}$, *i.e*

$$\mathcal{S} = \bigcup_{s\geq 0} \mathcal{L}(sP_\infty), \tag{4.5}$$

where each Riemann–Roch space $\mathcal{L}(sP_\infty)$ has an explicit basis as follows:

$$\mathcal{L}(sP_\infty) = \text{Span}\left(x^iy^j \mid 0 \leq i, 0 \leq j \leq a-1 \text{ and } ai+bj \leq s\right). \tag{4.6}$$

In summary, any function that is regular on all $\mathcal{X}_{a,b}$ except maybe at P_∞ can be seen as a bivariate polynomial in the functions x and y.

Definition 4.14 (Weighted degree)**.** Given a monomial of the form $x^iy^j \in \mathcal{S}$, we define its *weighted degree* by

$$\deg_{a,b}\left(x^iy^j\right) := ai+bj.$$

From this degree, we can define a monomial order $\prec$ over $\mathcal{S} \simeq \mathbb{F}_{q^m}[x,y]$ as follows: given two monomial x^uy^v and $x^{u'}y^{v'}$, we say that

$$x^uy^v \prec x^{u'}y^{v'}$$

if

$$\deg_{a,b}\left(x^uy^v\right) < \deg_{a,b}\left(x^{u'}y^{v'}\right) \text{ or } \left(\deg_{a,b}\left(x^uy^v\right) = \deg_{a,b}\left(x^{u'}y^{v'}\right) \text{ and } u < u'\right). \tag{4.7}$$

From the basis given in (4.6), any function $f \in \mathcal{S}$ can be written in the form

$$f(x,y) = c \cdot x^\beta y^\alpha + f'(x,y),$$

with $c \neq 0$, $\alpha \leq a-1$ and $f' \in \mathcal{S}$ such that any monomial x^iy^j of f' satisfies $ai+bj < \deg_{a,b}\left(x^\beta y^\alpha\right)$ and $j \leq a-1$. The *leading monomial* of f with respect to the monomial order $\prec$ is thus defined by $\text{LM}(f) := x^\beta y^\alpha$. This extends the definition of weighted degree to any such function by setting

$$\deg_{a,b}(f) := \deg_{a,b}(\text{LM}(f)).$$

It is easy to check that for any $f \in \mathcal{S}$, its weighted degree $\deg_{a,b}(f)$ is equal to the biggest integer s such that f belongs to the Riemann–Roch space $\mathcal{L}(sP_\infty)$. This way, any function in $\mathcal{L}(sP_\infty)$ can be seen as a polynomial in x and y such that $\deg_{a,b}(f) \leq s$.

Remark 4.15. For every $f \in \mathcal{S}$, we have $\deg_{a,b}(f) = -v_{P_\infty}(f)$.

4.3.2 The codes

For the rest of this section, fix a $C_{a,b}$ curve $\mathcal{X}_{a,b}$ over $\mathbb{F}_{q^m}$, whose defining equation is given by Equation (4.4). We now define a subclass of Goppa–like codes, associated with the one–point divisor sP_∞.

Definition 4.16. Let $s' > s$ be two integers such that there exists a function $g \in \mathcal{L}(s'P_\infty)$ with $\deg_{a,b}(g) = s'$. Given a set of rational points $\mathcal{P} \subset \mathcal{X}_{a,b}(\mathbb{F}_{q^m})$ such that $\mathcal{P} \cap \mathrm{Supp}(g) = \varnothing$, we define the *one–point Goppa–like* AG code associated to $\mathcal{P}, s$ and g as

$$\Gamma(\mathcal{P}, sP_\infty, g) := \mathcal{C}_{\mathcal{L}}(\mathcal{X}_{a,b}, \mathcal{P}, sP_\infty + (g))^\perp|_{\mathbb{F}_q}.$$

This definition might sound limiting, since we restrict ourselves to specific one–point divisor. This is motivated by the fact that these codes can be encoded quickly thanks to the nice basis of $\mathcal{L}(sP_\infty)$ (see [BESP10]), which is desirable if we aim to build a McEliece cryptosystem based on it. Moreover, this property will be key in the upcoming sections as it allows a better understanding of the square of the dual, under some additional conditions on s and s'.

In the next two sections, we generalize the result given in [MT21] in the case of classical Goppa codes, by defining a weighted Euclidean division on the ring $\mathcal{S}$ (see Equation (4.5)), whose elements are seen as bivariate polynomials.

4.3.3 Weighted Euclidean division

The following proposition generalizes the classical Euclidean division of univariate polynomials in the case of function in $\mathcal{S}$ with respect to the weighted degree $\deg_{a,b}$ (see Definition 4.14). Before that, we need the following definition:

Definition 4.17. For any function $h \in \mathcal{S}$ with leading monomial $\mathrm{LM}(h) = x^\beta y^\alpha$ and $\alpha < a$, we define over $\mathbb{F}_{q^m}$ the space

$$\mathcal{R}(h) := \mathrm{Span}\left(x^u y^v \mid u \leq \beta + b - 1 \text{ and } v \leq a - 1 \text{ not both } u \geq \beta \text{ and } v \geq \alpha\right).$$

Note that the dimension of $\mathcal{R}(h)$ is equal to $\deg_{a,b}(h)$.

Proposition 4.18. *Fix any nonzero function $g \in \mathcal{S}$. Then for any function $f \in \mathcal{S}$, there exist $f_1, f_2 \in \mathcal{S}$ such that*

$$f = f_1 g + f_2 \text{ with } f_2 \in \mathcal{R}(g).$$

Moreover, we have $\deg_{a,b}(f_2) \leq \deg_{a,b}(f)$.

Proof. Since $f \in \mathcal{S}$, we can see f as a bivariate polynomial in x and y (see Equation 4.5). In the polynomial ring $\mathbb{F}_{q^m}[x, y]$, we perform the division of f by a Gröbner basis of the ideal generated by the equation $f_{a,b}$ of the curve $\mathcal{X}_{a,b}$ and the polynomial g with respect to the monomial order $\prec$ defined in Equation (4.7). The fact that f_2 lies in $\mathcal{R}(g)$ and the result on the dimension of $\mathcal{R}(g)$ both follow from [GH00, Proposition 4].

Finally, if we had $\deg_{a,b}(f) < \deg_{a,b}(f_2)$ with $f = f_1 g + f_2$, this would mean that

$$\mathrm{LM}(f_2) = -\mathrm{LM}(f_1 g) = \lambda x^u y^v,$$

for some $\lambda \in \mathbb{F}_{q^m}^*$ with both $u \geq \beta$ and $v \geq \alpha$, which is not possible by definition of $\mathcal{R}(g)$. □

We will now use the weighted Euclidean division defined above to better control the elements in $\mathrm{Tr}\left(\mathcal{C} \star \mathcal{C}^{q^i}\right)$ for $\mathcal{C} = \mathcal{C}_{\mathcal{L}}(\mathcal{X}_{a,b}, \mathcal{P}, sP_\infty + (g))$, where we fixed $s' > s \geq 0$, $g \in \mathcal{S}$ with $\deg_{a,b}(g) = s'$ and a set of points $\mathcal{P} \subset \mathcal{X}_{a,b}(\mathbb{F}_{q^m})$ such that $\mathcal{P} \cap \mathrm{Supp}(g) = \varnothing$.

Before diving into technical proofs, let us fix some notation that we will use in the rest of the paper. We extend the evaluation map defined in Equation (1.7) on all the ring $\mathcal{S}$:

$$\mathrm{ev}_{\mathcal{P}} : \left\{ \begin{array}{ccc} \mathcal{S} & \to & \mathbb{F}_{q^m}^n \\ f & \mapsto & (f(P_1), \ldots, f(P_n)) \end{array} \right.$$

Now, $\mathrm{ev}_{\mathcal{P}}$ is an algebra homomorphism, where $\mathbb{F}_q^n$ is endowed with the product $\star$, *i.e.* $\mathrm{ev}_{\mathcal{P}}(f) \star \mathrm{ev}_{\mathcal{P}}(f') = \mathrm{ev}_{\mathcal{P}}(f \cdot f')$ for any $f,\ f' \in \mathcal{S}$.

For two functions $f,\ f' \in \mathcal{S}$, we will write $f \equiv_{\mathcal{P}} f'$ if $\mathrm{ev}_{\mathcal{P}}(f) = \mathrm{ev}_{\mathcal{P}}(f')$.

We also extend the trace operator on $\mathcal{S}$ by defining $\mathrm{Tr}(f) = f + f^q + \cdots + f^{q^{m-1}}$ for every $f \in \mathcal{S}$. Since the map $\mathrm{ev}_{\mathcal{P}}$ is an algebra homomorphism, we have $\mathrm{ev}_{\mathcal{P}}(\mathrm{Tr}(f)) = \mathrm{Tr}(\mathrm{ev}_{\mathcal{P}}(f))$. Moreover, for any function $f \in \mathcal{S}$, we have

$$\mathrm{Tr}(f^q) \equiv_{\mathcal{P}} \mathrm{Tr}(f). \tag{4.8}$$

Lemma 4.19. *Take $i \geq 1$. Let $f \in \mathcal{S}$ such that $\deg_{a,b}(f) < s'(q^i+1)$. Then there exists $f' \in \mathcal{R}\left(g^{q^i-q^{i-1}+1}\right)$ such that the vectors $\mathrm{Tr}\left(\frac{f}{g^{q^i+1}}\right) \equiv_{\mathcal{P}} \mathrm{Tr}\left(\frac{f'}{g^{q^i+1}}\right)$.*

Proof. By Proposition 4.18, we can write $f = f_1 g^{q^i-q^{i-1}+1} + f_2$ with $f_2 \in \mathcal{R}\left(g^{q^i-q^{i-1}+1}\right)$ and $\deg_{a,b}(f_2) \leq \deg_{a,b}(f)$. Therefore, using Equation (4.8), we get

$$\begin{aligned}\mathrm{Tr}\left(\frac{f}{g^{q^i+1}}\right) &= \mathrm{Tr}\left(\frac{f_1 g^{q^i-q^{i-1}+1}}{g^{q^i+1}}\right) + \mathrm{Tr}\left(\frac{f_2}{g^{q^i+1}}\right)\\ &\equiv_{\mathcal{P}} \mathrm{Tr}\left(\frac{f_1^q g}{g^{q^i+1}}\right) + \mathrm{Tr}\left(\frac{f_2}{g^{q^i+1}}\right).\end{aligned}$$

By definition, the second term has the expected form. Let us examine the first term. If $f_1 = 0$, we are done. Otherwise, the definition of f_1 gives $\deg_{a,b}(f_1) = \deg_{a,b}(f) - s'(q^i - q^{i-1} + 1)$, and

$$\begin{aligned}\deg_{a,b}(f_1^q g) &= q\deg_{a,b}(f_1) + s'\\ &= q\deg_{a,b}(f) - s'(q-1)(q^i+1).\end{aligned}$$

Then $\deg_{a,b}(f_1^q g) < \deg_{a,b}(f)$ if and only if $\deg_{a,b}(f) < s'(q^i+1)$, which holds by definition of f. Performing a new division by replacing f with $f_1^q g$ gives a new decomposition $f_1^q g = f_1' g^{q^i-q^{i-1}+1} + f_2'$, with $f_2' \in \mathcal{R}\left(g^{q^i-q^{i-1}+1}\right)$ and $\deg_{a,b}(f_2') \leq \deg_{a,b}(f)$. In particular, we can decompose $\mathrm{Tr}\left(\frac{f_1^q g}{g^{q^i+1}}\right)$ into a sum of traces as we did for $\mathrm{Tr}\left(\frac{f}{g^{q^i+1}}\right)$. If $f_1' = 0$, the result is proved. Otherwise, we can repeat another time the division process. As the weighted degree of the successive quotients decrease, we ultimately end up with a quotient equal to zero, which proves the result. □

Definition 4.20. For any $1 \leq i \leq \lfloor \frac{m}{2} \rfloor$, we define

$$\mathcal{T}_i(s,g) := \left\{ \mathrm{ev}_{\mathcal{P}}\left(\mathrm{Tr}\left(\frac{f}{g^{q^i+1}}\right)\right) \mid f \in \mathcal{R}\left(g^{q^i-q^{i-1}+1}\right) \cap \mathcal{L}(s(q^i+1)P_\infty) \right\}$$

and we set

$$\mathcal{T}_0(s,g) := \left\{ \mathrm{ev}_{\mathcal{P}}\left(\mathrm{Tr}\left(\frac{f}{g^2}\right)\right) \mid f \in \mathcal{L}(2sP_\infty) \right\}.$$

The vector spaces $\mathcal{T}_i(s,g)$ have been designed so that we have

$$\mathrm{Tr}\left(\mathcal{C} \star \mathcal{C}^{q^i}\right) \subseteq \mathcal{T}_i(s,g) \tag{4.9}$$

for all $i \in \left\{0, \ldots, \lfloor \frac{m}{2} \rfloor\right\}$. Indeed, it is straightforward for $i = 0$ and it follows from Lemma 4.19 for $i \geq 1$, noticing that $f \in \mathcal{L}(sP_\infty) \cdot \mathcal{L}(sP_\infty)^{q^i} \subseteq \mathcal{L}(s(q^i+1)P_\infty)$.

We will benefit from these inclusions to improve the bound given in Proposition 4.10, provided that we can efficiently compute the dimension of the trace codes $\mathcal{T}_i(s,g)$. This is studied in the next section.

4.3.4 Upper bound in Goppa–like case

In the proposition below, we study the intersection

$$M_i(s,g) := \mathcal{R}\left(g^{q^i-q^{i-1}+1}\right) \cap \mathcal{L}(s(q^i+1)P_\infty) \tag{4.10}$$

for every $i \in \{1, \ldots, \lfloor m/2 \rfloor\}$, in order to better grasp the trace codes $\mathcal{T}_i(s,g)$'s introduced in Definition 4.20. First we set some notation: fix $i \in \{1, \ldots, \lfloor m/2 \rfloor\}$ and write $\mathrm{LM}(g) = x^\beta y^\alpha$ with $a\beta + b\alpha = s'$. By reducing modulo the equation $f_{a,b}$ of $\mathcal{X}_{a,b}$, we can write the function $g^{q^i-q^{i-1}+1}$ such that its leading monomial with respect to the monomial order $\prec$ is

$$\mathrm{LM}\left(g^{q^i-q^{i-1}+1}\right) = x^{\beta_i} y^{\alpha_i} \tag{4.11}$$

where $\alpha_i \in \{0, \ldots, a-1\}$ is the remainder of the Euclidean division of $\alpha(q^i - q^{i-1} + 1)$ by a and

$$\beta_i = \beta(q^i - q^{i-1} + 1) + b\,\frac{\alpha(q^i - q^{i-1} + 1) - \alpha_i}{a} = \frac{s'(q^i - q^{i-1} + 1) - b\alpha_i}{a}. \tag{4.12}$$

Depending on the weighted degree s' of g, we can compute the exact dimension of the vector space $M_i(s,g)$ defined in Equation (4.10).

Proposition 4.21. *1. If $s'(q^i - q^{i-1} + 1) > s(q^i + 1) + a$, then $M_i(s,g) = \mathcal{L}(s(q^i+1)P_\infty)$;*

2. If $s'(q^i - q^{i-1} + 1) \leq s(q^i + 1) + 1 - 2\mathfrak{g}_{a,b}$, then $M_i(s,g) = \mathcal{R}(g^{q^i - q^{i-1}+1})$;

3. If there exists $v^* \in \{1, \ldots, \alpha_i - 1\}$ such that

$$s(q^i+1) + a - b(a + v^* - \alpha_i) < s'(q^i - q^{i-1} + 1) \leq s(q^i+1) + a - b(a + v^* - 1 - \alpha_i),$$

we have

$$\dim_{\mathbb{F}_{q^m}}(M_i(s,g)) = \sum_{v=v^*}^{a-1} \left\lfloor \frac{s(q^i+1) - bv}{a} \right\rfloor + v^*(\beta_i + b) + a - v^*.$$

4. Otherwise, there exists $v^* \in \{\alpha_i + 1, \ldots, a\}$ such that

$$s(q^i+1) + a - b(v^* - \alpha_i) < s'(q^i - q^{i-1} + 1) \leq s(q^i+1) + a - b(v^* - 1 - \alpha_i),$$

in which case

$$\dim_{\mathbb{F}_{q^m}}(M_i(s,g)) = \sum_{v=v^*}^{a-1} \left\lfloor \frac{s(q^i+1) - bv}{a} \right\rfloor + v^*\beta_i + \alpha_i b + a - v^*.$$

Proof. Using the notation above, we can write

$$\begin{aligned}\mathcal{R}\left(g^{q^i - q^{i-1}+1}\right) &:= \mathrm{Span}_{\mathbb{F}_{q^m}}\{x^u y^v \mid u \leq \beta_i + b - 1, v \leq a-1 \text{ not both } u \geq \beta_i \text{ and } v \geq \alpha_i\} \\ &= \mathrm{Span}\begin{pmatrix} 1, x, \ldots, x^{\beta_i+b-1}, \\ \cdots \\ y^{\alpha_i-1}, y^{\alpha_i-1}x, \ldots, y^{\alpha_i-1}x^{\beta_i+b-1}, \\ y^{\alpha_i}, y^{\alpha_i}x, \ldots, y^{\alpha_i}x^{\beta_i-1}, \\ \cdots \\ y^{a-1}, y^{a-1}x, \ldots, y^{a-1}x^{\beta_i-1} \end{pmatrix}\end{aligned}$$

Next, we define for any $v \in \{0, \ldots, a-1\}$:

$$\ell_v^i := \max\left\{u \geq 0 \mid x^u y^v \in \mathcal{L}(s(q^i+1)P_\infty)\right\} = \left\lfloor \frac{s(q^i+1) - bv}{a} \right\rfloor,$$

implying

$$\mathcal{L}(s(q^i+1)P_\infty) = \mathrm{Span}\begin{pmatrix} 1, x, \ldots, x^{\ell_0^i}, \\ y, yx, \ldots, yx^{\ell_1^i}, \\ \cdots \\ y^{a-1}, y^{a-1}x, \ldots, y^{a-1}x^{\ell_{a-1}^i} \end{pmatrix}.$$

We thus have a description of a basis of both spaces $\mathcal{R}(g^{q^i - q^{i-1}+1})$ and $\mathcal{L}(s(q^i+1)P_\infty)$, leading to an exact formula for the dimension of their intersection $M_i(s,g)$ for any value of i:

$$\dim_{\mathbb{F}_{q^m}} M_i(s,g) = \sum_{v=0}^{\alpha_i - 1} \min(\beta_i + b, \ell_v^i + 1) + \sum_{v=\alpha_i}^{a-1} \min(\beta_i, \ell_v^i + 1). \tag{4.13}$$

It remains to compute the corresponding minima with respect to v:

(i) If $0 \leq v \leq \alpha_i - 1$, by using (4.12), we get

$$\beta_i + b \leq \ell_v^i + 1 \iff s'(q^i - q^{i-1} + 1) \leq s(q^i+1) + a - b(a + v - \alpha_i) := F(v).$$

(ii) Otherwise, $\alpha_i \leq v \leq a - 1$ and

$$\beta_i \leq \ell_v^i + 1 \iff s'(q^i - q^{i-1} + 1) \leq s(q^i+1) + a - b(v - \alpha_i) := G(v).$$

Note that both F and G are decreasing with v, and we easily check that $F(0) = G(a)$. Thus, we have the following sequence of integers

$$F(\alpha_i - 1) \leq \cdots \leq F(0) = G(a) \leq G(a-1) \leq \cdots \leq G(\alpha_i).$$

Depending on the value of s', there are a few cases to consider:

- $s'(q^i - q^{i-1} + 1) > G(\alpha_i)$, in which case $M_i(s,g) = \mathcal{L}(s(q^i+1)P_\infty)$;
- $s'(q^i - q^{i-1} + 1) \leq F(\alpha_i - 1)$, and $M_i(s,g) = \mathcal{R}(g^{q^i - q^{i-1}+1})$;
- There exists $v^* \in \{1, ..., \alpha_i - 1\}$ such that $F(v^*) < s'(q^i - q^{i-1} + 1) \leq F(v^* - 1)$;
- There exists $v^* \in \{\alpha_i, ..., a\}$ such that $G(v^*) < s'(q^i - q^{i-1} + 1) \leq G(v^* - 1)$.

The formulas for the dimension of $M_i(s,g)$ follows from the above computations and (4.13). □

Note that item *1.* corresponds to the case where $\mathcal{T}_i(s,g) = \mathrm{Tr}\Big(g^{-(q^i+1)} \cdot \mathcal{L}(s(q^i+1)P_\infty)\Big)$, which would produce the same bound as the one given in Proposition 4.10. Instead, we focus on item *2.*, since in this case, we can show some inclusion relations between the $\mathcal{T}_i(s,g)$'s.

Proposition 4.22. *Keep above notation and let $i^* \in \big\{0, \ldots, \lfloor \frac{m}{2} \rfloor - 1\big\}$ be the smallest integer such that*

$$sq^{i^*} \geq (s'-s)(q^{i^*+1} - q^{i^*} + 1) + 2\mathfrak{g}_{a,b} - 1. \tag{4.14}$$

Then

$$\mathcal{T}_{i^*}(s,g) \subseteq \mathcal{T}_{i^*+1}(s,g) \subseteq \cdots \subseteq \mathcal{T}_{\lfloor \frac{m}{2} \rfloor}(s,g).$$

Proof. From Proposition 4.21 (2), we know that (4.14) implies

$$M_{i^*+1}(s,g) = \mathcal{R}(g^{q^{i^*+1} - q^{i^*+1-1}+1}).$$

We can easily check that the function

$$i \mapsto \frac{s(q^i+1) + 1 - 2\mathfrak{g}_{a,b}}{q^i - q^{i-1} + 1}$$

is increasing with i, hence we also have

$$M_i(s,g) = \mathcal{R}(g^{q^i - q^{i-1}+1}),\ \forall\ i \in \left\{i^*, \ldots, \left\lfloor \frac{m}{2} \right\rfloor + 1\right\}. \tag{4.15}$$

We now prove the inclusions between the $\mathcal{T}_i's$, assuming first that $i^* \neq 0$ (since the definition of $\mathcal{T}_0$ is a bit different). Let $i \in \big\{i^*, \ldots, \lfloor \frac{m}{2} \rfloor - 1\big\}$, and recall that

$$\mathcal{T}_i(s,g) := \left\{ \mathrm{ev}_{\mathcal{P}} \left(\mathrm{Tr}\Big(\frac{f}{g^{q^i+1}} \Big) \right) \mid f \in \mathcal{R}\left(g^{q^i - q^{i-1}+1}\right) \cap \mathcal{L}(s(q^i+1)P_\infty) \right\}$$

Given an element $\mathrm{ev}_{\mathcal{P}}\left(\mathrm{Tr}\Big(\frac{f}{g^{q^i+1}}\Big)\right)$ in $\mathcal{T}_i(s,g)$, we want to show that it belongs to $\mathcal{T}_{i+1}(s,g)$. Applying Proposition 4.18 by replacing f with $fg^{q^{i+1}-q^i}$ and g by $g^{q^{i+1}-q^i+1}$, we obtain

$$fg^{q^{i+1}-q^i} = f_1 g^{q^{i+1}-q^i+1} + f_2, \tag{4.16}$$

with $f_2 \in \mathcal{R}(g^{q^{i+1}-q^i+1}) = M_i(s,g)$ (using (4.15)) and $\deg_{a,b}(f_2) \leq \deg_{a,b}\left(fg^{q^{i+1}-q^i}\right)$. Next, we write

$$\begin{aligned}
\mathrm{Tr}\left(\frac{f}{g^{q^i+1}}\right) &= \mathrm{Tr}\left(\frac{fg^{q^{i+1}-q^i}}{g^{q^{i+1}+1}}\right) \\
&= \mathrm{Tr}\left(\frac{f_1 g^{q^{i+1}-q^i+1}}{g^{q^{i+1}+1}}\right) + \mathrm{Tr}\left(\frac{f_2}{g^{q^{i+1}+1}}\right) \\
&\equiv_{\mathcal{P}} \mathrm{Tr}\left(\frac{f_1^q g}{g^{q^{i+1}+1}}\right) + \mathrm{Tr}\left(\frac{f_2}{g^{q^{i+1}+1}}\right).
\end{aligned}$$

By assumption on f_2, we immediately have that $\mathrm{ev}_{\mathcal{P}}\left(\mathrm{Tr}\Big(\frac{f_2}{g^{q^{i+1}+1}}\Big)\right) \in \mathcal{T}_{i+1}(s,g)$. If $f_1 = 0$, we are done. Otherwise, we have from (4.16):

$$\deg_{a,b}(f_1) = \deg_{a,b}\left(fg^{q^{i+1}-q^i}\right) - \deg_{a,b}\left(g^{q^{i+1}-q^i+1}\right) = \deg_{a,b}(f) - s'.$$

Thus

$$\begin{aligned}
\deg_{a,b}(f_1^q g) < \deg_{a,b}\left(fg^{q^{i+1}-q^i}\right) &\iff q\deg_{a,b}(f) + (1-q)s' < \deg_{a,b}(f) + s'(q^{i+1} - q^i) \\
&\iff \deg_{a,b}(f) < s'(q^i+1),
\end{aligned}$$

which is true since in particular $f \in \mathcal{L}(s(q^i+1)P_\infty)$ and $s < s'$. Since the weighted degree decreases, we can repeat the division process until eventually we obtain a quotient f_1 equal to zero (as in the proof of Lemma 4.19), which proves that $\mathcal{T}_i(s,g) \subseteq \mathcal{T}_{i+1}(s,g)$.

In the case $i^* = 0$, we also have to prove that $\mathcal{T}_0(s,g) \subseteq \mathcal{T}_1(s,g)$, which differs from the other cases due to the definition of $\mathcal{T}_0$. Let $\mathrm{ev}_{\mathcal{P}}\left(\mathrm{Tr}\left(\frac{f}{g^2}\right)\right) \in \mathcal{T}_0(s,g)$, for some $f \in \mathcal{L}(2sP_\infty)$. Using Proposition 4.18, this time replacing f with fg^{q-1} and g with g^{q+1} yields

$$fg^{q-1} = f_1 g^q + f_2,$$

with $f_2 \in \mathcal{R}(g^q) = M_1(s,g)$ (using (4.15) again). Thus, we can write

$$\mathrm{Tr}\left(\frac{f}{g^2}\right) = \mathrm{Tr}\left(\frac{f_1^q g}{g^{q+1}}\right) + \mathrm{Tr}\left(\frac{f_2}{g^{q+1}}\right),$$

with $\mathrm{ev}_{\mathcal{P}}\left(\mathrm{Tr}\left(\frac{f_2}{g^{q+1}}\right)\right) \in \mathcal{T}_1(s,g)$. Since $\deg_{a,b}(f_1) = \deg_{a,b}\left(fg^{q-1}\right) - \deg_{a,b}(g^q) = \deg_{a,b}(f) - s'$, we have

$$\begin{aligned}
\deg_{a,b}(f_1^q g) < \deg_{a,b}\left(fg^{q-1}\right) &\iff q\deg_{a,b}(f) + (1-q)s' < \deg_{a,b}(f) + s'(q-1) \\
&\iff (q-1)\deg_{a,b}(f) < 2s'(q-1) \\
&\iff \deg_{a,b}(f) < 2s',
\end{aligned}$$

which holds since $s < s'$ and $f \in \mathcal{L}(2sP_\infty)$. Repeating the division process until we find a quotient equal to zero shows that $\mathcal{T}_0(s,g) \subseteq \mathcal{T}_1(s,g)$. The other inclusions hold as in the case $i^* \geq 1$. □

Combining the inclusions (4.9) with both the above propositions lead to a better understanding of the dimension of the square of the dual of one–point Goppa–like AG codes, namely:

Corollary 4.23. *With notation of Proposition 4.22, set $k := \dim_{\mathbb{F}_{q^m}} C_{\mathcal{L}}(\mathcal{X}_{a,b}, \mathcal{P}, sP_\infty + (g))$. Then, for all $e \in \left\{0, \ldots, \lfloor \frac{m}{2} \rfloor\right\}$, the dimension of $\Gamma(\mathcal{P}, sP_\infty, g)^\perp)^{\star 2}$ is bounded from above by*

$$\dim_{\mathbb{F}_q}(\Gamma(\mathcal{P}, sP_\infty, g)^\perp)^{\star 2} \leq \left(\frac{m-1}{2} - e\right) mk^2 + \dim_{\mathbb{F}_q}\left(\sum_{i=0}^{e} \mathcal{T}_i(s,g)\right).$$

Moreover, if $i^* \leq e \leq \lfloor \frac{m}{2} \rfloor$ is the integer satisfying equation (4.14) , we have

$$\begin{aligned}
\dim_{\mathbb{F}_q}(\Gamma(\mathcal{P}, sP_\infty, g)^\perp)^{\star 2} \leq &\left(\frac{m-1}{2} - e\right) mk^2 + ms'(q^e - q^{e-1} + 1) \\
&+ \dim_{\mathbb{F}_q}\left(\sum_{i=0}^{i^*-1} \mathcal{T}_i(s,g)\right) - \dim_{\mathbb{F}_q}\left(\mathcal{T}_e(s,g) \cap \sum_{i=0}^{i^*-1} \mathcal{T}_i(s,g)\right).
\end{aligned}$$

Proof. Starting from Equation (4.3) and using Equation (4.9), we have

$$\begin{aligned}
\dim_{\mathbb{F}_q}(\Gamma(\mathcal{P}, sP_\infty, g)^\perp)^{\star 2} &\leq \sum_{i=0}^{\lfloor m/2 \rfloor} \dim_{\mathbb{F}_q} \mathrm{Tr}\left(\mathcal{C} \star \mathcal{C}^{q^i}\right) \\
&\leq \dim_{\mathbb{F}_q} \sum_{i=0}^{e} \mathcal{T}_i(s,g) + \sum_{i=e+1}^{\lfloor m/2 \rfloor} \mathrm{Tr}\left(\mathcal{C} \star \mathcal{C}^{q^i}\right) \\
&\leq \dim_{\mathbb{F}_q} \sum_{i=0}^{e} \mathcal{T}_i(s,g) + \left(\frac{m-1}{2} - e\right) mk^2,
\end{aligned}$$

for all $e \in \left\{0, \ldots, \lfloor \frac{m}{2} \rfloor\right\}$. If $i^* \leq e \leq \lfloor \frac{m}{2} \rfloor$, Proposition 4.22 gives

$$\sum_{i=i^*}^{e} \mathcal{T}_i(s,g) = \mathcal{T}_e(s,g).$$

Consequently, we can write

$$\dim_{\mathbb{F}_q}\left(\sum_{i=0}^{e} \mathcal{T}_i(s,g)\right) = \dim_{\mathbb{F}_q}\left(\sum_{i=0}^{i^*-1} \mathcal{T}_i(s,g)\right) + \dim_{\mathbb{F}_q} \mathcal{T}_e(s,g) - \dim_{\mathbb{F}_q}\left(\mathcal{T}_e(s,g) \cap \sum_{i=0}^{i^*-1} \mathcal{T}_i(s,g)\right).$$

To finish the proof, we use the fact that $M_e(s,g) = \mathcal{R}\left(g^{q^e-q^{e-1}+1}\right)$ (see. Proposition 4.21, 2), hence we have

$$\dim_{\mathbb{F}_q} \mathcal{T}_e(s,g) \leq m \cdot \dim_{\mathbb{F}_{q^m}} \mathcal{R}\left(g^{q^e-q^{e-1}+1}\right) = ms'(q^e - q^{e-1} + 1).$$

□

Despite the fact that the upper bound given in Corollary 4.23 can be numerically computed with the knowledge of the degree s and the function g, it is hard to give a close formula for any parameter, since the intersections of the trace codes $\mathcal{T}_i(s,g)$'s are hard to manipulate. However, if we assume that $i^* = 0$, we can sharpen the above result.

Theorem 4.24. *Suppose that* $s \geq (s'-s)q + 2\mathfrak{g}_{a,b} - 1$ *and let* $e^* := \min\left(\left\lfloor \frac{m}{2} \right\rfloor, \left\lceil \log_q\left(\frac{k^2}{s'(q-1)^2}\right)\right\rceil + 1\right)$. *Then*

$$\dim_{\mathbb{F}_q}(\Gamma(\mathcal{P}, sP_\infty, g)^\perp)^{\star 2} \leq \binom{mk+1}{2} - \frac{m}{2}(k^2(2e^*+1) + k - 2s'(q^{e^*} - q^{e^*-1} + 1)).$$

Proof. The condition $s \geq (s'-s)q + 2\mathfrak{g}_{a,b} - 1$ exactly implies that $i^* = 0$ and

$$\mathcal{T}_0(s,g) \subseteq \mathcal{T}_1(s,g) \subseteq \cdots \subseteq \mathcal{T}_{\lfloor \frac{m}{2} \rfloor}(s,g),$$

by Proposition 4.22. Thus, using Corollary 4.23 and the inequality

$$\dim_{\mathbb{F}_q} \mathcal{T}_e(s,g) \leq m \dim_{\mathbb{F}_{q^m}} \mathcal{R}(g^{q^e-q^{e-1}+1}),$$

we get

$$\begin{aligned}
\dim_{\mathbb{F}_q}(\Gamma(\mathcal{P}, sP_\infty, g)^\perp)^{\star 2} &\leq \min\left(ms'(q^e - q^{e-1} + 1) + \left(\frac{m-1}{2} - e\right)mk^2\right) \\
&\leq \min\left(\frac{m}{2}\left(2s'(q^e - q^{e-1} + 1) + k^2(m-1) - 2k^2e\right)\right) \\
&\leq \min\left(\binom{mk+1}{2} - \frac{m}{2}\left(k^2(2e+1) + k - 2s'(q^e - q^{e-1} + 1)\right)\right).
\end{aligned}$$

the minimum being taking over $e \in \left\{1, \ldots, \lfloor \frac{m}{2} \rfloor\right\}$. To get the best bound, we need to maximize the function

$$F(e) = ek^2 - s'(q^e - q^{e-1} + 1)$$

over $\left\{1, \ldots, \lfloor \frac{m}{2} \rfloor\right\}$. We compute the discrete derivative:

$$\begin{aligned}
\Delta F(e) = F(e+1) - F(e) &= (e+1)k^2 - s'(q^{e+1} - q^e + 1) - ek^2 + s'(q^e - q^{e-1} + 1) \\
&= k^2 - s'q^{e-1}(q-1)^2.
\end{aligned}$$

This function is decreasing with e, and the smallest value for which $\Delta F(e) \leq 0$ corresponds to its maximum. It is the smallest value of e such that $k^2 \leq s'q^{e-1}(q-1)^2$, *i.e.*

$$e = \left\lceil \log_q\left(\frac{k^2}{s'(q-1)^2}\right)\right\rceil + 1.$$

□

Several computational experiments showed then when the code $\mathcal{C} := \mathcal{C}_{\mathcal{L}}(\mathcal{X}_{a,b}, \mathcal{P}, sP_\infty + (g))$ is sufficiently random, the bound given in Theorem 4.24 is sharp, leading to a distinguisher if the parameters of $\mathcal{C}$ are not well–chosen. We give a concrete example showing the sharpness of our bound (computations have been done using Magma [BCP97]).

Example 4.25. Set $q = 3$ and $m = 3$. We consider the curve $\mathcal{X}$ over $\mathbb{F}_{q^m} = \mathbb{F}_{729}$ defined by

$$y^2 + y = x^3 + x + 2.$$

This elliptic curve $\mathcal{X}$ is a particular case of $C_{2,3}$ curve with genus $\mathfrak{g} = 1$. Set $s' = s + 1$ for $s \geq 0$, and $g \in \mathbb{F}_{q^m}(\mathcal{X})$ such that $g = x^\beta y^\alpha + g'$, where $a\beta + b\alpha = s + 1$; and g' is sampled at random in $\mathcal{L}(sP_\infty)$. For each such g, consider $\mathcal{P}_g := \mathcal{X}(\mathbb{F}_{q^m}) \setminus \mathrm{Supp}(g)$. Using Magma, we then compare the true dimension of the square of the dual of $\mathcal{C}_g := \Gamma(\mathcal{P}_g, sP_\infty, g)$ with the upper bound given in Theorem 4.24 for $s \in \{4, \ldots, 10\}$. Results can be found in Table 4.2.

In our computing experiments, we can check that g always has simple zeros, hence $\left[\frac{sP_\infty + (g)}{3}\right] = -P_\infty$. This example illustrates how the bound can be sharp when we are outside the scope of Proposition 4.11.

n	s	$\dim_{\mathbb{F}_q} \mathcal{C}_g$	$\dim_{\mathbb{F}_q} (\mathcal{C}_g)^{\star 2}$	$\dim_{\mathbb{F}_q} (\mathcal{C}_g^{\perp})^{\star 2}$	Upper bound in Theorem 4.24
781	4	757	781	234	234
783	5	753	783	327	327
782	6	746	782	402	402
783	7	741	783	483	483
782	8	734	782	570	570
782	9	728	782	663	663
781	10	721	781	762	762

Table 4.2: Sharpness of the bound.

In the last section, we discuss how to efficiently choose the parameters of a one–point Goppa–like AG code in order to resist this distinguisher.

4.4 Analysis of the distinguisher

In the previous section, we provided an (experimentally) sharp upper bound on the dimension of the square of the dual of a one–point Goppa–like AG code, which could lead to a distinguisher for the corresponding code. More precisely, let $\mathcal{C} := C_{\mathcal{L}}(\mathcal{X}_{a,b}, \mathcal{P}, sP_\infty + (g))$ be an AG code as above, with $\deg_{a,b}(g) := s' > s \geq 2\mathfrak{g}_{a,b} - 1$. We showed that if s and s' are such that $s \geq (s'-s)q + 2\mathfrak{g}_{a,b} - 1$, then

$$\dim_{\mathbb{F}_q} (\Gamma(\mathcal{P}, sP_\infty, g)^{\perp})^{\star 2} \leq \min\left(\frac{m}{2}\left(2s'(q^{e^*} - q^{e^*-1} + 1) + k^2(m - 1 - 2e^*)\right), n\right), \tag{4.17}$$

where $e^* := \left\lceil \log_q\left(\frac{k^2}{s'(q-1)^2}\right)\right\rceil + 1$. Thus, the code is distinguishable from a random one only if the right–hand side of Equation (4.17) is smaller than the length n of the code. It is possible to study when this case occurs, by starting to bound from above the maximal possible length: since $\mathcal{P} \cap \mathrm{Supp}(g) = \varnothing$, this maximum is reached when $\mathcal{P} = \mathcal{X}_{a,b}(\mathbb{F}_{q^m}) \backslash \{P_\infty\}$ and g has no rational zero, that is

$$n = \#\mathcal{P} = |\mathcal{X}_{a,b}(\mathbb{F}_{q^m})| - 1 \leq q^m + 2\mathfrak{g}_{a,b}\sqrt{q^m},$$

using the Hasse–Weil bound (Theorem 1.71). In order to protect the code against the distinguisher, the parameters have to be chosen such that

$$\frac{m}{2}\left(2s'(q^{e^*} - q^{e^*-1} + 1) + k^2(m - 1 - 2e^*)\right) \geq q^m + 2\mathfrak{g}_{a,b}\sqrt{q^m}. \tag{4.18}$$

Remark 4.26. As already discussed, the bound given in Theorem 4.24 looks to be sharp whenever the function g is *randomly* chosen. As it is an experimental consideration, we warn the reader that the condition given in Equation (4.18) might not be sufficient : more precisely, it may happens that our bound is bigger than n, but the real dimension of $(\Gamma(\mathcal{P}, sP_\infty, g)^{\perp})^{\star 2}$ is not.

In what follows, we focus on two specific classes of $C_{a,b}$ curves. First, we determine the maximal (with respect to the dimension) codes we can distinguish in the case where $\mathcal{X}_{a,b}$ is an elliptic curve. This case is relevant since it is the closest to the case of classical Goppa codes, and we will see that our results are very similar to the one given in [MT21]. Next up, we focus on the particular case of the Hermitian curve, which also turns out to be a $\mathcal{X}_{a,b}$ curve. It is well–known to be a good candidate to construct efficient codes as it is a maximal curve. Due to its high genus, we then show that any one–point Goppa–like code defined on it cannot be distinguished.

4.4.1 High rate distinguishable codes in the case of elliptic curves

Let $\mathcal{X}_{a,b}$ be an elliptic curve, *i.e.* $a = 2$ and $b = 3$. For some set of parameters which produces codes of cryptographic size, we compute the maximal distinguishable value of s. To get close to the case of classical Goppa codes, we also fix $s' = s + 1$.

As it was noticed in [MT21] and as we can see in Table 4.3, we are only able to distinguish high rate codes. The smallest distinguishable rates are roughly the same as the one given in [MT21].

q	m	n	Largest distinguishable s	Corresponding rate
2	12	4218	14	$0,963$
2	13	6688	18	$0,982$
3	7	2186	15	$0,962$
3	8	6393	24	$0,977$
5	5	3043	27	$0,961$
5	6	4500	22	$0,971$
5	6	6688	30	$0,976$
7	4	2395	27	$0,957$
7	5	4650	26	$0,971$
7	5	8192	37	$0,979$
17	3	4820	92	$0,943$

Table 4.3: Largest distinguishable Goppa–like AG code in elliptic case.

4.4.2 Codes on the Hermitian curve

As the Hermitian curve is a particular case of $C_{a,b}$ curve, we investigate the behaviour of one–point Goppa–like AG codes constructed on it with respect to our distinguisher. In particular, we show that all these codes resist to it, since Equation (4.18) always holds in this setting, essentially because the genus of the Hermitian curve is too high with respect to the size of the field. Let us first recall some known results about the Hermitian curve (see for example [Sti09]).

Let $m \geq 1$ be an even integer and denote by $q_0 := q^{m/2}$, so that $\mathbb{F}_{q^m} = \mathbb{F}_{q_0^2}$. The Hermitian curve $\mathcal{H}$ over $\mathbb{F}_{q_0^2}$ is defined by the equation

$$\mathcal{H} : y^{q_0} + y = x^{q_0+1}.$$

Its genus is given by $\mathfrak{g}_{\mathcal{H}} = \frac{q_0(q_0-1)}{2}$ and it is a maximal curve, *i.e.* $\#\mathcal{H}(\mathbb{F}_{q_0^2}) = q_0^3 + 1$.

Proposition 4.27. *Suppose $s \geq (s'-s)q + 2\mathfrak{g}_{\mathcal{H}} - 1$. Then for any choice of g and $\mathcal{P}$, the one–point Goppa–like code $\Gamma(\mathcal{P}, sP_\infty, g)$ resists the distinguisher given in Theorem 4.24.*

Proof. As discussed at the beginning of the section, the code cannot be distinguished whenever Equation (4.18) holds. In this particular case, we know exactly the number of rational points, hence the length n of the Goppa–like code is at most q_0^3. Since m is even, we are left to prove that

$$\mathfrak{B}(e^*) := ms'(q^{e^*} - q^{e^*-1} + 1) + \left(\frac{m}{2} - e^*\right) mk^2 \geq q_0^3, \tag{4.19}$$

where $k := \dim_{\mathbb{F}_{q^m}} \mathcal{C}_{\mathcal{L}}(\mathcal{H}, \mathcal{P}, sP_\infty + (g)) = s + 1 - \mathfrak{g}_{\mathcal{H}}$.

- If $e^* < \frac{m}{2}$, then $\mathfrak{B}(e^*) > mk^2$. Using the assumption on s and s', we know that $s \geq 2\mathfrak{g}_{\mathcal{H}} + q - 1$. This yields $k \geq \mathfrak{g}_{\mathcal{H}} + q$ and thus

$$\begin{aligned}
\mathfrak{B}(e^*) - q_0^3 &> m(\mathfrak{g}_{\mathcal{H}}^2 + 2\mathfrak{g}_{\mathcal{H}} q + q^2) - q_0^3 \\
&> \frac{m}{4}(q_0^4 - 2q_0^3 + q_0^2) + mq(q_0^2 - q_0 + q) - q_0^3 \\
&\geq \frac{1}{2}(q_0^4 - 2q_0^3 + q_0^2) + 4(q_0^2 - q_0 + 2) - q_0^3 \qquad (m \geq 2 \text{ and } q \geq 2) \\
&> \frac{q_0}{2}(q_0^3 - 4q_0^2 + 9q_0 - 8) > 0,
\end{aligned}$$

  since $q_0 \geq 2$. Inequality (4.19) holds in this case.

- If $e^* = \frac{m}{2}$, then since $q_0 = q^{m/2}$, we have $\mathfrak{B}\left(\frac{m}{2}\right) = ms'(q_0 - q_0 q^{-1} + 1)$. Moreover, $s' > s$ implies $s' \geq 2\mathfrak{g}_{\mathcal{H}} + q$, and

$$\begin{aligned}
\mathfrak{B}\left(\frac{m}{2}\right) - q_0^3 &\geq m(2\mathfrak{g}_{\mathcal{H}} + q)(q_0 - q_0 q^{-1} + 1) - q_0^3 \\
&\geq 2\left(q_0^2(q_0 - 1)\left(\frac{q-1}{q}\right) + q_0(q_0 - 1) + q_0(q-1) + q\right) - q_0^3 \\
&\geq q_0^3\left(2\left(\frac{q-1}{q}\right) - 1\right) + 2q_0^2\left(1 - \left(\frac{q-1}{q}\right)\right) + 2(q_0(q-2) + q).
\end{aligned}$$

Clearly, the last expression is minimal for $q = 2$, so we finally get

$$\mathfrak{B}\left(\frac{m}{2}\right) \geq q_0^2 + 4 > 0,$$

which proves (4.19) in this case and conclude the proof.

□

Consequently, it is still reasonable to consider the Hermitian curve to build efficient SSAG code–based cryptosystem. In Table 4.4, we provide parameters for one–point Goppa–like Hermitian codes that resist the distinguisher given in Section 4.3. They also improve key sizes compared to the subfield subcodes of 1–point Hermitian codes parameters reported in [NEK21, Tables 2 and 3], which already reduced key sizes compared with binary Goppa codes. The notation are as follow:

- q_0 is such that our codes are defined over $\mathbb{F}_{q_0^2}$ and $m = 2$ ($i.e$ $q = q_0$);
- s is the degree of the divisor $D = sP_\infty + (g)$;
- n and k denote the length and the dimension of our codes, respectively;
- t is the correction capability;
- Prange complexity denotes the exponent in the complexity of running the ISD algorithm (see. [Pra62]);
- Key sizes are computed via the formula $k(n-k)\lceil \log_2(q) \rceil$, and expressed in Bytes.

q_0	s	n	k	t	Prange complexity	Key size (Bytes)
11	265	1320	898	77	153	142 108
13	312	2188	1718	77	198	302 798
16	354	4078	3608	56	199	847 880
13	490	2189	1363	166	270	422 189
16	460	4080	3398	109	313	1 158 718

Table 4.4: Goppa-Like Hermitian codes parameters $\Gamma(\mathcal{P}, sP_\infty, g)$ over $\mathbb{F}_{q^2}$.

Chapter 5

IOP of Proximity to AG codes on the Hermitian tower

The context of this Chapter can be found in Section 2.2. Here, we are interested in proximity tests to several families of AG codes, as they are good candidates to construct short proof systems. The main idea is to adapt the protocol FRI described in Section 2.2.3, which is an efficient IOP system to test proximity to a Reed–Solomon code. This Chapter is based on [BNLR22], in which IOPPs for some families of AG code are proposed and studied: the case of Kummer codes and the case of AG codes defined over the so–called Hermitian tower. As my contribution to this work concerns the second family, we focus on this case. At several points, some technical results will be given without proofs, in which cases we refer the reader to [BNLR22].

In 2020, Bordage and Nardi [BN20] gave a clear criterion for constructing IOPPs for AG codes with linear proof length and sublinear query complexity, as well as a concrete instantiation for AG codes defined over Kummer curves. In the case of AG code defined over a tower of Hermitian curve, we then provide a family of foldable codes compatible with their definition of proximity testing, as well as properties of the IOPP than can be derived from it.

The Chapter is organized as follows: In Section 5.1, we start by giving a definition for AG codes to be compatible with proximity testing. Then, an explicit family of foldable AG code along the Hermitian tower is given in Section 5.2. Finally, Sections 5.3 and 5.4 are dedicated respectively to the folding operator used to reduce the proximity test to a smaller code and the corresponding IOPP and its properties.

5.1 Sequence of AG codes compatible with proximity tests

In this section, we give a general definition of foldable AG codes, valid both in the case of codes along the Hermitian tower and codes over Kummer–type curve (treated in [BNLR22]). More precisely, this can be done by considering a sequence of curves equipped with automorphisms subgroups. As we want to keep this section quite general, let $\mathbb{F}$ be any finite field.

5.1.1 Sequence of curves

Let $\mathcal{X}$ be a curve defined over the finite field $\mathbb{F}$ and a finite solvable group $\mathcal{G} \subseteq \mathrm{Aut}(\mathcal{X})$. By solvability of $\mathcal{G}$, there exists a sequence of subgroups

$$\{\mathrm{Id}\} := \mathcal{G}_0 \lhd \mathcal{G}_1 \lhd \cdots \lhd \mathcal{G}_r := \mathcal{G}, \tag{5.1}$$

such that each $\mathcal{G}_{i-1}$ is a normal subgroup of $\mathcal{G}_i$ and the factor group $\Gamma_i := \mathcal{G}_{r+1-i}/\mathcal{G}_{r-i} \simeq \mathbb{Z}/p_i\mathbb{Z}$ is cyclic of order p_i (for $1 \leq i \leq r$). In particular, the cardinality of $\mathcal{G}$ equals $|\mathcal{G}| = \prod_{i=1}^{r} p_i$.

From Galois theory, the group $\Gamma_r = \mathcal{G}_1/\mathcal{G}_0 = \mathcal{G}_1$ acts on $\mathcal{X}_r := \mathcal{X}$, we then define the corresponding quotient curve $\mathcal{X}_{r-1} := \mathcal{X}_r/\Gamma_r$. Repeating this process for each $i \in \{1, \cdots, r\}$, we recursively obtain a sequence of curves as follows:

$$\mathcal{X}_r := \mathcal{X} \text{ and } \mathcal{X}_{i-1} := \mathcal{X}_i/\Gamma_i.$$

We set $F_i := \mathbb{F}(\mathcal{X}_i)$ their corresponding function field and we denote by $\pi_i : \mathcal{X}_i \to \mathcal{X}_{i-1}$ the canonical

projection modulo the action of Γ_i. We obtain a sequence of curves

$$\mathcal{X} := \mathcal{X}_r \xrightarrow{\pi_r} \mathcal{X}_{r-1} \xrightarrow{\pi_{r-1}} \cdots \xrightarrow{\pi_{i+1}} \mathcal{X}_i \xrightarrow{\pi_i} \mathcal{X}_{i-1} \longrightarrow \cdots \xrightarrow{\pi_1} \mathcal{X}_0 \simeq \mathcal{X}/\mathcal{G}, \tag{5.2}$$

which corresponds via Theorem 1.35 to a tower of function field

$$F_0 \subseteq F_1 \subseteq \cdots \subseteq F_r := \mathbb{F}(\mathcal{X}),$$

where for each $i \in \{0, \cdots, r\}$, $F_i := \mathbb{F}(\mathcal{X}_i)$. Even if the sequence of curves depends on the choice of the normal series (5.1), the last curve $\mathcal{X}_0$ is always isomorphic to $\mathcal{X}/\mathcal{G}$. From now on, such a sequence of curves is referred to as an $(\mathcal{X}, \mathcal{G})$–*sequence*.

5.1.2 Sequence of codes

Let $(\mathcal{X}_i)_i$ be a $(\mathcal{X}, \mathcal{G})$–sequence as above. For any $i \in \{0, \ldots, r\}$, we aim to define an AG code

$$C_i := C_{\mathcal{L}}\left(\mathcal{X}_i, \mathcal{P}_i, G_i\right)$$

on the curve $\mathcal{X}_i$ associated to a divisor $G_i \in \mathrm{Div}(F_i)$ and a support $\mathcal{P}_i \subseteq \mathcal{X}_i(\mathbb{F})$. The upcoming discussion explains how to choose $\mathcal{P}_i$ and G_i.

Choice of the evaluation sets. For our protocol, we need, for each $i \in \{1, \cdots, r\}$, that every point in $\mathcal{P}_{i-1}$ admits exactly $p_i := |\Gamma_i|$ preimages under the projection π_i. For this reason, we choose the first support $\mathcal{P}_r \subseteq \mathcal{X}(\mathbb{F})$ as a (disjoint) union of $\mathcal{G}$–orbits of size $|\mathcal{G}|$, *i.e.* such that $\mathcal{G}$ acts freely on $\mathcal{P}_r$. This way, we can define for every $i \in \{1, \cdots, r\}$ the set $\mathcal{P}_{i-1} := \pi_i(\mathcal{P}_i)$.

Choice of the divisors. Let $G_r \in \mathrm{Div}(F_r)$ be a divisor that is globally Γ_r–invariant. This ensures that $\mathrm{Supp}(G_r) \cap \mathcal{P}_r = \varnothing$. For simplicity, we will assume that G_r is in fact supported by Γ_r–fixed points.

Remark 5.1. As done several time in this thesis, we will later consider one–point divisors, whose support is reduced to a point which totally ramifies in the tower. Under the action of $\mathcal{G}$, this means that such a point is a full orbit.

To make our protocol complete and sound, we need the sequence of divisors $(G_i)_{i=0}^r$ to satisfy the following properties:

- each G_i is supported by Γ_i–invariant points;
- each Riemann–Roch space $\mathcal{L}_{F_i}(G_i)$ admits an explicit decomposition in terms of Riemann–Roch spaces on the fixed field F_{i-1} (see Equation (5.3));
- for any $1 \leq i \leq r$, G_{i-1} needs to be *compatible* with the choice of G_i and the structure of its Riemann–Roch space (explanation is delayed to Definition 5.4).

We know discuss these restrictions by definition what we mean by decomposition of Riemann–Roch spaces.

Definition 5.2. Let $i \in \{1, \cdots, r\}$ and $G_i \in \mathrm{Div}(F_i)$. We say that the function μ_i *partitions* $\mathcal{L}_{F_i}(G_i)$ with respect to the action of the order p_i subgroup Γ_i if

$$\mathcal{L}_{F_i}(G_i) = \bigoplus_{j=0}^{p_i-1} \mu_i^j \pi_i^* \left(\mathcal{L}_{F_{i-1}}(E_{i,j})\right), \tag{5.3}$$

with

$$E_{i,j} := \left\lfloor \frac{1}{p_i} \pi_{i*} \left(G_i - j(\mu_i)^{F_i}\right) \right\rfloor \in \mathrm{Div}(F_{i-1}) \text{ for } 0 \leq j \leq p_i - 1,$$

where given a divisor $D = \sum_{P \in \mathbb{P}_{F_i}} n_p P \in \mathrm{Div}(F_i)$, we set:

$$\left\lfloor \frac{1}{n} D \right\rfloor = \sum_{P \in \mathbb{P}_{F_i}} \left\lfloor \frac{n_P}{n} \right\rfloor P$$

and

$$\pi_{i*} D = \sum_{P \in \mathbb{P}_{F_i}} n_P \pi_i(P).$$

Remark 5.3. In the literature, the divisor $\pi_{i*}(D)$ is sometimes referred to as the *pushforward* of the divisor D. We warn the reader that this definition differs from our notion of pushforward, as defined in Definition 3.2 with the notation $\widetilde{D}$.

Definition 5.4 (Compatibility)**.** Fix $i \in \{1, \cdots, r\}$, $G_i \in \mathrm{Div}(F_i)$ and a function $\mu_i \in F_i$ which partitions $\mathcal{L}_{F_i}(G_i)$ in the sense of Equation (5.3). A divisor $G_{i-1} \in \mathrm{Div}(F_{i-1})$ is said to be (G_i, μ_i)–*compatible* if the following assertions hold:

1. for every $j \in \{0, \cdots, p_i - 1\}$, $G_{i-1} \geq E_{i,j}$;
2. for every $j \in \{0, \cdots, p_i - 1\}$, there exists a function $\nu_{i-1,j} \in F_{i-1}$ such that

$$(\nu_{i-1,j})_\infty^{F_{i-1}} = G_{i-1} - E_{i,j}.$$

The functions $\nu_{i-1,j}$ are called *balancing functions*.

The first requirement imposed on the sequence of divisors ensures that $\mathcal{L}(E_{i,j}) \subseteq \mathcal{L}(G_{i-1})$ and second means that for every $f_j \in \mathcal{L}(E_{i,j})$, the function $\nu_{i-1,j} f_j$ lies in $\mathcal{L}(G_{i-1})$. We now come to the definition of foldable AG code.

Definition 5.5 (Foldable AG codes)**.** Let $\mathcal{C} = C_{\mathcal{L}}(\mathcal{X}, \mathcal{P}, G)$ be an AG code on a curve $\mathcal{X}$. We say it is *foldable* if the following conditions are satisfied:

1. there exists a finite solvable group $\mathcal{G} \subseteq \mathrm{Aut}(\mathcal{X})$ that acts freely on $\mathcal{P}$. A composition series of $\mathcal{G}$ as in Equation (5.1) provides an $(\mathcal{X}, \mathcal{G})$–sequence of curves $(\mathcal{X}_i)_i$;
2. there exists $e \in (0, 1)$ such that $|\mathcal{G}| > |\mathcal{P}|^e$;
3. there exists two sequences $(\mu_i)_i \in F_i = \mathbb{F}(\mathcal{X}_i)$ and $(G_i)_i \in \mathrm{Div}(\mathcal{X}_i)$ such that $G_r = G$ and for every $i \in \{1, \cdots, r\}$:
   - the divisor G_i is supported by Γ_i–fixed points;
   - the function μ_i partitions $\mathcal{L}_{\mathcal{X}_i}(G_i)$ in the sense of Definition 5.2;
   - G_{i-1} is (G_i, μ_i)–compatible (Definition 5.4).

Remark 5.6. The second requirement in Definition 5.4 is really compelling and requires geometric knowledge about the curves $\mathcal{X}_i$. In fact, on a general curve, not every effective divisor is the pole locus of a function, and characterizing which divisor indeed arises this way is at the center of the Weierstrass gap theory, at least while considering one–point divisors. We will come back at discussion in concrete instances, while seeking for balancing functions $\nu_{i-1,j}$.

5.2 Foldable AG codes along the Hermitian tower

5.2.1 Preliminaries

Let $\mathbb{F}_q$ be a finite field. In what follows, we deal with curves and AG codes defined over the field $\mathbb{F}_{q^2}$, as the well–known Hermitian curve as to be defined over a field of square cardinality. The study of the Hermitian tower has been initiated in [She93]. In addition, details about the Hermitian curve can also be found in [Sti09, Section 6.4]. In the discussion below, we describe an efficient way to build a sequence of AG codes $C_{\mathcal{L}}(\mathcal{X}_i, \mathcal{P}_i, G_i)$ along a tower of Hermitian curves.

Sequence of curves. We consider the sequence of function fields $\mathcal{F} = (F_i)_{i \geq 0}$ over $\mathbb{F}_{q^2}$, recursively defined by $F_0 = \mathbb{F}_{q^2}(x_0)$ and $F_i = F_{i-1}(x_i)$, where

$$x_i^q + x_i = x_{i-1}^{q+1}, \text{ for all } i \geq 1. \tag{5.4}$$

According to Theorem 1.35, the tower $\mathcal{F}$ corresponds to a tower of curves $(\mathcal{X}_i)_{i \geq 0}$ such that $F_i = \mathbb{F}_{q^2}(\mathcal{X}_i)$, for any $i \geq 0$. Also, we can view each curve $\mathcal{X}_i$ embedded into an i–dimensional projective space with variables $(x_0, \cdots, x_n)$ defined by the equations (5.4).

Remark 5.7. For $i = 1$, the field F_1 is nothing but the Hermitian function field $\mathcal{H}$ over $\mathbb{F}_{q^2}$, those main properties are given in [Sti09, Lemma 6.4.4].

Let $\mathfrak{g}_i := \mathfrak{g}(F_i)$ denotes the genus of the function field F_i. An explicit formula for every $i \geq 0$ is given by the following proposition:

Proposition 5.8 ([She93, Proposition 4]). *We have* $\mathfrak{g}_0 = 0$ *and for all* $i \geq 1$,

$$\mathfrak{g}_i = \frac{1}{2}[(q^2-1)((q+1)^i - q^i) + 1 - q^i] = \frac{1}{2}\left(\sum_{k=1}^{i} q^{i+1}\left(1+\frac{1}{q}\right)^{k-1} + 1 - (1+q)^i\right). \tag{5.5}$$

Additionally, for every $i \geq 0$, we can recursively prove that the number of $\mathbb{F}_{q^2}$–rational places in $\mathcal{X}_i$ is given by

$$|\mathcal{X}_i(\mathbb{F}_{q^2})| = q^{i+2} + 1.$$

We end up with an infinite sequence of curves $(\mathcal{X}_i)_{i\geq 0}$, called the *Hermitian tower*, as follows:

$$\ldots \xrightarrow{\pi_{i+1}} \mathcal{X}_i \xrightarrow{\pi_i} \mathcal{X}_{i-1} \xrightarrow{\pi_{i-1}} \ldots \xrightarrow{\pi_1} \mathcal{X}_0 \simeq \mathbb{P}^1, \tag{5.6}$$

where Γ_i stands for the automorphism acting on $\mathcal{X}_i$ and $\pi_i : \mathcal{X}_i \to \mathcal{X}_{i-1}$ is the corresponding quotient map. This tower is a specific tower of Artin–Schreier extensions, which have been extensively studied (see [Sti09], Section 3.7). We now recall some classical results that will be useful to design fordable AG codes along this tower.

Automorphisms and projection maps. By definition of the Hermitian tower [Sti09, Proposition 3.7.10], the Galois group of the extension F_i/F_{i-1} is the group of automorphisms defined by $(x_1, ..., x_{i-1}, x_i) \mapsto (x_1, ..., x_{i-1}, x_i + \alpha)$, where α runs in

$$S = \left\{\alpha \in \mathbb{F}_{q^2} \mid \alpha^q + \alpha = 0\right\}.$$

Note that if we fix a non–zero element $\alpha \in S$, then for every $\beta \in \mathbb{F}_q$, $\alpha\beta$ also lies in S. Hence, S is an additive group which is isomorphic to $\mathbb{F}_q$. The corresponding projection map $\pi_i : \mathcal{X}_i \to \mathcal{X}_{i-1}$ consists in the projection onto the first i coordinates.
For every $i \geq 0$, we set $\Pi_i : \mathcal{X}_i \to \mathcal{X}_0$ to be the composition of the first i quotient maps, *i.e.*

$$\Pi_i := \pi_1 \circ \pi_2 \circ \cdots \circ \pi_{i-1} \circ \pi_i. \tag{5.7}$$

The point at infinity. In what follows, let us denote by $P_\infty^{(0)}$ the unique pole of the function x_0 in the rational function field F_0, which corresponds to the point at infinity on the projective line $\mathcal{X}_0 = \mathbb{P}^1$. The following lemma gives us the ramification behaviour of this point alongside the tower.

Lemma 5.9. *Let* $i \geq 1$. *The place* $P_\infty^{(0)}$ *is totally ramified in* F_i, *which means that the preimage* $\Pi_i^{-1}\left(\left\{P_\infty^{(0)}\right\}\right)$ *consists in a unique place in* F_i, *denoted by* $P_\infty^{(i)}$. *Moreover,* $P_\infty^{(0)}$ *is the unique place that is ramified in the tower* $\mathcal{F}$.

Proof. See [She93] or [GX12, Section 3.1] for a more recent summary. □

This specific behaviour of the point at infinity encourages us to define a sequence of one–point AG codes associated to it, meaning that our divisors $G_i \in \mathrm{Div}(F_i)$ will be taken as multiple of the corresponding point at infinity $P_\infty^{(i)}$, *i.e.*

$$G_i := d_i P_\infty^{(i)} \text{ , for any } i \geq 0,$$

where $d_i > 0$ which will be chosen carefully later on.

Let us now focus on the principal divisors $(x_j)^{F_i} (0 \leq j \leq i)$ and their valuation at $P_\infty^{(i)}$ along the tower.

Lemma 5.10. *The following two assertions hold:*

1. *For* $i \geq 0$, *we have*
$$(x_i)^{F_i} = (q+1)^i \left(P^{(i)} - P_\infty^{(i)}\right),$$
where $P^{(i)}$ *is the unique common zero of the functions* $x_0, \cdots, x_i$.

2. *Let* $i \geq 0$. *Then for* $0 \leq j \leq i$, *the valuation of the function* $x_j \in F_i$ *at* $P_\infty^{(i)}$ *is given by*
$$\nu_{P_\infty^{(i)}}(x_j) = -q^{i-j}(q+1)^j.$$

Proof. 1. For this part of the proof, see also [BFGM16, Section 2]. We prove the result by induction on $i \geq 0$. Let $P^{(0)}$ be the zero of x in the rational function field $F_0 = \mathbb{F}_{q^2}(x_0)$, *i.e.*

$$(x_0) = P^{(0)} - P_\infty^{(0)}.$$

Since F_1 is the Hermitian function field (particular case of Artin–Schreier extension), it is well-known ([Sti09, Lemma 6.4.4]) that $P^{(0)}$ completely splits in F_1/F_0. More precisely, there are exactly q elements $\beta \in \mathbb{F}_{q^2}$ such that $\beta^q + \beta = 0$, and each of them corresponds to a place $P_{0,\beta}^{(1)}$ in F_1 such that $x_0(P_{0,\beta}^{(1)}) = 0$ and $x_1(P_{0,\beta}^{(1)}) = \beta$. With such notation, we denote by $P^{(1)} := P_{0,0}^{(1)}$ the unique common zero of x_0 and x_1 in F_1. From the equation $x_1^q + x_1 = x_0^{q+1}$, the functions x_0 and x_1 have the same zeros in F_1, and

$$q \cdot \nu_{P_\infty^{(1)}}(x_1) = -(q+1) \cdot e(P_\infty^{(1)} | P_\infty^{(0)}),$$

that is $\nu_{P_\infty^{(1)}}(x_1) = -(q+1)$. Consequently, $(x_1)^{F_1} = (q+1)\left(P^{(1)} - P_\infty^{(1)}\right)$, which proves the result for $i = 1$. Since any extension F_i/F_{i-1} corresponds to the same Artin–Schreier extension, we can recursively prove that for any $i \geq 0$, the functions $x_i, x_{i-1}, \cdots, x_1, x_0$ have only one common zero in F_i, denoted by $P^{(i)}$. More precisely, this place arises as the only extension in F_i of $P^{(i-1)}$ such that $x_i(P^{(i)}) = 0$. Using this time the equation $x_i^q + x_i = x_{i-1}^{q+1}$, we deduce from the formula for $(x_{i-1})^{F_{i-1}}$ that

$$q \cdot \nu_{P_\infty^{(i)}}(x_i) = -(q+1)(q+1)^{i-1} \cdot e(P_\infty^{(i)} | P_\infty^{(i-1)}),$$

hence $\nu_{P_\infty^{(i)}}(x_i) = -(q+1)^i$.

2. Consequence of 1 applied to $(x_j)^{F_j}$, and the fact that

$$e\left(P_\infty^{(i)} | P_\infty^{(j)}\right) = [F_i : F_j] = q^{i-j}.$$

□

Basis of the Riemann–Roch space associated with the divisor $d_i P_\infty^{(i)}$. Given $i \geq 0$, $P_\infty^{(i)}$ is the unique pole of all functions $x_0, \cdots, x_i$. Since we know all their valuation at $P_\infty^{(i)}$ (see Lemma 5.10, *2*), we have an explicit basis for of the Riemann–Roch space associated to the one–point divisor $mP_\infty^{(i)}$.

Lemma 5.11. *For all $i \geq 1$ and $m \geq 1$, we have*

$$\mathcal{L}_{F_i}(mP_\infty^{(i)}) = \text{Span}\left(x_0^{a_0} \cdots x_i^{a_i} \mid 0 \leq a_0, 0 \leq a_j \leq q-1 \text{ and } \sum_{j=0}^{i} a_j q^{i-j}(q+1)^j \leq m \right).$$

Proof. It is clear that all functions $x_0^{a_0}, \cdots, x_i^{a_i}$ belong to $\mathcal{L}_{F_i}(mP_\infty^{(i)})$. To show that they indeed form a basis, we make use of the Weierstrass gap theory at $P_\infty^{(i)}$ (whose Weierstrass semigroup is defined in Equation (5.8)). More precisely, we show that we have exactly $\dim_{\mathbb{F}_{q^2}} \mathcal{L}_{F_i}(mP_\infty^{(i)})$ such functions. Details can be found in [She93, Proposition 6]. □

5.2.2 Construction of foldable AG codes

We aim to define a sequence of AG codes on the tower $(\mathcal{X}_i)_{i \geq 0}$ that is compatible with the definition of our folding operator (see Definition 5.16). More especially, for some fixed $i \geq 0$, we consider the one–point AG code

$$C_{\mathcal{L}}\left(\mathcal{X}_i, \mathcal{P}_i, G_i\right), \text{ where } \mathcal{P}_i \subseteq \mathcal{X}_i(\mathbb{F}_{q^2}) \backslash \left\{P_\infty^{(i)}\right\} \text{ and } G_i = d_i P_\infty^{(i)},$$

defined on the curve $\mathcal{X}_i$. To obtain a sequence of foldable codes, we need to describe the Riemann–Roch space of G_i using Riemann–Roch spaces on lower curves. In the case of AG code over Kummer–type curves, a decomposition as in Equation (5.3) is performed by using [Mah04, Theorem 2]. Unfortunately, the latter theorem only works for specific cyclic extensions whose degree is prime to the characteristic of the base field, which is not the case here since the degree of our automorphism exactly equals the characteristic. Hence, we have to find a decomposition *by hand*, which can be done thanks to the explicit basis of the Riemann–Roch spaces given in Lemma 5.11, and is the subject of the next proposition.

Proposition 5.12. *Let $i \geq 1$. Set $G_i = d_i P_\infty^{(i)}$ for some integer $d_i > 0$. Then*

$$\mathcal{L}_{F_i}(G_i) = \bigoplus_{j=0}^{q-1} x_i^j \pi_i^* \left(\mathcal{L}_{F_{i-1}}(E_{i,j}) \right)$$

with

$$E_{i,j} := \left\lfloor \frac{1}{q} \pi_{i*} \left(G_i - j(x_i)^{F_i} \right) \right\rfloor \in \mathrm{Div}(F_{i-1}), \text{ for } 0 \leq j \leq q-1,$$

where the notation π_{i*} was defined in 5.2. In other words, the function $x_i \in F_i$ partitions $\mathcal{L}_{F_i}(G_i)$ in the sense of to Definition 5.4.

Proof. From Lemma 5.11, $\mathcal{L}_{F_i}(G_i)$ contains linear combinations of monomials of the form $x_0^{a_0} \cdots x_i^{a_i}$; with $a_0 \geq 0$, $0 \leq a_j \leq q-1$ for $0 < j \leq i$ and $\sum_{j=0}^{i} a_j q^{i-j} (q+1)^j \leq m$. Since the $a_j's$ (for $j > 0$) run in $\{0, \cdots, q-1\}$, the proof follows from the fact that the function $x_0^{a_0} \cdots x_{i-1}^{a_{i-1}} \in F_i$ lies in the Riemann–Roch space $\mathcal{L}_{F_i}\left(G_i - j(x_i)^{F_i}\right)$, which means that $x_0^{a_0} \cdots x_{i-1}^{a_{i-1}}$ (seen in F_{i-1} this time) belongs to $\mathcal{L}_{F_{i-1}}(E_{i,j})$. □

In order to make G_{i-1} compatible with (G_i, x_i), we need the existence of q balancing functions $\nu_{i-1,j} \in F_{i-1}$ (for $0 \leq j \leq q-1$) such that

$$G_{i-1} - E_{i,j} = (\nu_{i-1,j})_\infty^{F_{i-1}}.$$

The divisor $(x_i)^{F_i}$ being known (see Lemma 5.10, *1*), we have

$$E_{i,j} = \left\lfloor \frac{d_i - j(q+1)^i}{q} \right\rfloor P_\infty^{(i-1)}, \text{ for all } i, j.$$

Hence, we have to *balance* the divisors

$$G_{i-1} - E_{i,j} = \left(d_{i-1} - \left\lfloor \frac{d_i - j(q+1)^i}{q} \right\rfloor \right) P_\infty^{(i-1)},$$

which leads to study the Weierstrass semigroup at $P_\infty^{(i-1)}$, denoted from now on by $\mathcal{H}\left(P_\infty^{(i-1)}\right)$ (see [Sti09, Section 1.6]). The generators of this semigroup can be found by using Lemma 5.10, since $P_\infty^{(i-1)}$ is the unique pole of the functions $x_0, \cdots, x_{i-1}$ and we know their valuation at this point. More precisely, we have

$$\mathcal{H}\left(P_\infty^{(i-1)}\right) = \left\langle q^{i-1-k}(q+1)^k \mid 0 \leq k \leq i-1 \right\rangle_{\mathbb{N}}. \tag{5.8}$$

Remark 5.13. In the spirit of the FRI protocol (see Section 2.2.3), we could be tempted to choose G_{i-1} as $E_{i,0}$. Such a choice would be valid in the sense of Definition 5.4 if and only if there exists, for every $0 \leq j \leq q-1$, a balancing function $\nu_{i-1,j} \in F_{i-1}$ such that $G_{i-1} - E_{i,j} = (\nu_{i-1,j})_\infty^{F_{i-1}}$, *i.e.*

$$\left\lfloor \frac{d_i}{q} \right\rfloor - \left\lfloor \frac{d_i - j(q+1)^i}{q} \right\rfloor \in \mathcal{H}\left(P_\infty^{(i-1)}\right).$$

However, when i increases, this condition is never satisfied, meaning that we will have to make a smarter choice for G_{i-1}.

To ensure that $\deg\left(G_{i-1} - E_{i,j}\right)$ is never a Weierstrass gap at $P_\infty^{(i-1)}$, the idea is to increase the degree d_{i-1} of G_{i-1}. Taking the Weierstrass Gap Theorem [Sti09, Theorem 1.6.8] into consideration, we can prove:

Theorem 5.14. *Let $i \geq 1$. Suppose $G_i = d_i P_\infty^{(i)}$ for some integer d_i. We set $G_{i-1} = d_{i-1} P_\infty^{(i-1)}$, with*

$$d_{i-1} := \left\lfloor \frac{d_i}{q} \right\rfloor + 2\mathfrak{g}_{i-1}.$$

Then G_{i-1} is (G_i, x_i)–compatible (see Definition 5.4).

Proof. By the Weierstrass Gap Theorem [Sti09, Theorem 1.6.8], we know that

$$\max\left(\mathbb{N}\backslash\mathcal{H}\left(P_\infty^{(i-1)}\right)\right) \leq 2\mathfrak{g}_{i-1} - 1.$$

Then, for any $0 \leq j \leq q-1$, the difference

$$m_{i,j} := \deg(G_{i-1} - E_{i,j}) = \left(\left\lfloor \frac{d_i}{q} \right\rfloor - \left\lfloor \frac{d_i - j(q+1)^i}{q} \right\rfloor + 2\mathfrak{g}_{i-1}\right) \tag{5.9}$$

always belongs to the Weierstrass semigroup at $P_\infty^{(i-1)}$, meaning that some balancing function does exist. □

About the balancing functions. Since we know a $\mathbb{N}$–basis of $\mathcal{H}\left(P_\infty^{(i-1)}\right)$ (see Equation (5.8)), we are able to explicit the form of the balancing functions $\nu_{i-1,j}$, for any $1 \leq i \leq r$ and $0 \leq j \leq q-1$. In particular, for a fixed i, they can be chosen as product of powers of $x_0, \cdots, x_{i-1}$. More precisely, if $a_{i,j} := (a_{i,j}(0), \cdots, a_{i,j}(i-1)) \in \mathbb{N}^i$ is a vector of integers such that

$$m_{i,j} = \sum_{k=0}^{i-1} a_{i,j}(k) \cdot q^{i-1-k}(q+1)^k, \tag{5.10}$$

then $m_{i,j}$ (defined in Equation (5.9)) is in fact in $\mathcal{H}\left(P_\infty^{(i-1)}\right)$. The corresponding choice for the balancing function is then given by

$$\nu_{i,j} = \prod_{k=0}^{i-1} x_k^{a_{i,j}(k)}.$$

Note that finding a vector $a_{i,j}$ satisfying Equation (5.10) leads to study the diophantine equation

$$m_{i,j} = \sum_{k=0}^{i-1} a_k \cdot q^{i-1-k}(q+1)^k$$

with i unknowns $a_k \in \mathbb{N}$, for which we know by construction that there exist solutions (and we only need one).

A family of foldable AG codes. Let us denote by $i_{\max}$ the level in the tower $(\mathcal{X}_i)_{i\geq 0}$ such that $\mathcal{X}_{i_{\max}}$ is the curve on which the code we want to test proximity is defined.

Proposition 5.15. *Let $\mathcal{P}_0 \subseteq \mathbb{P}^1(\mathbb{F}_{q^2})\backslash\left\{P_\infty^{(0)}\right\}$ and define $\mathcal{P}_{i_{\max}} \subseteq \mathcal{X}_{i_{\max}}(\mathbb{F}_{q^2})$ as the preimage of $\mathcal{P}_0$ under $\Pi_{i_{\max}}$ (defined in (5.7)). For any fixed integer $d_{i_{\max}}$, the AG code $C_{\mathcal{L}}\left(\mathcal{X}_{i_{\max}}, \mathcal{P}_{i_{\max}}, d_{i_{\max}}P_\infty^{(i_{\max})}\right)$ is foldable in the sense of Definition 5.5.*

Proof. The group $\mathcal{G} \simeq \mathbb{Z}/p^{i_{\max}}\mathbb{Z}$ acts on the curve $\mathcal{X}_{i_{\max}}$, and its action on $\mathcal{P}_{i_{\max}}$ is obviously free by definition of $\mathcal{P}_{i_{\max}}$. We have $|\mathcal{P}_{i_{\max}}| = |\mathcal{P}_0|p^{i_{\max}}$, implying the existence of some $e \in (0,1)$ such that $|\mathcal{G}| > |\mathcal{P}_{i_{\max}}|^e$. Finally, the third and last condition for an AG code to be foldable (see Definition 5.5) follows from Theorem 5.14. □

To make sure we get compatible divisors and as stated in Theorem 5.14, we need to increase the degree of each folded divisor by twice the genus of the curve at each step. The counterpart is that the dimension of each folded code decreases much slowly than their length. To construct a sound AG–IOPP system based on this family, we need to ensure that we do not get a last code (*i.e.* a Reed–Solomon code) which is trivial. To avoid this problem, we need to control the dimension of each foldable code. An efficient way to do this is to consider codes of the specific form

$$\mathcal{C}_\alpha := \mathcal{C}_{\mathcal{L}}\left(\mathcal{X}_{i_{\max}}, \mathcal{X}_{i_{\max}}(\mathbb{F}_{q^2})\backslash\left\{P_\infty^{(i_{\max})}\right\}, (2\alpha+1)\mathfrak{g}_{i_{\max}}P_\infty^{(i_{\max})}\right), \tag{5.11}$$

for some $\alpha > \frac{1}{2}$. In fact, the choice $\mathcal{P}_{i_{\max}} = \mathcal{X}_{i_{\max}}(\mathbb{F}_{q^2})\backslash\left\{P_\infty^{(i_{\max})}\right\}$ is the biggest possible, and is in fact $\mathcal{G}$–invariant since it is easily proved (by induction) that every rational point on $\mathcal{X}_{i_{\max}}$, excepted for $P_\infty^{(i_{\max})}$, totally splits in the tower. In this case, the length of the code is $n_{i_{\max}} = q^{i_{\max}+2}$.

In [BNLR22], we provide a sufficient condition on α and $i_{\max}$ to get a constant fixed rate when q goes to infinity, which can be done be studying the genus formula given in Proposition 5.8. The code $\mathcal{C}_\alpha$ will be used to control the rate of the last folded code (see Section 5.4.2).

5.3 Folding operators for AG codes

In Section 5.2.2, we determined the needed properties for an AG code to be foldable (in the case of the Hermitian tower). We now construct the so–called *folding operator*, which ensures that at each step of the IOPP protocol, the proximity test of a given AG code can be reduced to the proximity of its *folded code*. We then study its properties.

To keep the framework general, we use below the same notation as in Section 5.1. Let $(\mathcal{C}_i)_{i=0}^r = (C_{\mathcal{L}}(\mathcal{X}_i, \mathcal{P}_i, G_i))_{i=0}^r$ be a family of foldable AG codes on a sequence of curves $(\mathcal{X}_i)_i$ defined over some finite field $\mathbb{F}$, where $\mathcal{C}_r$ satisfies all conditions in Definition 5.5. In the context of an IOPP, we want to test proximity of a function $f^{(r)} : \mathcal{P}_r \to \mathbb{F}$ to the code $\mathcal{C}_r$. To do so, we aim to inductively reduce the problem to a smaller one, consisting of testing proximity to some smaller code $\mathcal{C}_i$. More precisely, our goal is to define from any function $f^{(i)} : \mathcal{P}_i \to \mathbb{F}$ another function $f^{(i-1)} : \mathcal{P}_{i-1} \to \mathbb{F}$ such that the relative distance $\Delta(f^{(i-1)}, \mathcal{C}_{i-1})$ is roughly equal to $\Delta(f^{(i)}, \mathcal{C}_i)$.

The folding operator. Fix $i \in \{1, \dots, r\}$ and consider an arbitrary function $f : \mathcal{P}_i \to \mathbb{F}$. For each $P \in \mathcal{P}_{i-1}$, we denote by $S_P := \pi_i^{-1}(\{P\})$ the set of p_i places in $\mathcal{X}_i$ above P. Let

$$I_{f,P}(X) := \sum_{j=0}^{p_i-1} a_{j,P} X^j \in \mathbb{F}[X]$$

be the univariate polynomial of degree less than p_i which interpolates the set of points

$$\{(\mu_i(Q), f(Q)) \mid Q \in S_P\}$$

(*i.e.* for all $Q \in S_P$, we have $I_{f,P}(\mu_i(Q)) = f(Q)$), where μ_i is the function that partitions $\mathcal{L}_{F_i}(G_i)$. For any $j \in \{0, \dots, p_i - 1\}$, we then define the function

$$f_j : \left\{ \begin{array}{ccc} \mathcal{P}_{i-1} & \to & \mathbb{F} \\ P & \mapsto & a_{j,P}. \end{array} \right.$$

By assumption, we have $|\mathcal{P}_{i-1}| = \frac{|\mathcal{P}_i|}{p_i}$, hence the idea is to define p_i functions f_j such that f corresponds to the evaluation of a function in $\mathcal{L}_{F_i}(G_i)$ if and only if each f_j coincides with a function in $\mathcal{L}_{F_{i-1}}(E_{i,j}) \subset \mathcal{L}_{F_{i-1}}(G_{i-1})$. Now, instead of testing whether $f_j \in \mathcal{C}_{i-1}$ for all j, we reduce those claims into a single one by taking a random linear combination of f_j's, referred to as the *folding of f*. At this point, note that for soundness analysis, the introduction of the balancing functions $\nu_{i-1,j}$ in the definition of compatible divisors guaranties that no f_j corresponds to a function lying in $\mathcal{L}_{F_{i-1}}(G_{i-1}) \backslash \mathcal{L}_{F_{i-1}}(E_{i,j})$. This explains why the folding operation takes it into account.

Definition 5.16 (Folding operator)**.** For any $\mathbf{z} = (z_1, z_2) \in \mathbb{F}^2$, we define the *folding* of f as the function $\mathbf{Fold}[f, \mathbf{z}] : \mathcal{P}_{i-1} \to \mathbb{F}$ such that

$$\mathbf{Fold}[f, \mathbf{z}] := \sum_{j=0}^{p_i-1} z_1^j f_j + \sum_{j=0}^{p_i-1} z_2^{j+1} \nu_{i-1,j} f_j.$$

Properties of the folding operator. In this paragraph, we give without proofs three key properties satisfied by our folding operator. This will, using [ABN22, Theorem 1], prove the completeness and the soundness of our AG–IOPP. The proofs of the upcoming results can be found in [BNLR22], Section 7.2.

Proposition 5.17. *The folding operator defined above satisfies the following properties:*

1. ***Locality.*** *Let $\mathbf{z} \in \mathbb{F}^2$. Then for each $P \in \mathcal{P}_{i-1}$, the value of $\mathbf{Fold}[f, \mathbf{z}](P)$ can be computed with exactly p_i queries to the function f, namely at the points $\pi_i^{-1}(\{P\})$.*
2. ***Completeness.*** *Let $\mathbf{z} \in \mathbb{F}^2$. If $f \in \mathcal{C}_i$, then $\mathbf{Fold}[f, \mathbf{z}] \in \mathcal{C}_{i-1}$.*
3. ***Distance preservation.*** *For $\delta > 0$, if f is δ–far from $\mathcal{C}_i$ (i.e. $\Delta(f, \mathcal{C}_i) > \delta$), then its folding $\mathbf{Fold}[f, \mathbf{z}]$ is δ–far from $\mathcal{C}_{i-1}$ with high probability on $\mathbf{z} \in \mathbb{F}^2$.*

5.4 AG–IOPP on the Hermitian tower

Given a family $(\mathcal{C}_i)_{i=0}^r = (C_{\mathcal{L}}(\mathcal{X}_i, \mathcal{P}_i, G_i))_{i=0}^r$ of foldable AG codes over a finite field $\mathbb{F}$, we informally describe the IOPP system (P,V) for testing proximity of a function $f^{(r)} : \mathcal{P}_r \to \mathbb{F}$ to the code $\mathcal{C}_r$ in a general case, before studying its properties in the case of the family of codes along the Hermitian tower given in Proposition 5.15. Again, proofs and details can be found in [BNLR22], Section 8.

5.4.1 Description of the AG–IOPP system

Our IOPP system is divided into two phases, referred to as COMMIT and QUERY phase. Before any interaction, both the verifier V and the prover P agree on the sequence of foldable codes $(\mathcal{C}_i)_{i=0}^r$ satisfying Definition 5.5 (that is, for all $i \in \{1, \dots, r\}$, they agree on: the curve $\mathcal{X}_i$, support $\mathcal{P}_i$, divisor G_i, function μ_i and balancing functions $\nu_{i-1,j}$ for $j \in \{0, \dots, p_i - 1\}$).

COMMIT phase. This first phase consists in an interaction over r rounds between the prover P and the verifier V. For each round $i \in \{1, \dots, r\}$, the verifier samples a random *challenge* $\mathbf{z}^{(i)} = (z_1^{(i)}, z_2^{(i)}) \in \mathbb{F}^2$. As an answer, the prover gives oracle access to a function $f^{(i-1)} : \mathcal{P}_{i-1} \to \mathbb{F}$, which is expected to be equal to $\mathbf{Fold}\left[f^{(i)}, \mathbf{z}^{(i)}\right]$. To compute the values of $f^{(i-1)}$ on the set $\mathcal{P}_{i-1}$, an honest prover exploits the fact that the folding $f^{(i-1)}$ of $f^{(i)}$ is *locally computable* (see Proposition 5.17, *1*). This phase ends up with P sending a final function $f^{(0)} : \mathcal{P}_0 \to \mathbb{F}$.

The COMMIT phase is depicted in Figure 5.1. For both this figure and Figure 5.2, full lines means that the data can be seen in full, while dotted lines correspond to an oracle access.

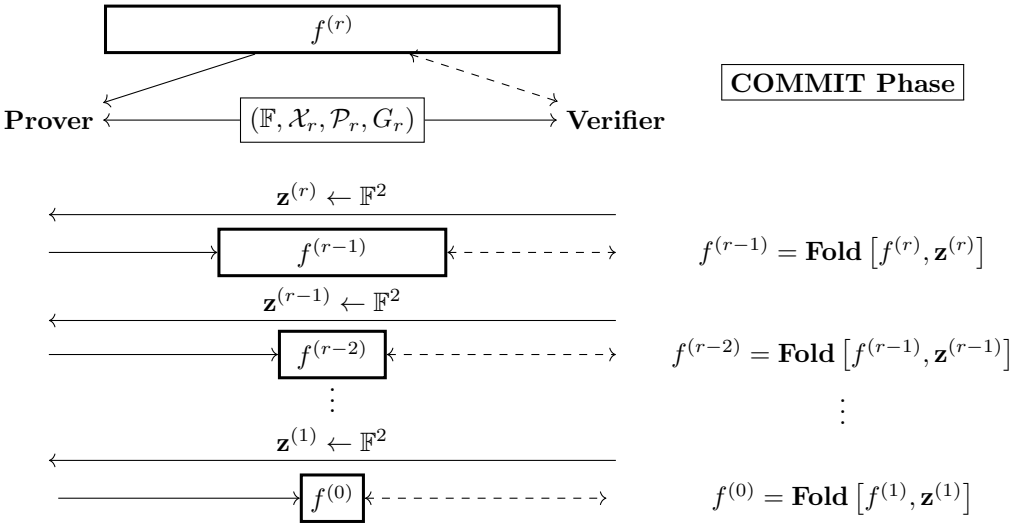

Figure 5.1: AG–IOPP : COMMIT phase

QUERY phase. During the QUERY step, the task of the verifier is to check that each consecutive pair of oracle functions $\left(f^{(i)}, f^{(i-1)}\right)$ is consistent, *i.e.* that $f^{(i-1)}$ is indeed constructed as the folding of $f^{(i)}$. More precisely, the idea is to check that the equality

$$f^{(i-1)}(P) = \mathbf{Fold}\left[f^{(i)}, \mathbf{z}^{(i)}\right](P) \tag{5.12}$$

holds at a random point $P \in \mathcal{P}_{i-1}$. Again, the local property of the folding operator ensures that each test only requires p_i queries to $f^{(i)}$ and one to $f^{(i-1)}$. This verification test is referred to as *round consistency test*. The set of points in which Equation (5.12) has to be checked is chosen by V in the following way: at the beginning, a random $Q_r \in \mathcal{P}_r$ is sampled. Then, for each round $i \in \{1, \dots, r\}$, V computes the next location test as $Q_{i-1} := \pi_i(Q_i)$. The set $\{Q_1, \dots, Q_r\}$ is called *query path*. Note that the correlation between the round consistency tests allows to improve the soundness of the IOPP.

For the *final test*, V reads $f^{(0)} : \mathcal{P}_0 \to \mathbb{F}$ entirely to decide if it belongs to $C_{\mathcal{L}}(\mathcal{X}_0, \mathcal{P}_0, G_0)$ or not. The QUERY phase of the protocol can be repeated several times before the final decision of the verifier, hence improving the soundness error. Picture 5.2 sums up this discussion.

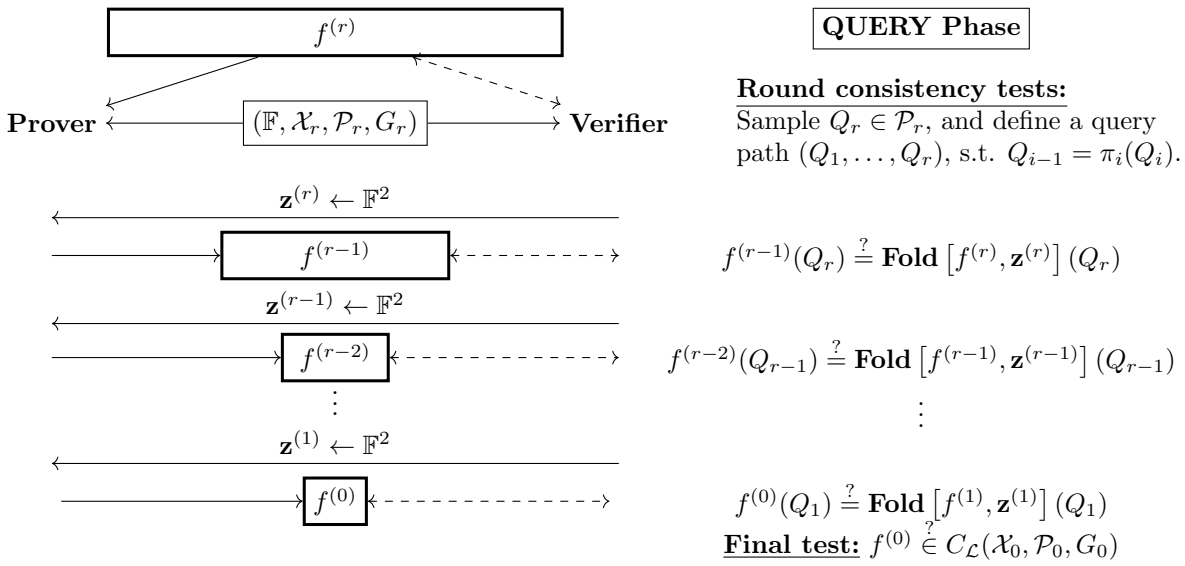

Figure 5.2: AG–IOPP : QUERY phase

5.4.2 Properties of the AG–IOPP with the Hermitian tower

Here, we study the properties of the AG IOPP system described in the previous section, in the case of the family of foldable codes along the Hermitian tower given in Section 5.2.2. All is summarized in the informal theorem below.

Theorem 5.18 ([BNLR22, Theorem 45])**.** *Let $i_{\max} \geq 0$ and consider a family of foldable AG codes $(\mathcal{C}_i)_{i=0}^{i_{\max}} = \left(C_{\mathcal{L}}\left(\mathcal{X}_i, \mathcal{P}_i, d_i P_\infty^{(i)}\right)\right)_{i=0}^{i_{\max}}$ as in Proposition 5.15. The length $n = \#\mathcal{P}_{i_{\max}}$ of $\mathcal{C}_{i_{\max}}$ is at most $q^{i_{\max}+2}$. Then the* IOPP *system described in Section 5.4.1 is an $i_{\max}$–round interactive proof with the following properties:*

1. *Perfect completeness: If $f^{(i_{\max})} \in \mathcal{C}_{i_{\max}}$ and $f^{(i_{\max}-1)}, \ldots, f^{(0)}$ are honestly generated by the prover, then* V *accepts the proof with probability 1.*
2. *Soundness: Assume that $f^{(i_{\max})}$ is δ–far from $\mathcal{C}_{i_{\max}}$. Then for any prover* P* *(possibly malicious), we have*
$$\Pr\left[\left\langle \mathrm{P}^* \leftrightarrow \mathrm{V}^{f^{(i_{\max})}}(\mathcal{C}_{i_{\max}})\right\rangle = \text{accept}\right] \leq \mathrm{err}(\delta),$$
where $\mathrm{err}(\delta)$ *is small and depends on both the error during the* COMMIT *phase and the ones due to the t iterations of the* QUERY *phase.*

Moreover, we have:

rounds complexity	$r(n) < \log(n)$
proof length	$\ell(n) < n$
query complexity	$q(n) \leq tq\log(n) + 1$
prover complexity	$t_p(n) = \mathcal{O}\left(n \cdot \mathrm{M}_{\mathbb{F}_{q^2}}(q)\log(q)\right)$
verifier complexity	$t_v(n) = \mathcal{O}\left(\log(n) \cdot \mathrm{M}_{\mathbb{F}_{q^2}}(q)\log(q)\right)$,

where $\mathrm{M}_{\mathbb{F}_{q^2}}(d)$ denotes the cost of multiplying two degree-d univariate polynomials over $\mathbb{F}_{q^2}$.

After the interaction between P and V, we end up with a proximity test of a function $f^{(0)}$ to an AG code $\mathcal{C}_0 := C_{\mathcal{L}}(\mathcal{X}_0, \mathcal{P}_0, d_0 P_\infty^{(0)})$. As $\mathcal{X}_0$ equals the projective line, the code $\mathcal{C}_0$ is nothing but a Reed–Solomon code. Taking this into consideration, we can replace the *final test* of our AG–IOPP system with a proximity test to a RS code (*i.e.* a FRI protocol, see Section 2.2.3), until we get a dimension one RS code.

As already discussed at the end of section 5.2.2, we need to ensure that the RS code $\mathcal{C}_0$ is not trivial, *i.e.* its rate is smaller than 1. In [BNLR22, Section 8.3.2], this is examined by considering the foldable code defined in Equation (5.11), that is

$$\mathcal{C}_\alpha := \mathcal{C}_{\mathcal{L}}\left(\mathcal{X}_{i_{\max}}, \mathcal{X}_{i_{\max}}(\mathbb{F}_{q^2}) \setminus \left\{P_\infty^{(i_{\max})}\right\}, (2\alpha+1)\mathfrak{g}_{i_{\max}} P_\infty^{(i_{\max})}\right), \text{ with } \alpha > \frac{1}{2}.$$

We then provide a sufficient condition on α and $i_{\max}$ to bound from above the rate of $\mathcal{C}_0$, as well as the following examples:

q	$i_{\max}$	n	$R_{i_{\max}}$	$1-\rho >$
2^4	3	2^{20}	1/8	1/3
2^5	5	2^{35}		
2^4	4	2^{24}	1/16	1/3
2^5	3	2^{25}		3/4
	5	2^{35}		1/2
2^6	4	2^{36}		3/4
	5	2^{42}		2/3
	7	2^{54}		1/2
2^4	3	2^{20}	1/32	1/2

Table 5.1: Example of parameters of foldable codes of rate R along the Hermitian tower.

In Table 5.1, we display some examples of initial level $i_{\max}$ and initial rate $R_{i_{\max}}$ of $\mathcal{C}_{i_{\max}}$ for which the AG-IOPP reduces to the proximity test of the RS code $\mathcal{C}_0$ of rate ρ. In particular, the lower bound on $1-\rho$ provides an estimation of the minimum distance of $\mathcal{C}_0$.

Conclusion

For the three years that ended with this manuscript, we focused on how we could use algebraic geometry codes in post–quantum cryptography. More precisely, we worked in two different directions: the first consists in studying AG code-based cryptosystems (such as McEliece's encryption scheme), by either proposing new ones or analyzing the security of existing ones. The second way to use this family of codes is when constructing new and efficient proof systems. It is now time to sum up our contributions and present some perspectives.

In the case of McEliece's encryption scheme based on structured SSAG codes, we provided a security reduction of the corresponding secret key. Currently, the technique introduced can be applied whenever the public SSAG code is quasi–cyclic, either built on a Kummer cover or an elementary abelian p–cover of curves. With some additional work, we hope that this security reduction could be generalized to a solvable Galois cover of curves, at least under some technical assumptions. Moreover, some needed hypotheses could be weakened. The consequence of this work is that cautions have to be taken while constructing McEliece's scheme based on these codes in order to guarantee a good security level.

With the desire to generalize the construction of the distinguisher for alternant and classical Goppa codes proposed in [MT21], we define a new class of SSAG codes whose structure mimics theirs: Goppa–like AG codes. After successfully adapting the techniques of [MT21] to build a distinguisher, we study different sets of parameters for Goppa–like AG codes. The specific case of one–point Goppa–like AG codes constructed on the Hermitian curve might be interesting for designing efficient SSAG–based cryptosystems for the following reasons:

- they can be encoded efficiently, as the evaluation space is well–known;
- they are resistant to our distinguisher;
- for similar parameters, they improve key sizes compared with binary Goppa codes.

The remaining work of this thesis focuses on using AG codes while designing efficient proof systems, especially in the context of proximity tests to a linear code. Recognizing the effectiveness of the Reed–Solomon–based FRI protocol, a first AG codes-based IOPP has been proposed in [BN20]. Following this work, we propose another protocol, this time relying on codes constructed on the Hermitian tower, with the aim of correcting problems imposed by the choice of Reed–Solomon codes. In fact, considering recursive towers enables us to construct an IOPP on a polylogarithmic-size alphabet. Concerning efficiency, our concrete instance reaches quasilinear prover time and polylogarithmic verification.

As a conclusion, we point out perspectives motivated by our work: both Chapters 3 and 4 provide new cryptanalysis tools in the context of McEliece's encryption schemes, that might be useful for future constructions. The use of AG codes in proximity tests being now initiated, there are probably lots of ways to improve it: a first idea could be to consider codes constructed from an optimal tower of curves (Appendix B goes in that direction). Observing that the area of code–based cryptography is constantly in progress nowadays, we hope that our contributions could lead to significant improvements in the future.

Appendix A

Algorithm for retrieving the equation of a cover

In this appendix, we present a formal algorithm that describes the attack proposed in Section 3.3.2, in the context of a Kummer cover of curves $\mathcal{Y} \to \mathcal{X}$, where $(\mathcal{X}, P_\infty) \in \mathcal{B}$ (the class $\mathcal{B}$ is defined in Section 3.3.1). Keeping the same notation, recall that we aim to recover the evaluation vector $\mathbf{y} = y(Q_{i,j})_{i,j}$. For $\xi \in \mu_\ell^*(\mathbb{F}_{q^m})$, we are led to solve linear systems of the form

$$\Delta_1(\xi) \begin{cases} \mathbf{E}(\xi^2) \cdot (\mathbf{z}_\omega \star \mathbf{y})^T &= 0 \\ \mathbf{M} \cdot \mathbf{D}_1 \cdot (\mathbf{z}_\omega \star \mathbf{y})^T &= 0 \\ \vdots & \\ \mathbf{M} \cdot \mathbf{D}_s \cdot (\mathbf{z}_\omega \star \mathbf{y})^T &= 0 \end{cases} \quad \text{and} \quad \Delta_2(\xi) \begin{cases} \mathbf{E}(\xi^3) \cdot (\mathbf{z}_\omega \star \mathbf{y}^{\mathbf{2}})^T &= 0 \\ \mathbf{M} \cdot \mathbf{D}_1 \cdot (\mathbf{z}_\omega \star \mathbf{y}^{\mathbf{2}})^T &= 0 \\ \vdots & \\ \mathbf{M} \cdot \mathbf{D}_s \cdot (\mathbf{z}_\omega \star \mathbf{y}^{\mathbf{2}})^T &= 0. \end{cases}$$

In this framework, we describe in Algorithm 1 the attack of Section 3.3.2 in the context of a McEliece cryptosystem based on a quasi–cyclic SSAG code defined on $\mathcal{Y}$. For simplicity, we write $K = \mathbb{F}_{q^m}(\mathcal{X}) = \mathbb{F}_{q^m}(x, z)$, which is a degree a extension of the rational function field. Hence, any degree one place $P \in \mathcal{P}$ can be associated to the rational point in $\mathcal{X}$ with projective coordinates $[x(P) : z(P) : 1]$. Given $Q \in \mathcal{Q}$ such that $Q|P$, the representative point of Q on $\mathcal{Y}$ as projective coordinates $[x(P) : z(P) : y(Q) : 1]$ in $\mathbb{P}^3$.

A full complexity analysis of Algorithm 1 below is hard to estimate because of the interpolation step (see the discussion below Proposition 3.9). For completeness, we still provide a complexity analysis when $\mathcal{X} = \mathbb{P}^1$, since in this case we can use the classical Lagrange's interpolation method to recover f. Remark that this specific case coincides with the one considered in [Bar18a, Chapter 5].

Proposition A.1. *Suppose $K = \mathbb{F}_{q^m}(x)$ is the rational function field and consider $L = K(y)$, with*

$$y^\ell = f,$$

where $f \in \mathbb{F}_{q^m}[T]$ is a square–free polynomial of degree d. Let n, k be the length and the dimension of the public SSAG respectively, $r := n/\ell$ be the number of orbits in $\mathcal{Q}$ and $s := \ell(D - B)$. If $r \geq d + 1$, then Algorithm 1 finds an equation of $\mathcal{Y}$, as well as the secret structure of the public code in $\mathcal{O}((\varphi(\ell) + 1)(n^\omega + n^{\omega-1}sk)$ operations over $\mathbb{F}_{q^m}$, where ω is the exponent of linear algebra.

Proof. The complexity of solving a linear system with k equations and n unknowns is in $\mathcal{O}(n^{\omega-1}k)$ operations over the base field, where ω is the exponent of linear algebra. Since both $\Delta_1(\xi)$ and $\Delta_2(\xi)$ consist in $sk+n$ equations for n unknowns, the cost of line 10 is $\mathcal{O}(n^\omega + n^{\omega-1}sk)$ operations over $\mathbb{F}_{q^m}$ (operations need to be done in $\mathbb{F}_{q^m}$ and not $\mathbb{F}_q$ as the roots of unity might be in $\mathbb{F}_{q^m} \backslash \mathbb{F}_q$). Since we have to seek for the correct root of unity ξ, this step might be repeated at most $\varphi(\ell)$–times, where φ is the Euler totient function. After solving the first system, we can assume that we recovered the correct value for ξ and thus the second system only has to be resolved once, with the same complexity.

Next, we have to realize one Lagrange's interpolation at line 18 (which is the classical one in this case, since f is a one variable polynomial) in order to recover a defining equation of the Kummer cover. As we assumed $r \leq d + 1$, Lagrange's interpolation finds a unique polynomial f of degree d such that a plane model of $\mathcal{Y}$ is given by $y^\ell = f(x)$ in $\mathcal{O}(d^2)$ operations over $\mathbb{F}_{q^m}$ (which is negligible compared to the cost of solving the linear systems).

Finally, the last step we have to care about is at line 20, where we are left to compute the pullback of the invariant divisor. As for the support, we need to recover the y–coordinates of points

in Supp(G). This can be done by finding roots of several polynomials: indeed, from Kummer's theorem [Sti09, Theorem 3.3.7], if $x(Q)$ denotes the x–coordinate of a point $R \in \mathrm{Supp}(\widetilde{G})$, then the y–coordinates of the extensions of P in Supp(G) are exactly the roots of the polynomial $T^\ell - f(x(Q)) \in \mathbb{F}_{q^m}[T]$. This step can be done by factorizing each polynomial using Berlekamp algorithm, whose cost is $\mathcal{O}(\ell^\omega + q^m \ell^2)$ operations over $\mathbb{F}_{q^m}$. In any practical cases, the length of the public code is larger that the cardinality of the base field (*i.e.* $n > q^m$) and thus this step is also negligible. As a result, the total cost of Algorithm 1 is in $\mathcal{O}((\varphi(\ell)+1)(n^\omega + n^{\omega-1}sk)$ over $\mathbb{F}_{q^m}$. □

Algorithm 1: Security reduction in Kummer cover

Input : A generator matrix $\mathbf{M}_{\mathrm{pub}}$ of the public SSAG code, $\mathcal{P} = \{P_1, \cdots, P_r\}$ and $\widetilde{G}$. The integers ℓ and d.

Output: A function $f \in K$ and the secret structure $(\mathcal{Q}, G)$.

$\mathbf{x} \longleftarrow (\underbrace{x(P_1), ..., x(P_1)}_{\ell-\text{times}}, ..., z(P_r), ..., z(P_r))$

$\mathbf{z} \longleftarrow (\underbrace{z(P_1), ..., z(P_1)}_{\ell-\text{times}}, ..., z(P_r), ..., z(P_r))$

$D \longleftarrow \left\lfloor \frac{(\ell-1)(d-1) - 2 + \ell a r^*}{\ell} \right\rfloor P_\infty - \widetilde{G} - A$

$B \longleftarrow \left\lceil \frac{a}{\ell} \right\rceil P_\infty$

$M \longleftarrow$ set of primitive ℓ–th roots of unity in $\mathbb{F}_{q^m}$

temp := 0

while temp = 0 **do**

  $\xi \xleftarrow{\$} M$

  Exclude(M,ξ)

  $\mathcal{S}_1 \longleftarrow$ Solve($\Delta_1(\xi)$)

  if $\dim(\mathcal{S}_1) = 1$ **then**

    $\mathcal{S}_2 \longleftarrow$ Solve($\Delta_2(\xi)$) `// ξ is now known`

    if $\dim(\mathcal{S}_2) = 1$ **then**

      temp := 1

      $\mathbf{y}_1 \xleftarrow{\$} \mathcal{S}_1 \backslash \{0\}$

      $\mathbf{y}_2 \xleftarrow{\$} \mathcal{S}_2 \backslash \{0\}$

      $\mathbf{y}^* \leftarrow \mathbf{y}_2 \star \mathbf{y}_1^{-1}$

      $f \longleftarrow$ Interpolate($\mathbf{x}, \mathbf{z}, \mathbf{y}^*$)

      $\mathcal{Q} \longleftarrow \{Q_{i,j} = (x_{i,j} : z_{i,j} : y^*_{i,j} : 1)\}$

$G \longleftarrow \pi^*(\widetilde{G})$

return f, $\mathcal{Q}$ and G

Note that Algorithm 1 can also be used in the case of an elementary abelian p–extension of $\mathbb{F}_{q^m}(x)$, by only changing a few lines: in fact, in the latter setup, we have to solve at most $p^u = \#\,\mathrm{Aut}(L/\mathbb{F}_{q^m}(x))$ linear systems of the form (3.16) ($p^u := [L : \mathbb{F}_{q^m}(x)]$), whose total cost is in $\mathcal{O}(p^u(n^\omega + n^{\omega-1}sk))$ operations over $\mathbb{F}_{q^m}$.

Appendix B

Foldable AG codes from a tower of modular curves

In the discussion below, we give valid setting to construct a family of foldable AG codes defined on an optimal tower of curves over a finite field $\mathbb{F}_q$. Let us recall what we mean by optimal towers: given a sequence of curves $\mathcal{X} = (\mathcal{X}_i)_i$, we define its *limit* by

$$\lambda(\mathcal{X}) = \lim_{i\to\infty} \frac{\#\mathcal{X}_i(\mathbb{F}_q)}{g(\mathcal{X}_i)}.$$

It is well–known from the Drinfeld–Vladut bound [GS07, Theorem 2.5] that for any such $\mathcal{X}$,

$$\lambda(\mathcal{X}) \leq \sqrt{q} - 1.$$

We say that the tower $\mathcal{X}$ is (asymptotically) *optimal* if the bound is attained, *i.e.*

$$\lambda(\mathcal{X}) = \sqrt{q} - 1.$$

Note that there may exists optimal tower over $\mathbb{F}_q$ only if q is a square (in which case there always exists one). More details about optimal towers can be found in [GS07, Chapter 1].

Below, we focus on the optimal tower over a finite field of cardinality $q = p^2$, where p is an odd prime, recursively defined by the equation

$$y^2 = \frac{x^2+1}{2x}.$$

This tower has been proven to be optimal in [GS07, Section 4.3] and also *modular* [Elk01], as it comes from the modular tower $(X_0^{2^n})_{n\geq 0}$. Optimal towers are good candidates to design long AG codes since they asymptotically have the maximal number of rational points for a given base field, with respect to the Drinfeld–Vladut bound. Below, we keep notation as in Chapter 5.

B.1 Preliminaries

Let $q = p^2$ be a power of an odd prime p.

Sequence of curves and automorphisms. We consider the tower of function field $\mathcal{F} = (F_i)_{i\geq 0}$ over $\mathbb{F}_q$, recursively defined by $F_0 = \mathbb{F}_q(x_0)$ and $F_i = F_{i-1}(x_i)$, where

$$x_i^2 = \frac{x_{i-1}^2+1}{2x_{i-1}} \ , \ \forall i \geq 1. \tag{B.1}$$

For any $i \geq 0$, we denote by $\mathcal{X}_i$ the curve over $\mathbb{F}_q$ with function field $F_i = \mathbb{F}_q(\mathcal{X}_i)$, and by $\mathfrak{g}_i$ its genus. We obtain an infinite tower of curves as in Equation 5.6, where $\Gamma_i = \mathbb{Z}/2\mathbb{Z}$ and $\pi_i : \mathcal{X}_i \to \mathcal{X}_{i-1}$ is the projection map. We also set $\Pi_i := \pi_1 \circ \cdots \circ \pi_{i-1} \circ \pi_i$ the composition of the first i quotient maps.

Ramification and genus. Note that every extension F_i/F_{i-1} is a Kummer extension of degree 2, meaning that to obtain a good decomposition for Riemann–Roch spaces (as in Equation (5.3)) along this tower, we can use the same tool as in [BNLR22] for the case of foldable codes over Kummer type curves, which is Maharaj's theorem [Mah04, Theorem 2.2]. To apply it efficiently, we first need to control the principal divisors of $x_i \in F_i$ for any $i \geq 0$. Thankfully, a lot of work has been done in [NOQ11] to understand the ramification behaviour in the tower. In their paper, they also provide a genus formula which will be useful at some point, as well as an explicit basis for Riemann–Roch spaces associated to one–point divisors. For the remaining of this section, we keep their notation and recall some of their results.

From now on, any place in F_i will be denoted with an exponent "(i)" to signify that it belongs to the i-th function field F_i is the tower. For each $\alpha \in \mathbb{F}_q \cup \{\infty\}$, we denote by $P_\alpha^{(0)}$ the unique zero of $x_0 - \alpha$ in the rational function field $F_0 = \mathbb{F}_q(x_0)$, and we consider the set

$$\mathcal{R} = \left\{ P_0^{(0)}, P_\infty^{(0)}, P_{\pm 1}^{(0)}, P_{\pm i}^{(0)} \right\} \subseteq \mathbb{P}_{F_0}.$$

Lemma B.1 ([NOQ11, Lemma 2.2])**.** *Let $i \geq 1$. Then*

1. *The places $P_0^{(0)}, P_\infty^{(0)}, P_i^{(0)}$ and $P_{-i}^{(0)}$ are totally ramified in F_i/F_0. If $P_0^{(i)}, P_\infty^{(i)}, P_i^{(i)}, P_{-i}^{(i)}$ denote their unique extension, we have:*

$$\nu_{P_\infty^{(i)}}(x_i) = -1, \quad \nu_{P_0^{(i)}}(x_i) = \begin{cases} 1\ , & i = 0 \\ -1\ , & i \geq 1 \end{cases} \quad \text{and} \quad \nu_{P_{\pm i}^{(i)}}(x_i) = \begin{cases} 0\ , & i = 0 \\ 1\ , & i = 1 \\ -1\ , & i \geq 2; \end{cases}$$

2. *The place $P_1^{(i-1)}$ totally splits in F_i. Its two extensions $P_1^{(i)}$ and $P_{-1}^{(i)}$ satisfy $x_i(P_1^{(i)}) = 1$ and $x_i(P_{-1}^{(i)}) = -1$.*

3. *$P_{-1}^{(i-1)}$ totally splits in F_i; its two extensions $Q_i^{(i)}$ and $Q_{-i}^{(i)}$ are the only zeros of the function $1 + x_i^2 \in F_i$, and*

$$\nu_{Q_{\pm i}^{(i)}}(1 + x_i^2) = 2^i;$$

4. *Let $0 \leq r \leq i$, and denote by $Q_r^{(i)} | P_{-1}^{(r)}$ an extension of $P_{-1}^{(r)}$ in F_i.*

   (i) If $0 \leq r \leq \lfloor \frac{i-3}{2} \rfloor$, then $Q_r^{(i)} | Q_r^{(i-1)}$ with $e(Q_r^{(i)} | Q_r^{(i-1)}) = 2$ and $\nu_{Q_r^{(i)}}(x_i) = -1$. Moreover, the sum of the degree of such places equals 2^{r+2}.

   (ii) If $\lfloor \frac{i-1}{2} \rfloor \leq r \leq i-2$, then $Q_r^{(i)} | Q_r^{(i-1)}$ with $e(Q_r^{(i)} | Q_r^{(i-1)}) = 1$ and

$$\nu_{Q_r^{(i)}}(x_i) = \begin{cases} -2^{2r+2-i}, & r \leq i-3 \\ -2^{i-2}, & r = i-2. \end{cases}$$

   The sum of the degree of such places equals 2^{i-r}.

It turns out that all places described in Lemma B.1 are all the places above the set $\mathcal{R}$ in $\mathcal{F}$, and more importantly; $(x_i)^{F_i}$ is only supported by these places. Except for $Q_{\pm i}^{(0)} = P_{\pm i}^{(0)}$, all places considered above are distinct.

Definition B.2. For any $i \geq 0$ and $0 \leq r \leq i$, we define

$$D_r^{(i)} := \sum_{Q_r^{(i)} | P_{-1}^{(r)}} Q_r^{(i)} \in \mathrm{Div}(F_i).$$

We have $D_i^{(i)} = P_{-1}^{(i)}$, and we can extend this definition for $r = -2$, and -1 by setting

$$D_{-2}^{(i)} := P_0^{(i)}$$

and

$$D_{-1}^{(i)} := P_i^{(i)} + P_{-i}^{(i)}.$$

Corollary B.3. *1. For any $i \geq 0$ and $-2 \leq r \leq i$, we have*

$$\deg\left(D_r^{(i)}\right) = \begin{cases} 2^{i-r}, & \text{if } i \leq 2r+2 \\ 2^{r+2}, & \text{if } i \geq 2r+2. \end{cases}$$

2. *If $i \geq 1$ and $-2 \leq r \leq i$, then a place in* $\mathrm{Supp}\left(D_r^{(i)}\right)$ *ramifies in F_i/F_{i-1} if and only if $i \geq 2r+3$.*

Proof. Immediate consequence of Lemma B.1 *4* (see also [NOQ11, Corollary 2.4]). □

The ramification of the places $P_{-1}^{(r)}$ in the tower being somewhat technical, you can find it for the first stages of the tower in Figure B.1.

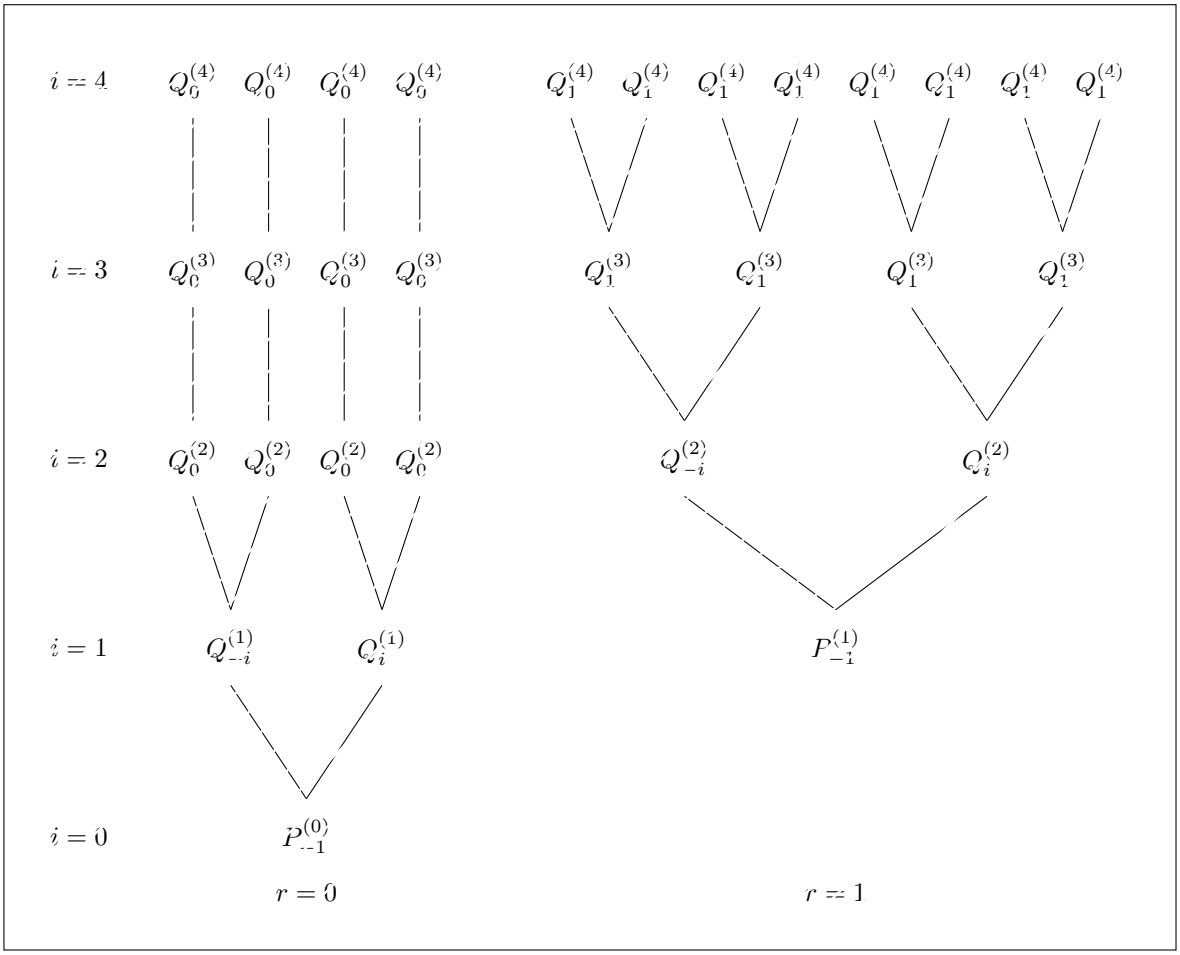

Figure B.1: Ramification of $P_{-1}^{(r)}$ for $r \in \{0,1\}$ and $i \leq 4$

We now have all the ingredients we need to explicit the principal divisors we will need later on.

Proposition B.4. *1. We have $(x_0)^{F_0} = P_0^{(0)} - P_\infty^{(0)}$, $(x_1)^{F_1} = D_{-1}^{(1)} - P_0^{(1)} - P_\infty^{(1)}$ and for any $i \geq 2$:*

$$(x_i)^{F_i} = 2^{i-2} D_{i-2}^{(i)} - P_\infty^{(i)} - \sum_{r=-2}^{\lfloor \frac{i-3}{2} \rfloor} D_r^{(i)} - \sum_{\lfloor \frac{i-1}{2} \rfloor}^{i-3} 2^{2r-i+2} D_r^{(i)};$$

2. We have $(1+x_0)^{F_0} = P_{-1}^{(0)} - P_\infty^{(0)}$, $(1+x_1)^{F_1} = 2P_{-1}^{(1)} - P_0^{(1)} - P_\infty^{(1)}$ and for any $i \geq 2$:

$$(1+x_i)^{F_i} = 2^i P_{-1}^{(i)} - P_\infty^{(i)} - \sum_{r=-2}^{\lfloor \frac{i-3}{2} \rfloor} D_r^{(i)} - \sum_{\lfloor \frac{i-1}{2} \rfloor}^{i-3} 2^{2r-i+2} D_r^{(i)};$$

3. We have $(1+x_0^2)^{F_0} = D_{-1}^{(0)} - 2P_\infty^{(0)}$, $(1+x_1^2)^{F_1} = 2D_0^{(1)} - 2P_0^{(1)} - 2P_\infty^{(1)}$ and for any $i \geq 2$:

$$(1+x_i^2)^{F_i} = 2^i D_{i-1}^{(i)} - 2P_\infty^{(i)} - 2\sum_{r=-2}^{\lfloor \frac{i-3}{2} \rfloor} D_r^{(i)} - \sum_{\lfloor \frac{i-1}{2} \rfloor}^{i-3} 2^{2r-i+3} D_r^{(i)};$$

Proof. For items *1* and *2*, we refer to [NOQ11, Proposition 2.5]. For *3*, note that the functions x_i and $(1+x_i^2)$ have the same poles, and for any $P^{(i)} \in \mathrm{Supp}(x_i)_\infty^{F_i}$, we have $\nu_{P^{(i)}}(1+x_i^2) = 2 \cdot \nu_{P^{(i)}}(x_i)$. Moreover, $D_{i-1}^{(i)} = Q_i^{(i)} + Q_{-i}^{(i)}$ by definition, so Lemma B.1 *3.* gives the result, since

$$\deg((1+x_i^2)_0^{F_i}) = 2[F_i : F_0] = 2^{i+1} = 2^i \deg\left(D_{i-1}^{(i)}\right).$$

□

We conclude this preliminary section with a genus formula for each function field in the tower $\mathcal{F}$.

Proposition B.5 ([NOQ11, Proposition 2.6])**.** *For $i \geq 0$, the genus $\mathfrak{g}_i$ of F_i is given by*

$$\begin{aligned}\mathfrak{g}_i &= \begin{cases} \left(2^{\frac{i+2}{2}} - 1\right)\left(2^{\frac{i}{2}} - 1\right), & \text{if } i \equiv 0[2] \\ \left(2^{\frac{i+1}{2}} - 1\right)^2, & \text{if } i \equiv 1[2]. \end{cases} \\ &= \begin{cases} 2^{i+1} - 3 \cdot 2^{\frac{i}{2}} + 1, & \text{if } i \equiv 0[2] \\ 2^{i+1} - 2^{\frac{i+3}{2}} + 1, & \text{if } i \equiv 1[2]. \end{cases}\end{aligned}$$

B.2 Towards foldable AG codes

As in the case of the Hermitian tower (see Section 5.2), we want to define a sequence of foldable AG codes along the tower $\mathcal{F}$. In [NOQ11], the authors give an explicit way to split Riemann–Roch spaces associated to the one–point divisor $sP_\infty^{(i)} \in \mathrm{Div}(\mathcal{X}_i)$, using Riemann–Roch spaces on $\mathcal{X}_{i-1}$. This motivates the use of these one–point divisors to define our sequence of codes.

For any fixed $i \geq 1$, the group $\mathcal{G}_i = \mathbb{Z}/2^i\mathbb{Z}$ acts on $\mathcal{X}_i$ and the quotient curve $\mathcal{X}_i/\mathcal{G}_i$ is equal to the projective line $\mathbb{P}^1$. We want to deal with an AG code on $\mathcal{X}_i$ of the form

$$C_{\mathcal{L}}\left(\mathcal{X}_i, \mathcal{P}_i, G_i\right),$$

where $\mathcal{P}_i \subseteq \mathcal{X}_i(\mathbb{F}_{q^2}) \backslash \left\{P_\infty^{(i)}\right\}$ is a support made of distinct orbits of size 2^i and $G_i = d_i P_\infty^{(i)}$ for some $d_i \geq 0$. Since $P_\infty^{(i)}$ is totally ramified in the tower (see Lemma B.1, *1.*), the divisor G_i is also $\mathcal{G}_i$–invariant.

Splitting of Riemann–Roch spaces. The following theorem gives the desired decomposition of the Riemann–Roch space associated to the one–point divisor $G_i = d_i P_\infty^{(i)}$. In particular, it implies that each $x_i \in F_i$ partitions $\mathcal{L}_{F_i}(G_i)$ with respect to Definition 5.2.

Theorem B.6. *Let $i \geq 1$ and $G_i = d_i P_\infty^{(i)} \in \mathrm{Div}(F_i)$, for some $d_i \geq 0$. Then*

$$\mathcal{L}_{F_i}(G_i) = \pi_{i-1}^*\left(\mathcal{L}_{F_{i-1}}(E_{i,0})\right) \oplus x_i \pi_{i-1}^*\left(\mathcal{L}_{F_{i-1}}(E_{i,1})\right),$$

where

$$E_{i,0} = \left\lfloor \frac{d_i}{2} \right\rfloor P_\infty^{(i-1)}$$

and

$$E_{i,1} = 2^{i-2} D_{i-2}^{(i-1)} + \left\lfloor \frac{d_i - 1}{2} \right\rfloor P_\infty^{(i-1)} - \sum_{r=-2}^{\lfloor \frac{i-3}{2} \rfloor} D_r^{(i-1)} - \sum_{\lfloor \frac{i-1}{2} \rfloor}^{i-3} 2^{2r-i+2} D_r^{(i-1)} \text{ if } i \geq 2.$$

The formula for $E_{1,1}$ is a bit different, i.e.

$$E_{1,1} = D_{-1}^{(0)} + \left\lfloor \frac{d_1 - 1}{2} \right\rfloor P_\infty^{(0)} - P_0^{(0)}.$$

Proof. Consequence of [Mah04, Theorem 2.2] applied to the Kummer extension F_i/F_{i-1}. The structure of $E_{i,1}$ follows from the formula for $(x_i)^{F_i}$ (see Proposition B.4, *1.*) and the ramification behaviour given in Lemma B.1. □

Weak compatibility. Fix $i \geq 1$. Since $P_\infty^{(i)}$ is not the only zero of $E_{i,1}$, we cannot consider a one–point divisor of the form $G_i = d_i P_\infty^{(i)}$ while satisfying $G_{i-1} \geq E_{i,1}$, which is needed for G_{i-1} to be (G_i, x_i)–compatible (see Definition 5.4). To overcome this problem, we introduce a weaker version of compatibility, referred to as *weak-compatibility*. Keep in mind that this new definition implies slight changes to the folding operator defined in Definition 5.16.

Definition B.7 (Weak compatibility)**.** Let $i \geq 1$, and take the general framework of Section 5.1.2, *i.e.* F_i/F_{i-1} is cyclic of prime order p_i. Let $G_i \in \mathrm{Div}(F_i)$ and $\mu_i \in F_i$ be a function that partitions $\mathcal{L}_{F_i}(G_i)$ with respect to Definition 5.2. A divisor $G_{i-1} \in \mathrm{Div}(F_{i-1})$ is said to be *weak* (G_i, μ_i)–*compatible* if there exist functions $f_{i-1,j} \in F_{i-1}$ $(0 \leq j \leq p_i - 1)$ such that

1. For every $j \in \{0, \cdots, p_i - 1\}$, $G_{i-1} + (f_{i-1,j})^{F_{i-1}} \geq E_{i,j}$;
2. For every $j \in \{0, \cdots, p_i - 1\}$, there exists a *weakly balancing* function $\nu_{i-1,j} \in F_{i-1}$ such that
$$(\nu_{i-1,j})_\infty^{F_{i-1}} = G_{i-1} - E_{i,j} + (f_{i-1,j})^{F_{i-1}}.$$

The functions $f_{i-1,j}$ introduced above allows to *compensate* the poles of the divisor $G_{i-1} - E_{i,1}$ without modifying its degree.

Regarding Theorem B.6, we could be tempted to chose
$$G_{i-1} := E_{i,0} = \left\lfloor \frac{d_i}{2} \right\rfloor P_\infty^{(i-1)}.$$
However, the complicated structure of $E_{i,1}$ will lead to an impossibility of finding the weakly balancing function $\nu_{i-1,1}$ if we do so. Actually, the Weierstrass gap theory tells us that such a function does exist only if the divisor $G_{i-1} - E_{i,1}$ has high enough degree, as explained by the following proposition:

Proposition B.8. *Let $D = \sum_{i=1}^{s} n_i P_i$ be a divisor on a given function field F with genus $\mathfrak{g}$ and $1 \leq r \leq s$. If $\deg(D) \geq 2\mathfrak{g}(F) - 1 + r$, there exists a function $h \in F$ such that*
$$\forall 1 \leq i \leq r, \nu_{P_i}(h) = -n_i.$$
In particular, if $r = s$, we have $(h)_\infty = D$.

Proof. Set $D' = D - \sum_{i=1}^{r} P_i = \sum_{i=1}^{r} (n_i - 1)P_i + \sum_{i=r+1}^{s} n_i P_i$. By hypothesis, we have
$$\deg(D') = \deg(D) - r \geq 2\mathfrak{g}(F) - 1.$$
From the Riemann–Roch theorem (Theorem 1.67), this implies that for every $1 \leq i \leq r$, $\ell(D' + P_i) = \ell(D') + 1$. Hence, there exists a function $h_i \in \mathcal{L}(D' + P_i) \backslash \mathcal{L}(D')$, satisfying:

- For each $1 \leq i \leq r$, $\nu_{P_i}(h_i) = -n_i$;
- For each $1 \leq i \neq j \leq r$, $\nu_{P_j}(h_i) \geq \nu_{P_j}(D' + P_i) = -n_j + 1$

Set $h = \sum_{i=1}^{r} h_i \in F$. The strict triangular inequality gives
$$\nu_{P_i}(h) = \min\{\nu_{P_i}(h_i), \nu_{P_i}(h - h_i)\} = -n_i,$$
which yields the desired result. □

In our situation, the above Proposition implies that weakly balancing functions do exist if for any $i \geq 1$ and $j \in \{0, 1\}$, whenever we have
$$\deg\left(G_{i-1} - E_{i,j} + (f_{i-1,j})^{F_{i-1}}\right) \geq 2\mathfrak{g}(F_i) - 1 + \#\,\mathrm{Supp}\left(G_{i-1} - E_{i,j} + (f_{i-1,j})^{F_{i-1}}\right) \tag{B.2}$$
For this reason, and without changing the support of our sequence of divisors, we raise the valuation of G_{i-1} at $P_\infty^{(i-1)}$ at each step. More precisely, we set
$$G_{i-1} = \left(\left\lfloor \frac{d_i}{2} \right\rfloor + \alpha_{i-1}\right) P_\infty^{(i-1)}, \tag{B.3}$$
for some well–chosen $\alpha_{i-1} \geq 0$.

In what follows, we first discuss about the functions $f_{i-1,j}$, before getting back to the choice of the integer α_{i-1} that guarantees the existence of weakly balancing functions.

About the functions $f_{i-1,j}$. With the choice made in Equation (B.3), we have $G_{i-1} - E_{i,0} = \alpha_{i-1} P_\infty^{(i-1)} \geq 0$, meaning that we do not need to find a function $f_{i-1,0}$. However, it is clear that $G_{i-1} - E_{i,1}$ is not effective. Thus, we have to find $f_{i-1,1} \in F_{i-1}$ such that

$$G_{i-1} - E_{i,1} + (f_{i-1,1})^{F_{i-1}} \geq 0.$$

As we want to apply this construction to an effective IOPP, it is desired to have an explicit formula for $f_{i-1,1}$, for every $i \geq 1$. Actually, we show that the set $(f_{i-1,1})_{i\geq 1}$ can be constructed recursively, allowing the prover to precompute it.

For every $i \geq 2$, recall that Theorem B.6 gives

$$G_{i-1} - E_{i,1} = \left(\left\lfloor \frac{d_i}{2} \right\rfloor - \left\lfloor \frac{d_i - 1}{2} \right\rfloor + \alpha_{i-1} \right) P_\infty^{(i-1)} + \sum_{r=-2}^{\lfloor \frac{i-3}{2} \rfloor} D_r^{(i-1)} + \sum_{r=\lfloor \frac{i-1}{2} \rfloor}^{i-3} 2^{2r-i+2} D_r^{(i-1)} - 2^{i-2} D_{i-2}^{(i-1)}. \tag{B.4}$$

Written in this form, we know exactly the pole part of $G_{i-1} - E_{i,1}$, hence we seek for a function whose zero divisor is exactly supported by $D_{i-2}^{(i-1)}$, in order to compensate it. This requires two lemmas.

Lemma B.9. *Let $i \geq 3$. Then*

$$\left(\frac{1 + x_{i-1}^2}{1 + x_{i-2}} \right)^{F_{i-1}} = 2^{i-2} D_{i-2}^{(i-1)} - \begin{cases} 2D_{-1}^{(2)}, \text{ if } i = 3 \\ 2^{i-4} D_{i-4}^{(i-1)} \text{ if } i \geq 4. \end{cases}$$

Proof. The whole proof uses the description of several principal divisors in the tower (see Proposition B.4). For the case $i = 3$, we have

$$\begin{aligned} \left(\frac{1 + x_2^2}{1 + x_1} \right)^{F_2} &= (1 + x_2^2)^{F_2} - (1 + x_1)^{F_1} \\ &= 4D_1^{(2)} - \left(2P_\infty^{(2)} + 2P_0^{(2)} + 2D_{-1}^{(2)} \right) - \pi^* \left(2P_{-1}^{(1)} - P_0^{(1)} - P_\infty^{(1)} \right) \\ &= 4D_1^{(2)} - \left(2P_\infty^{(2)} + 2P_0^{(2)} + 2D_{-1}^{(2)} \right) - \left(2D_1^{(2)} - 2P_0^{[2]} - 2P_\infty^{(2)} \right) \\ &= 2D_1^{(2)} - 2D_{-1}^2. \end{aligned}$$

Now for $i \geq 4$, we have

$$\left(\frac{1 + x_{i-1}^2}{1 + x_{i-2}} \right)^{F_{i-1}} = 2^{i-2} D_{i-2}^{(i-1)} + \left[(x_{i-2})_\infty^{F_{i-1}} - 2(x_{i-1})_\infty^{F_{i-1}} \right],$$

where

$$\begin{aligned} (x_{i-2})_\infty^{F_{i-1}} - 2(x_{i-1})_\infty^{F_{i-1}} = \pi^* &\left(P_\infty^{(i-2)} + \sum_{r=-2}^{\lfloor \frac{i-5}{2} \rfloor} D_r^{(i-2)} + \sum_{\lfloor \frac{i-3}{2} \rfloor}^{i-5} 2^{2r-i+4} D_r^{(i-2)} \right) \\ &- 2 \left(P_\infty^{(i-1)} + \sum_{r=-2}^{\lfloor \frac{i-4}{2} \rfloor} D_r^{(i-1)} + \sum_{\lfloor \frac{i-2}{2} \rfloor}^{i-4} 2^{2r-i+3} D_r^{(i-1)} \right) \end{aligned}$$

From Corollary B.3, any place in $D_r^{(i-2)}$ ramifies in F_{i-1}/F_{i-2} if and only if $i - 1 \geq 2r + 3$, *i.e.* $\left\lfloor \frac{i-4}{2} \right\rfloor \geq r$. Then

$$\begin{aligned} (x_{i-2})_\infty^{F_{i-1}} - 2(x_{i-1})_\infty^{F_{i-1}} &= 2 \sum_{r=-2}^{\lfloor \frac{i-5}{2} \rfloor} D_r^{(i-1)} + \sum_{\lfloor \frac{i-3}{2} \rfloor}^{i-5} 2^{2r-i+4} D_r^{(i-2)} \\ &\quad - 2 \left(\sum_{r=-2}^{\lfloor \frac{i-4}{2} \rfloor} D_r^{(i-1)} + \sum_{\lfloor \frac{i-2}{2} \rfloor}^{i-4} 2^{2r-i+3} D_r^{(i-1)} \right) \\ &= \begin{cases} -2^{2(i-4)-i+4} D_{i-4}^{(i-1)} = -2^{i-4} D_{i-4}^{(i-1)}, \text{ if } i \equiv 1[2] \\ -2D_{\lfloor \frac{i-4}{2} \rfloor}^{(i-1)} + 2 \cdot 2^{2\lfloor \frac{i-3}{2} \rfloor - i + 4} D_{\lfloor \frac{i-3}{2} \rfloor}^{(i-1)} - 2^{i-4} D_{i-4}^{(i-1)}, \text{ if } i \equiv 0[2] \end{cases} \\ &= -2^{i-4} D_{i-4}^{(i-1)}. \end{aligned}$$

The result follows. □

Despite having the good divisor, the above function cannot be chosen as $f_{i-1,1}$ since it would create another pole, but this time with a smaller valuation. Our blessing is that we can raise the valuation at $P_\infty^{(i-1)}$ by eventually taking a bigger α_{i-1}. As a result, finding a function whose only pole is $P_\infty^{(i-1)}$ together with this consideration on α_{i-1} will provide a valid choice for $f_{i-1,1}$. Such a function is given in the next lemma:

Lemma B.10. *Let $i \geq 3$ and set $h_{i-1} = \prod_{j=0}^{i-1} x_j \in F_{i-1}$. Then*

$$(h_{i-1})^{F_{i-1}} = \sum_{r=-2}^{\lfloor \frac{i-3}{2} \rfloor} D_r^{(i-1)} + \sum_{r=\lfloor \frac{i-1}{2} \rfloor}^{i-3} 2^{2r-i+3} D_r^{(i-1)} - (2^i - 1) P_\infty^{(i-1)}.$$

Proof. We proceed by induction on $i \geq 3$. Using Proposition B.4, we have

$$\begin{aligned}(h_2)^{F_2} &= (x_0)^{F_2} + (x_1)^{F_2} + (x_2)^{F_2} \\ &= \left(4D_{-2}^{(2)} - 4P_\infty^{(2)}\right) + \left(2D_{-1}^{(2)} - 2D_{-2}^{(2)} - 2P_\infty^{(2)}\right) + \left(D_0^{(2)} - D_{-2}^{(2)} - D_{-1}^{(2)} - P_\infty^{(2)}\right) \\ &= D_{-2}^{(2)} + D_{-1}^{(2)} + D_0^{(2)} - 7P_\infty^{(2)},\end{aligned}$$

which gives the initiation. now let us suppose the formula holds for a given $i \geq 3$. Recalling that $D_r^{(i)}$ ramifies in F_i/F_{i-1} if and only if $r \leq \lfloor \frac{i-3}{2} \rfloor$ (see Corollary B.3), we have

$$(h_{i-1})^{F_i} = 2 \sum_{r=-2}^{\lfloor \frac{i-3}{2} \rfloor} D_r^{(i)} + \sum_{r=\lfloor \frac{i-1}{2} \rfloor}^{i-3} 2^{2r-i+3} D_r^{(i)} - \left(2^{i+1} - 2\right) P_\infty^{(i)}.$$

This gives

$$\begin{aligned}(h_i)^{F_i} &= (h_{i-1})^{F_i} + (x_i)^{F_i} \\ &= \sum_{r=-2}^{\lfloor \frac{i-3}{2} \rfloor} D_r^{(i)} + \sum_{r=\lfloor \frac{i-1}{2} \rfloor}^{i-3} \left(2^{2r-i+3} - 2^{2r-i+2}\right) D_r^{(i)} + 2^{i-2} D_{i-2}^{(i)} - \left(2^{i+1} - 2 + 1\right) P_\infty^{(i)} \\ &= \sum_{r=-2}^{\lfloor \frac{i-2}{2} \rfloor} D_r^{(i)} + \sum_{r=\lfloor \frac{i}{2} \rfloor}^{i-2} 2^{2r-i+2} D_r^{(i)} - (2^{i+1} - 1) P_\infty^{(i)},\end{aligned}$$

which give the result for $i+1$. Note that for the last equality, we need to deal with the parity of i. Details are left to the reader. □

Both the above lemmas allow to construct an explicit function $f_{i-1,1}$ for $i \geq 3$, as well as a sufficient condition for the divisor $G_{i-1} - E_{i,1} + (f_{i-1,1})^{F_{i-1}}$ to be effective.

Proposition B.11. *Let $i \geq 3$ and set $f_{i-1,1} = h_{i-1}^2 \dfrac{1 + x_{i-1}^2}{1 + x_{i-2}} \in F_{i-1}$. Then*

$$\begin{aligned}G_{i-1} - E_{i,1} + (f_{i-1,1})^{F_{i-1}} = &\left(\left\lfloor \frac{d_i}{2} \right\rfloor - \left\lfloor \frac{d_i - 1}{2} \right\rfloor + \alpha_{i-1} + 2 - 2^{i+1}\right) P_\infty^{(i-1)} + 3 \sum_{r=-2, r \neq i-4}^{\lfloor \frac{i-3}{2} \rfloor} D_r^{(i-1)} \\ &+ 3 \sum_{r=\lfloor \frac{i-1}{2} \rfloor, r \neq i-4}^{i-3} 2^{2r-i+3} D_r^{(i-1)} + \begin{cases} D_{i-4}^{(i-1)}, \text{ if } i \in \{3,5\} \\ 2D_{i-4}^{(i-1)}, \text{ if } i = 4 \\ 13 \cdot 2^{i-6} D_{i-4}^{(i-1)}, \text{ if } i \geq 6. \end{cases}\end{aligned}$$

In particular, if $\alpha_{i-1} \geq 2^{i+1}$, the above divisor is effective.

Proof. We start by proving a formula for the divisor of $f_{i-1,1}$. From Lemmas B.9 and B.10, we have

$$\begin{aligned}(f_{i-1,1})^{F_{i-1}} &= 2(h_{i-1})^{F_{i-1}} + \left(\frac{1 + x_{i-1}^2}{1 + x_{i-2}}\right)^{F_{i-1}} \\ &= \sum_{r=-2}^{\lfloor \frac{i-3}{2} \rfloor} 2D_r^{(i-1)} + \sum_{r=\lfloor \frac{i-1}{2} \rfloor}^{i-3} 2^{2r-i+4} D_r^{(i-1)} - (2^{i+1} - 2) P_\infty^{(i-1)} + 2^{i-2} D_{i-2}^{(i-1)} \\ &\quad - \begin{cases} 2D_{-1}^{(2)}, \text{ if } i = 3 \\ 2^{i-4} D_{i-4}^{(i-1)} \text{ if } i \geq 4. \end{cases}\end{aligned}$$

At this step, we need to check in which sum $D_{i-4}^{(i-1)}$ belongs to with respect to i, in order to find the correct valuation. This leads to distinguish between the case $i \leq 5$ and $i \geq 6$. Finally, we have

$$\begin{aligned}(f_{i-1,1})^{F_{i-1}} &= \sum_{r=-2, r\neq i-4}^{\lfloor \frac{i-3}{2} \rfloor} 2D_r^{(i-1)} + \sum_{r=\lfloor \frac{i-1}{2} \rfloor, r\neq i-4}^{i-3} 2^{2r-i+4} D_r^{(i-1)} - (2^{i+1}-2)P_\infty^{(i-1)} + 2^{i-2}D_{i-2}^{(i-1)} \\ &+ \begin{cases} 2D_{i-4}^{(i-1)}, \text{ if } i \leq 5 \\ 2^{2(i-4)} D_{i-4}^{(i-1)} \text{ if } i \geq 6. \end{cases} - \begin{cases} 2D_{-1}^{(2)}, \text{ if } i = 3 \\ 2^{i-4} D_{i-4}^{(i-1)} \text{ if } i \geq 4. \end{cases} \\ &= \sum_{r=-2, r\neq i-4}^{\lfloor \frac{i-3}{2} \rfloor} 2D_r^{(i-1)} + \sum_{r=\lfloor \frac{i-1}{2} \rfloor, r\neq i-4}^{i-3} 2^{2r-i+4} D_r^{(i-1)} - (2^{i+1}-2)P_\infty^{(i-1)} + 2^{i-2}D_{i-2}^{(i-1)} \\ &+ \begin{cases} 0, \text{ if } i \in \{3,5\} \\ D_{i-4}^{(i-1)}, \text{ if } i = 4 \\ 3 \cdot 2^{i-4} D_{i-4}^{(i-1)}, \text{ if } i \geq 6 \end{cases}\end{aligned}$$

Thus, from the formula (B.4), we have

$$\begin{aligned}G_{i-1} - E_{i,1} + (f_{i-1,1})^{F_{i-1}} &= \left(\left\lfloor \frac{d_i}{2} \right\rfloor - \left\lfloor \frac{d_i - 1}{2} \right\rfloor + \alpha_{i-1} + 2 - 2^{i+1} \right) P_\infty^{(i-1)} + 3 \sum_{r=-2, r\neq i-4}^{\lfloor \frac{i-3}{2} \rfloor} D_r^{(i-1)} \\ &+ 3 \sum_{r=\lfloor \frac{i-1}{2} \rfloor, r\neq i-4}^{i-3} 2^{2r-i+3} D_r^{(i-1)} + \begin{cases} D_{i-4}^{(i-1)}, \text{ if } i \in \{3,5\} \\ 2D_{i-4}^{(i-1)}, \text{ if } i = 4 \\ 13 \cdot 2^{i-6} D_{i-4}^{(i-1)}, \text{ if } i \geq 6. \end{cases}\end{aligned}$$

It is clear from the formula that $G_{i-1} - E_{i,1} + (f_{i-1,1})^{F_{i-1}}$ is effective if and only if its valuation at $P_\infty^{(i-1)}$ is non negative, *i.e.*

$$\left\lfloor \frac{d_i}{2} \right\rfloor - \left\lfloor \frac{d_i - 1}{2} \right\rfloor + \alpha_{i-1} + 2 - 2^{i+1} \geq 0.$$

Hence, taking $\alpha_{i-1} \geq 2^{i+1}$ is a sufficient condition, for any $i \geq 3$. □

Remark B.12 (Cases $i = 1$ and $i = 2$). The cases $i \in \{1, 2\}$ have to be discussed separately, as the above formulas hold for $i \geq 3$ only. Setting $f_{0,1} = \frac{1+x_0^2}{x_0} \in F_0$ and $f_{1,1} = x_0(1 + x_1)^2 \in F_1$, we easily check that

$$G_0 - E_{1,1} + (f_{0,1})^{F_0} = \left(\left\lfloor \frac{d_1}{2} \right\rfloor - \left\lfloor \frac{d_1 - 1}{2} \right\rfloor + \alpha_0 - 2 \right) P_\infty^{(0)} + P_0^{(0)}$$

and

$$G_1 - E_{2,1} + (f_{1,1})^{F_1} = \left(\left\lfloor \frac{d_2}{2} \right\rfloor - \left\lfloor \frac{d_2 - 1}{2} \right\rfloor + \alpha_1 - 4 \right) P_\infty^{(1)} + 2D_0^{(1)}.$$

Notice that the condition on α_{i-1} given in Proposition B.11 still holds in these cases.

We now have an explicit formula for the function $f_{i-1,1}$ for any $i \geq 1$, as well as a condition to satisfy the first requirement of Definition B.7. It remains to check whenever we can find weakly balancing functions, which is the topic of the next paragraph.

Existence of weakly balancing functions. In the previous discussion, we showed that for any $i \geq 1$, with the choice of $f_{i-1,1}$ made in Proposition B.11 and taking

$$G_{i-1} := \left(\left\lfloor \frac{d_i}{2} \right\rfloor + \alpha_{i-1} \right) P_\infty^{(i-1)},$$

we have

$$G_{i-1} - E_{i,0} = \alpha_{i-1} P_\infty^{(i-1)} \geq 0$$

and

$$G_{i-1} - E_{i-1,1} + (f_{i-1}, 1)^{F_{i-1}} \geq 0,$$

provided that $\alpha_{i-1} \geq 2^{i+1}$. This ensure that G_{i-1} satisfies 1 of Definition B.7, for all $i \geq 1$. To prove the existence of weakly balancing functions, we get back to the sufficient condition (B.2), which leads to study the degree and the cardinality of the support of $G_{i-1} - E_{i,j} + (f_{i-1,j})^{F_{i-1}}$ ($j \in \{0, 1\}$).

Lemma B.13. *For any $i \geq 1$, we have*

$$\deg\left(G_{i-1} - E_{i-1,0} + (f_{i-1,0})^{F_{i-1}}\right) = \alpha_{i-1}$$

and

$$\deg\left(G_{i-1} - E_{i-1,1} + (f_{i-1,1})^{F_{i-1}}\right) = \left\lfloor \frac{d_i}{2} \right\rfloor - \left\lfloor \frac{d_i - 1}{2} \right\rfloor + \alpha_{i-1} + \begin{cases} -1 & \text{if } i = 1 \\ 0 & \text{if } i = 2 \\ 2^{\lfloor \frac{i+3}{2} \rfloor} - 2^{\lfloor \frac{3-i}{2} \rfloor} - 2^{i-1} & \text{overwise.} \end{cases}$$

Proof. Let $i \geq 3$.

The case $j = 0$ is immediate. If $j = 1$, note first that the degree of $G_{i-1} - E_{i-1,1} + (f_{i-1}, 1)^{F_{i-1}}$ does not depend on the choice of $f_{i-1,1}$. From (B.4) and Corollary B.3, we have:

$$\begin{aligned} \deg\left(G_{i-1} - E_{i-1,1}\right) &= \left\lfloor \frac{d_i}{2} \right\rfloor - \left\lfloor \frac{d_i - 1}{2} \right\rfloor + \alpha_{i-1} + \sum_{r=-2}^{\lfloor \frac{i-3}{2} \rfloor} 2^{r+2} + \sum_{r=\lfloor \frac{i-1}{2} \rfloor}^{i-3} 2^{r+1} - 2^{i-1} \\ &= \left\lfloor \frac{d_i}{2} \right\rfloor - \left\lfloor \frac{d_i - 1}{2} \right\rfloor + \alpha_{i-1} + 2^{\lfloor \frac{i+3}{2} \rfloor} - 2^{\lfloor \frac{3-i}{2} \rfloor} - 2^{i-1}. \end{aligned}$$

For $i = 1, 2$, we can compute the corresponding degrees from Remark B.12, since the formula for $E_{i,1}$ differs in this case. This gives

$$\deg\left(G_0 - E_{1,1}\right) = \left\lfloor \frac{d_1}{2} \right\rfloor - \left\lfloor \frac{d_1 - 1}{2} \right\rfloor + \alpha_0 - 1$$

and

$$\deg\left(G_1 - E_{2,1}\right) = \left\lfloor \frac{d_2}{2} \right\rfloor - \left\lfloor \frac{d_2 - 1}{2} \right\rfloor + \alpha_1.$$

□

Lemma B.14. *Let $i \geq 1$. Then*

$$\#\operatorname{Supp}\left(G_{i-1} - E_{i-1,0} + (f_{i-1,0})^{F_{i-1}}\right) = 1$$

and

$$\#\operatorname{Supp}\left(G_{i-1} - E_{i-1,1} + (f_{i-1,1})^{F_{i-1}}\right) = \begin{cases} 2 & \text{if } i = 1 \\ 3 & \text{if } i = 2 \\ 3 \cdot 2^{\lfloor \frac{i+1}{2} \rfloor} - 4 & \text{overwise.} \end{cases}$$

Proof. All cases $j = 0$ and $j = 1$ together with $i = 1, 2$ are easy. We are left to check the cases $j = 1$ and $i \geq 3$. Note that in order to compute the support, we can get ride of the valuations, provides that they are not zero. Moreover, any place in $\#\operatorname{Supp}(D_r^{(i-1)})$ as degree one, meaning that the degree of $D_r^{(i-1)}$ equals the cardinality of its support. Thus, both Proposition B.11 and *2* of Corollary B.3 gives

$$\begin{aligned} \#\operatorname{Supp}\left(G_{i-1} - E_{i-1,1} + (f_{i-1,1})^{F_{i-1}}\right) &= 1 + \sum_{r=-2}^{\lfloor \frac{i-3}{2} \rfloor} \#\operatorname{Supp}\left(D_r^{i-1}\right) + \sum_{r=\lfloor \frac{i-1}{2} \rfloor}^{i-3} \#\operatorname{Supp}\left(D_r^{i-1}\right) \\ &= 1 + \sum_{r=-2}^{\lfloor \frac{i-3}{2} \rfloor} 2^{r+2} + \sum_{r=\lfloor \frac{i-1}{2} \rfloor}^{i-3} 2^{i-1-r} \\ &= 1 + (2^{\lfloor \frac{i+3}{2} \rfloor} - 1) + 2^i \cdot 2^{-\lfloor \frac{i-1}{2} \rfloor}(1 - 2^{-\lfloor \frac{i-3}{2} \rfloor}) \\ &= 2^{\lfloor \frac{i+3}{2} \rfloor} + 2^{\lfloor \frac{i+1}{2} \rfloor} - 4 \\ &= 3 \cdot 2^{\lfloor \frac{i+1}{2} \rfloor} - 4. \end{aligned}$$

□

We can now gives a sufficient condition on α_{i-1} for all $i \geq 1$ such that with our construction, all divisors G_{i-1} are weak (G_i, x_i)–compatible. Recall that we already imposed $\alpha_{i-1} \geq 2^{i+1}$. Let us check if this restriction coincides with Equation (B.2).

Proposition B.15. *Suppose $\alpha_0 \geq 4$, $\alpha_1 \geq 8$ and for all $i \geq 3$:*

$$\alpha_{i-1} \geq \begin{cases} 2^{i+2} + 2^{i-1} - 2^{\frac{i+6}{2}} + 3 \cdot 2^{\frac{i}{2}} - 2, & \text{if } i \equiv 0[2] \\ 2^{i+2} + 2^{i-1} - 2^{\frac{i+5}{2}} + 2^{\frac{i+1}{2}} - 2, & \text{if } i \equiv 1[2] \end{cases} \tag{B.5}$$

Then for all $i \geq 1$, the divisor G_{i-1} defined by Equation (B.3) *are weak (G_i, x_i)–compatible with respect to the chosen functions $f_{i-1,j}$.*

Proof. From Proposition B.5, we have $\mathfrak{g}_1 = 1$ and $\mathfrak{g}_2 = 3$. Thus, Equation (B.2) applied for $i \in \{1, 2\}$ gives $\alpha_0 \geq 4$ and $\alpha_1 \geq 8$, which is coherent. Now, let $i \geq 3$. We have to check the inequality

$$\left\lfloor \frac{d_i}{2} \right\rfloor - \left\lfloor \frac{d_i - 1}{2} \right\rfloor + \alpha_{i-1} + 2^{\lfloor \frac{i+3}{2} \rfloor} - 2^{\lfloor \frac{3-i}{2} \rfloor} - 2^{i-1} \geq 2\mathfrak{g}_i - 1 + 3 \cdot 2^{\lfloor \frac{i+1}{2} \rfloor} - 4,$$

i.e.

$$\begin{aligned} \alpha_{i-1} &\geq 2\mathfrak{g}_i + 3 \cdot 2^{\lfloor \frac{i+1}{2} \rfloor} - 2^{\lfloor \frac{i+3}{2} \rfloor} + 2^{i-1} + \underbrace{2^{\lfloor \frac{3-i}{2} \rfloor} - 5 - \left\lfloor \frac{d_i}{2} \right\rfloor + \left\lfloor \frac{d_i - 1}{2} \right\rfloor}_{\leq -4} \\ &\geq 2\mathfrak{g}_i + 3 \cdot 2^{\lfloor \frac{i+1}{2} \rfloor} - 2^{\lfloor \frac{i+3}{2} \rfloor} + 2^{i-1} - 4 \\ &\geq \begin{cases} 2 \cdot (2^{i+1} - 3 \cdot 2^{\frac{i}{2}} + 1) + 3 \cdot 2^{\frac{i}{2}} - 2^{\frac{i+2}{2}} + 2^{i-1} - 4, & \text{if } i \equiv 0[2] \\ 2 \cdot (2^{i+1} - 2^{\frac{i+3}{2}} + 1) + 3 \cdot 2^{\frac{i+1}{2}} - 2^{\frac{i+3}{2}} + 2^{i-1} - 4, & \text{if } i \equiv 1[2] \end{cases} \\ &\geq \begin{cases} 2^{i+2} + 2^{i-1} - 2^{\frac{i+6}{2}} + 3 \cdot 2^{\frac{i}{2}} - 2, & \text{if } i \equiv 0[2] \\ 2^{i+2} + 2^{i-1} - 2^{\frac{i+5}{2}} + 2^{\frac{i+1}{2}} - 2, & \text{if } i \equiv 1[2] \end{cases} \end{aligned}$$

Since $i \geq 3$, an easy computation shows that the above lower bound on α_{i-1} is always bigger than 2^{i+1}, leading to the sufficient condition on α_{i-1} to guarantee the existence of weakly balancing functions. □

B.3 Conclusion and future work

Throughout Section B.2, we gave valid setting to build a sequence of foldable one–point AG codes along the optimal tower $\mathcal{F}$ recursively defined by Equation (B.1), namely the codes

$$\left(C_{\mathcal{L}} \left(\mathcal{X}_i, \mathcal{P}_i, d_i P_\infty^{(i)} \right) \right)_{i \geq 0},$$

where $d_{i-1} = \left\lfloor \frac{d_i}{2} \right\rfloor + \alpha_{i-1}$, for a sequence of integers $(\alpha_i)_{i \geq 0}$ satisfying conditions of Proposition B.15.

While seeking for compatible divisors, we had to weaken the condition of Definition 5.4, as it was imposed if we want to consider one–point codes at each step. Surely, it implies that while using these codes into an AG–IOPP system, the folding operator as to be modified consequently. More importantly, proofs of the key properties of the IOPP (see 5.4.2 in the case of foldable codes along the Hermitian tower) might also require some adjustments, in particular for the *soundness*. Furthermore, as in the Hermitian tower case, the degree of our divisor needs to be increased at each step, meaning that we also need to control the sequence of rates of the folding family, in particular the last RS code.

To conclude, a lot of work needs to be done to construct an efficient AG–IOPP system based on our family of foldable codes, which could be done in the future. We hope that this could lead to some improvements in the area of Interactive Proofs.

Bibliography

[ABN22] Daniel Augot, Sarah Bordage, and Jade Nardi. Efficient multivariate low–degree tests via interactive oracle proofs of proximity for polynomial codes. *Designs, Codes and Cryptography*, 2022.

[ALM+98] Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, and Mario Szegedy. Proof verification and the hardness of approximation problems. *J. ACM*, 45(3):501–555, 1998. extended version of FOCS'92.

[AS92] Sanjeev Arora and Shmuel Safra. Probabilistic checking of proofs; A new characterization of NP. In *33rd Annual Symposium on Foundations of Computer Science, Pittsburgh, Pennsylvania, USA, 24-27 October 1992*, pages 2–13. IEEE Computer Society, 1992.

[Bab85] László Babai. Trading group theory for randomness. *STOC' 85*, 1985.

[Bar18a] Elise Barelli. *On the security of short McEliece keys from algebraic and algebraic geometry codes with automorphisms*. PhD thesis, universite Paris–Saclay, 2018.

[Bar18b] Elise Barelli. On the security of some compact keys for McEliece scheme. *CoRR*, abs/1803.05289, 2018.

[BBHR18] Eli Ben-Sasson, Iddo Bentov, Yinon Horesh, and Michael Riabzev. Fast Reed–Solomon interactive oracle proofs of proximity. In *45th International Colloquium on Automata, Languages, and Programming, ICALP 2018, July 9-13, 2018, Prague, Czech Republic*, pages 14:1–14:17, 2018.

[BCC+22] Daniel J. Bernstein, Tung Chou, Carlos Cid, Jan Gilcher, and al. Classic mceliece: conservative code–based cryptography: cryptosystem specification. https://classic.mceliece.org/nist.html, October 2022.

[BCG+17] Eli Ben-Sasson, Alessandro Chiesa, Ariel Gabizon, Michael Riabzev, and Nicholas Spooner. Interactive oracle proofs with constant rate and query complexity. In *44th International Colloquium on Automata, Languages, and Programming*, volume 80 of *LIPIcs. Leibniz Int. Proc. Inform.*, pages Art. No. 40, 15. Schloss Dagstuhl. Leibniz-Zent. Inform., Wadern, 2017.

[BCGO09] Thierry P. Berger, Pierre-Louis Cayrel, Philippe Gaborit, and Ayoub Otmani. Reducing key length of the McEliece cryptosystem. In *Progress in cryptology–AFRICACRYPT 2009*, volume 5580 of *Lecture Notes in Comput. Sci.*, pages 77–97. Springer, Berlin, 2009.

[BCI+20] Eli Ben-Sasson, Dan Carmon, Yuval Ishai, Swastik Kopparty, and Shubhangi Saraf. Proximity Gaps for Reed–Solomon codes. *IACR Cryptol. ePrint Arch.*, 2020:654, 2020.

[BCP97] Wieb Bosma, John Cannon, and Catherine Playoust. The Magma algebra system. I. The user language. *J. Symbolic Comput.*, 24(3-4):235–265, 1997. Computational algebra and number theory (London, 1993).

[BCS16] Eli Ben-Sasson, Alessandro Chiesa, and Nicholas Spooner. Interactive Oracle Proofs. In *Theory of Cryptography - 14th International Conference, TCC 2016-B, Beijing, China, October 31 - November 3, 2016, Proceedings, Part II*, pages 31–60, 2016.

[BESP10] Régis Blache, Jorge Estrada Sarlabous, and Maria Petkova. A geometric interpretation of reduction in the Jacobians of C_{ab} curves. In *Arithmetics, geometry, and coding theory (AGCT 2005)*, volume 21 of *Sémin. Congr.*, pages 17–34. Soc. Math. France, Paris, 2010.

[BFGM16] Albrecht Böttcher, Lenny Fukshansky, Stephan Ramon Garcia, and Hiren Maharaj. Lattices from Hermitian function fields. *J. Algebra*, 447:560–579, 2016.

[BFLS91] László Babai, Lance Fortnow, Leonid A. Levin, and Mario Szegedy. Checking computations in polylogarithmic time. In *Proceedings of the 23rd Annual ACM Symposium on Theory of Computing, May 5-8, 1991, New Orleans, Louisiana, USA*, pages 21–31, 1991.

[BGKS20] Eli Ben-Sasson, Lior Goldberg, Swastik Kopparty, and Shubhangi Saraf. DEEP-FRI: sampling outside the box improves soundness. In *11th Innovations in Theoretical Computer Science Conference, ITCS 2020, January 12-14, 2020, Seattle, Washington, USA*, pages 5:1–5:32, 2020.

[BKS18] Eli Ben-Sasson, Swastik Kopparty, and Shubhangi Saraf. Worst–case to average case reductions for the distance to a code. In *33rd Computational Complexity Conference, CCC 2018, June 22-24, 2018, San Diego, CA, USA*, pages 24:1–24:23, 2018.

[BLM11] Paulo S. L. M. Barreto, Richard Lindner, and Rafael Misoczki. Monoidic codes in cryptography. In *Post-quantum cryptography*, volume 7071 of *Lecture Notes in Comput. Sci.*, pages 179–199. Springer, Heidelberg, 2011.

[BMvT78] Elwyn R. Berlekamp, Robert J. McEliece, and Henk C. A. van Tilborg. On the inherent intractability of certain coding problems. *IEEE Trans. Inform. Theory*, IT-24(3):384–386, 1978.

[BN20] Sarah Bordage and Jade Nardi. Interactive oracle proofs of proximity to algebraic geometry codes. *CoRR*, abs/2011.04295, 2020.

[BNLR22] Sarah Bordage, Jade Nardi, Mathieu Lhotel, and Hugues Randriambololona. Interactive oracle proofs of proximity to algebraic geometry codes. *CoRR*, abs/2011.04295, 2022.

[BRS22] Peter Beelen, Johan Rosenkilde, and Grigory Solomatov. Fast decoding of AG codes over $C_{a,b}$ curves. *IEEE Trans. Inform. Theory*, 68(11):7215–7232, 2022.

[CCMZ15] Ignacio Cascudo, Ronald Cramer, Diego Mirandola, and Gilles Zémor. Squares of random linear codes. *IEEE Trans. Inform. Theory*, 61(3):1159–1173, 2015.

[CMCP17] Alain Couvreur, Irene Márquez-Corbella, and Ruud Pellikaan. Cryptanalysis of mceliece cryptosystem based on algebraic geometry codes and their subcodes. *IEEE Transactions on Information Theory*, 63(8):5404–5418, 2017.

[Cou14] Alain Couvreur. Codes and the Cartier operator. *Proc. Amer. Math. Soc.*, 142(6):1983–1996, 2014.

[CP20] Alain Couvreur and Isabella Panaccione. Power error locating pairs. *Des. Codes Cryptogr.*, 88(8):1561–1593, 2020.

[CR20] Alain Couvreur and Hugues Randriambololona. Algebraic geometry codes and some applications. *CoRR*, abs/2009.01281, 2020.

[Del75] Philippe Delsarte. On subfield subcodes of modified Reed–Solomon codes. *IEEE Trans. Inform. Theory*, IT-21(5):575–576, 1975.

[DH76] Whitfield Diffie and Martin E. Hellman. New directions in cryptography. *IEEE Trans. Inform. Theory*, IT-22(6):644–654, 1976.

[Elk01] Noam D. Elkies. Explicit modular towers, 2001.

[Fau09] Cédric Faure. Etudes de systèmes cryptographiques construits à l'aide de codes correcteurs, en métrique de hamming et en métrique rang. *Ph.D. thesis, Ecole Polytechnique X*, 2009.

[FM08] Cédric Faure and Lorenz Minder. Cryptanalysis of the McEliece cryptosystem over hyperelliptic curves. *Proceedings of the eleventh International Workshop on Algebraic and Combinatorial Coding Theory*, pages 99–107, 2008.

[FOP+16] Jean-Charles Faugere, Ayoub Otmani, Ludovic Perret, Frederic de Portzamparc, and Jean-Pierre Tillich. Folding alternant and Goppa codes with non–trivial automorphism groups. *IEEE. Trans. Inform. Theory*, 62(1):184–198, 2016.

[FOPT10] Jean-Charles Faugere, Ayoub Otmani, Ludovic Perret, and Jean-Pierre Tillich. Algebraic cryptanalysis of McEliece variants with compact keys. *Advances in Cryptology - EUROCRYPT 2010, LNCS*, vol. 6110:279–298, 2010.

[FR93] Gui Liang Feng and T. R. N. Rao. Decoding algebraic–geometric codes up to the designed minimum distance. *IEEE Trans. Inform. Theory*, 39(1):37–45, 1993.

[Gab05] Philippe Gaborit. Shorter keys for code based cryptography. *Proceedings of the 2005 International Workshop on Coding and Cryptography (WCC 2005) (Bergen, Norway)*, pages 81–91, March 2005.

[GH00] O. Geil and T. Høholdt. Footprints or generalized Bezout's theorem. *IEEE Transactions on Information Theory*, 46(2):635–641, 2000.

[GMR89] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. The knowledge complexity of interactive proof systems. *SIAM J. Comput.*, 18(1):186–208, 1989.

[GS91] Arnaldo García and Henning Stichtenoth. Elementary abelian p–extensions of algebraic function fields. *Manuscripta Math.*, 72(1):67–79, 1991.

[GS99] Venkatesan Guruswami and Madhu Sudan. Improved decoding of Reed–Solomon and algebraic–geometry codes. *IEEE Trans. Inform. Theory*, 45(6):1757–1767, 1999.

[GS07] Arnaldo García and Henning Stichtenoth. *Topics in geometry, coding theory and cryptography*, volume 6 of *Algebra and Applications*. Springer, Dordrecht, 2007.

[GX12] Venkatesan Guruswami and Chaoping Xing. Folded codes from function field towers and improved optimal rate list decoding. *CoRR*, abs/1204.4209, 2012.

[JLJ+89] Jørn Justesen, Knud J. Larsen, H. Elbrønd Jensen, Allan Havemose, and Tom Høholdt. Construction and decoding of a class of algebraic geometry codes. *IEEE Trans. Inform. Theory*, 35(4):811–821, 1989.

[JM96] Heeralal Janwa and Oscar Moreno. McEliece public key cryptosystems using algebraic–geometric codes. *Des. Codes Cryptogr.*, 8(3):293–307, 1996.

[Kil92] Joe Kilian. A note on efficient zero–knowledge proofs and arguments (extended abstract). *Proceedings of the 24th Annual ACM Symposium on Theory of Computing, May 4-6, 1992, Victoria, British Columbia, Canada*, pages 723–732, 1992.

[KLN23] Sabira El Khalfaoui, Mathieu Lhotel, and Jade Nardi. Goppa-like ag codes from $c_{a,b}$ curves and their behaviour under squaring their dual, 2023.

[LC16] Phong Le and Sunil Chetty. On the dimension of algebraic–geometric trace codes. *Mathematics*, 4(2):32, 2016.

[LFKN90] Carsten Lund, Lance Fortnow, Howard J. Karloff, and Noam Nisan. Algebraic methods for interactive proof systems. In *31st Annual Symposium on Foundations of Computer Science, St. Louis, Missouri, USA, October 22-24, 1990, Volume I*, pages 2–10. IEEE Computer Society, 1990.

[Liu02] Qing Liu. *Algebraic geometry and arithmetic curves*. Oxford, 2002.

[Mah04] Hiren Maharaj. Code construction on fiber products of Kummer covers. *IEEE Trans. Inform. Theory*, 50(9):2169–2173, 2004.

[MB09] Rafael Misoczki and Paulo Barreto. Compact McEliece keys from Goppa codes. *Selected Areas in Cryptography (Calgary, Canada)*, 2009.

[McE78] Robert J. McEliece. A public–key cryptosystem based on algebraic coding theory. *DSN Progress Report 44*, pages 114–116, 1978.

[Mic98] Silvio Micali. Computationally–sound proofs. In *Logic Colloquium '95 (Haifa)*, volume 11 of *Lecture Notes Logic*, pages 214–268. Springer, Berlin, 1998.

[Min07] Lorenz Minder. Cryptography based on error correcting codes. *Ph.D. thesis, Ecole Polytechnique Fédérale de Lausanne*, 2007.

[Miu93] S. Miura. Algebraic geometric codes on certain place curves. *Electronics and Communication in Japan (Part III: Fundamental Electronic Science)*, 76(12):1-13, 1993.

[Mor93] Carlos Moreno. *Algebraic Curves over Finite Fields*. Cambridge Tracts in Mathematics. Cambridge University Press, 1993.

[MS86] Florence J. MacWilliams and Neil J. A. Sloane. *The theory of error–correcting codes.* North-Holland. Amsterdam, fifth edition, 1986.

[MT21] Rocco Mora and Jean-Pierre Tillich. On the dimension and structure of the square of the dual of a Goppa code. *CoRR*, abs/2111.13038, 2021.

[Mum70] David Mumford. *Varieties defined by quadratic equations.* Questions on algebraic varieties, C.I.M.E., III Ciclo. Edizioni Cremonese, Rome, 1970.

[NEK21] Gábor P. Nagy and Sabira El Khalfaoui. Towards the security of McEliece's cryptosystem based on Hermitian subfield subcodes. *Prikl. Diskr. Mat. Suppl.*, pages 168–175, 2021.

[Nie86] H. Niederreiter. Knapsack–type cryptosystems and algebraic coding theory. *Problems Control Inform. Theory/Problemy Upravlen. Teor. Inform.*, 15(2):159–166, 1986.

[NOQ11] Francesco Noseda, Gilvan Oliveira, and Luciane Quoos. Bases for riemann–roch spaces of one–point divisors on an optimal tower of function fields. *arXiv*, 2011.

[Pel89] Ruud Pellikaan. On a decoding algorithm for codes on maximal curves. *IEEE Trans. Inform. Theory*, 35(6):1228–1232, 1989.

[Pet10] Christiane Peters. *Information–set decoding for linear codes over* $\mathbf{F}_q$, volume 6061 of *Lecture Notes in Comput. Sci.* Springer, Berlin, 2010.

[Pra62] Eugene Prange. The use of information sets in decoding cyclic codes. *IRE Trans.*, IT-8:S 5–S 9, 1962.

[RSA78] R. L. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public–key cryptosystems. *Comm. ACM*, 21(2):120–126, 1978.

[Sen94] Nicolas Sendrier. On the structure of a randomly permuted concatened code. *EUROCODE'94.*, pages 169–173, 1994.

[Sha92] Adi Shamir. IP = PSPACE. *J. ACM*, 39(4):869–877, 1992.

[She93] Ba-Zhong Shen. A justesen construction of binary concatanated codes that asymptotically meet the zyablov bound for low rate. *IEEE Transactions on Information Theory*, 39:239–242, 1993.

[Sho94] Peter W. Shor. Algorithms for quantum computation: discrete logarithms and factoring. In *35th Annual Symposium on Foundations of Computer Science (Santa Fe, NM, 1994)*, pages 124–134. IEEE Comput. Soc. Press, Los Alamitos, CA, 1994.

[SJM+95] Shajiro Sakata, Jørn Justesen, Y. Madelung, Helhe Elbrønd Jensen, and Tom Høholdt. Fast decoding of algebraic–geometric codes up to the designed minimum distance. *IEEE Trans. Inform. Theory*, 41(6, part 1):1672–1677, 1995. Special issue on algebraic geometry codes.

[SS92] Vladimir Michilovich Sidelnikov and S.O. Shestakov. On the insecurity of cryptosystem based on generalized Reed–Solomon codes. *Descrete Math. Appl.* ***1***, no. 4:439–444, 1992.

[Sti09] Henning Stichtenoth. *Algebraic function fields and codes*, volume 254 of *Graduate Texts in Mathematics.* Springer-Verlag, Berlin, second edition, 2009.

[Sud97] Madhu Sudan. Decoding of Reed Solomon codes beyond the error–correction bound. *J. Complexity*, 13(1):180–193, 1997.

[SV90] Alexei N. Skorobogatov and Sergei G. Vlăduţ. On the decoding of algebraic–geometric codes. *IEEE Trans. Inform. Theory*, 36(5):1051–1060, 1990.

[Tia19] Yongge Tian. Formulas for calculating the dimensions of the sums and the intersections of a family of linear subspaces with applications. *Beiträge zur Algebra und Geometrie / Contributions to Algebra and Geometry*, 60, 2019.

[TVN07] Michael A. Tsfasman, Serge G. Vlăduţ, and Dmitry Nogin. *Algebraic geometric codes.* Number n0. 139 in Cambridge Tracts in Mathematics. Cambridge University Press, 2007.

[WGR22] Violetta Weger, Niklas Gassner, and Joachim Rosenthal. A survey on code-based cryptography, 2022.

[Wir88] Michael Wirtz. On the parameters of Goppa codes. *IEEE transactions on information theory*, 34(5):1341–1343, 1988.

List of Figures

List of Algorithms

List of Tables

ECOLE DOCTORALE
CARNOT - PASTEUR

Titre : Utilisation de codes de géométrie algébrique structurés en cryptographie moderne.
Mots clés : Codes linéaires, cryptosystème de McEliece, cryptanalyse, géométrie algébrique, preuves par oracle intéractif.

Résumé : En 1978, McEliece introduit un nouveau schéma de chiffrement à clé publique fondé sur les codes correcteurs d'erreurs. Depuis, il s'est avéré avoir de nombreux avantages, comme un chiffrement et déchiffrement rapide, couplé au fait qu'il soit un bon candidat en cryptographie post–quantique. La principale contrainte est qu'il impose des tailles de clé trop grosses comparées aux autres systèmes de chiffrement à clé publique actuels. Dans ce contexte, nous étudions la sécurité de certaines variantes du schéma de McEliece, à base de sous–codes de codes de géométrie algébrique (SSAG). Plus précisément, nous montrons que la structure secrète du code SSAG public peut être récupérée à partir du celle de son sous–code invariant, qui a de plus petits paramètres.

Initialement fondée sur les codes de Goppa classique, la proposition initiale de McEliece est encore considérée comme sécurisé aujourd'hui. Tenant compte de ce fait, nous définissons une nouvelle famille de codes : les codes AG Goppa–like. L'idée est de copier la structure algébrique des codes de Goppa héritée du choix du multiplicateur, tout en considérant des courbes de genre supérieur, permettant de construire de plus longs codes. Se concentrant sur les codes à un point sur des courbes $C_{a,b}$, nous étudions le comportement de la dimension du carré de leur dual pour évaluer leur résistance aux attaques par distingueur. Comme cette famille peut être encodée rapidement, il s'agit d'un bon candidat pour remplacer les codes de Goppa classiques.

Dans le contexte des preuves par oracle interactif (IOPs), nous engageons l'étude de tests de proximité à des codes AG. Le problème de tester la proximité à un code $\mathcal{C}$ consiste à distinguer le cas ou un mot donné en entrée appartient à $\mathcal{C}$ du cas où il en est très éloigné. Dans le but de généraliser le protocole FRI s'appuyant sur les codes de Reed-Solomon, nous proposons un cadre valide pour définir un IOP de Proximité aux codes AG (AG–IOPP). Comme exemple concret, nous nous concentrons sur les codes construits à partir d'une tour de courbes Hermitiennes, qui peuvent être définis sur un alphabet de taille polylogarithmique. Nous donnons également une famille de codes repliables dont l'AG-IOPP correspondant atteint une complexité quasilinéaire pour le prouveur et polylogarithmique pour le vérifieur.

Title : Using structured algebraic geometry codes in modern cryptography.
Keywords : Linear codes, McEliece cryptosystem, cryptanalysis, algebraic geometry, interactive oracle proofs.

Abstract : In 1978, McEliece introduced a new public–key cryptosystem, based on error-correcting codes. Since then, it has demonstrated to have a lot of advantages, such as a fast encryption and decryption, in addition to the fact that it is a good candidate for post-quantum cryptography. The main constraint is that it imposes large keys sizes compared with other actual public–key cryptosystems. In this context, we study the security of variants of McEliece's encryption schemes based on structured subfield subcode of algebraic geometry codes (SSAG). More precisely, we show that the underlying secret structure of the public SSAG can be recovered from that of its invariant code, which has smaller parameters.

Initially based on the family of classical Goppa codes, the first proposal of McEliece is still considered secure today. Taking this into account, we define a new family of codes: Goppa–like AG codes. The idea is to mimic the algebraic structure of Goppa codes inherited from the choice of the multiplier while considering higher genus curves, which allows to construct longer codes. Focusing one one–point codes from $C_{a,b}$ curves, we study the behavior of the square of their dual to determine their resistance to distinguisher attacks. As this family can be efficiently encoded, it is a good candidate to replace classical Goppa codes.

In the context of Interactive Oracle Proofs (IOPs), we initiate the study of proximity tests to AG codes. The problem of testing proximity to a code $\mathcal{C}$ consists in distinguishing between the case where an input word belong to $\mathcal{C}$ and the case where it is far from it. Aiming to generalize the FRI protocol based on Reed-Solomon codes, we give valid setting to design an efficient IOP of Proximity to AG codes (AG–IOPP). As concrete instantiation, we focus on AG codes arising from a tower of Hermitian curves, which can be defined over polylogarithmic–size alphabet. We also give a family of foldable AG codes on this tower whose corresponding AG–IOPP achieves quasilinear prover time and polylogarithmic verification.

Classification AMS : 11T71, 14G50, 14H05, 11G20, 14Q20.

UBFC UNIVERSITÉ BOURGOGNE FRANCHE-COMTÉ

Université de Bourgogne Franche-Comté
32, avenue de l'Observatoire
25000 Besançon, France