CURVES AND CODES

by

John R. Kerl

A Thesis Presented in Partial Fulfillment
of the Requirements for the Degree
Master of Arts

ARIZONA STATE UNIVERSITY

May 2005

CURVES AND CODES

by

John R. Kerl

has been approved

May 2005

APPROVED:

____, Chair

____

____

Supervisory Committee

ACCEPTED:

____

Department Chair

____

Dean, Division of Graduate Studies

ABSTRACT

This paper gives an example-driven overview of algebraic-geometry codes, with attention confined to bivariate Goppa codes. J. Walker's *Codes and Curves* uses notation that is standard for graduate mathematics, but unfortunately does not discuss decoding; O. Pretzel's *Codes and Algebraic Curves* gives a full discussion, but using non-standard notation. The current work is a synthesis of the two, extending Walker's work by including a discussion of the Skorobogatov-Vlăduţ (SV) decoding algorithm.

First, notation for finite fields is given; then, the engineering problem is defined. Selected concepts from algebraic geometry are introduced, illustrated by examples. The construction and encoding of Goppa codes is described, followed by an exposition of the SV decoding algorithm. Several worked examples are shown. Software implementations of the encoder and decoder are discussed, followed by performance analysis and comparison with other codes. Finally, directions for further research are sketched.

This thesis is dedicated to Dr. Philip Leonard, Professor Emeritus of Mathematics at Arizona State University, who showed me — much to my astonishment — the connections between linear feedback shift registers and finite fields, revealing that engineering and abstract algebra need not be mutually exclusive.

ACKNOWLEDGMENTS

This work is inspired by, and is a continuation of, J. Walker's *Codes and Curves* [Wal].

TABLE OF CONTENTS

Page

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

Preliminaries

This is an expository paper. Proofs due to other authors are cited as such; proofs without citation are due to this author. Graduate abstract algebra at the level of [DF] is taken for granted. The following symbols are used:

- $\mathbb{Z}$ denotes the integers, $\mathbb{Q}$ the rationals, $\mathbb{R}$ the real numbers, and $\mathbb{C}$ the complex numbers.
- Given a field K, $K[x]$ denotes the ring of univariate polynomials with coefficients in K. Likewise, $K[x_1, \ldots, x_n]$ denotes the ring of polynomials with n variables and coefficients in K. The field of quotients of the integral domain $K[x_1, \ldots, x_n]$ is written $K(x_1, \ldots, x_n)$.
- Given an extension field L of K and $u \in L$, $K(u)$ denotes the smallest extension field of K containing u. If u is algebraic over K (which is the case for finite fields), then $K(u) = K[u]$. The algebraic closure of a field K is written $\overline{K}$.
- Given functions $f(x)$ and $g(x)$, the product of functions is written $(fg)(x)$ or $fg(x)$, both equal to $f(x)g(x)$ by definition. The notation $f(g(x))$ is used for composition of functions.
- The number of elements in a finite set S is written $\#S$.
- A finite field with q elements is denoted $\mathbb{F}_q$. This is unambiguous since a finite field of a given order is unique up to isomorphism. Necessarily, $q = p^r$ for some prime integer p, called the *characteristic* of $\mathbb{F}_q$, and some positive integer r, called the *degree* of $\mathbb{F}_q$ over $\mathbb{F}_p$. For this paper, p is always 2.

- Vectors in a finite-dimensional vector space, say of dimension m, are usually written in the form $\boldsymbol{u}$, having components $(u_1, \ldots, u_m)$ with respect to the standard basis. In particular, if a vector $\boldsymbol{v}$ is written in boldface type, then $(v_1, \ldots, v_n)$ in non-boldface type denotes the components of $\boldsymbol{v}$.

- Likewise, given a matrix A in uppercase, a_{ij} in lowercase refers to the element in the ith row and jth column of A. The entries of A may be specified by an expression in the form $(f(i,j))_{ij}$, e.g. $A = (\frac{1}{i+j+1})_{ij}$.

- Given a field K and vectors $\boldsymbol{u} = (u_1, \ldots, u_n)$ and $\boldsymbol{v} = (v_1, \ldots, v_n)$ in the n-dimensional vector space K^n, $\boldsymbol{u} \cdot \boldsymbol{v}$ denotes the standard dot product $\sum_{i=1}^{n} u_i v_i$.

- The ideal generated by an element a of a ring R is denoted $\langle a \rangle$.

- The notation $\lfloor x \rfloor$ denotes the largest integer less than or equal to x.

CHAPTER 2

Finite Fields

A compact notation for finite fields is presented in this chapter, along with computational notes. See [LN] for full information on finite fields.

2.1. Compact Notation

The finite field $\mathbb{F}_q$ necessarily forms a vector space over $\mathbb{F}_p$. In order to define vector-vector multiplication on $\mathbb{F}_q$, i.e. in order to make $\mathbb{F}_q$ into an $\mathbb{F}_p$-algebra, $\mathbb{F}_q$ may be viewed as $\mathbb{F}_p[x]/\langle m(x)\rangle$ for some degree-r monic irreducible $m(x)$ over $\mathbb{F}_p$. If $u = x + \langle m(x)\rangle$, then $\mathbb{F}_q = \mathbb{F}_p[u]$. This makes u a root of $m(x)$ in the extension field $\mathbb{F}_q$.

Since $m(x)$ has degree r, elements of $\mathbb{F}_p[u]$ are of the form $a_{r-1}u^{r-1} + \ldots + a_1u + a_0$ for $a_i \in \mathbb{F}_p$. This may be written as an r-tuple of coefficients, $(a_{r-1}, \ldots, a_1, a_0)$, e.g. $(1, 0, 1, 1)$, where $m(x)$ is taken from context. Since $p = 2$ for this paper, one may further abbreviate by omitting parentheses and commas to write $a_{r-1} \ldots a_1a_0$, e.g. 1011. Further abbreviation may be obtained using a compact hexadecimal (base-16) notation, clustering quadruples of digits beginning at the right. This is illustrated using table 1.

Binary	0000	0001	0010	0011	0100	0101	0110	0111
Hex	0	1	2	3	4	5	6	7
Binary	1000	1001	1010	1011	1100	1101	1110	1111
Hex	8	9	a	b	c	d	e	f

Table 1. Binary and hexadecimal notation

The hexadecimal digits a-f are written upright, to distinguish them from variables a-f. For example, the following are equivalent notations for an element of $\mathbb{F}_q$ where $r \geq 10$: $u^9 + u^7 + u^6 + u^4 + u^3 + 1$, $(1,0,1,1,0,1,1,0,0,1)$, 1011011001, 2d9.

This compact notation is important for the following reasons: (1) Since this paper deals with vectors over $\mathbb{F}_q$, the compact notation avoids nested parentheses. (2) The compact notation makes finite-field elements look simply like numbers, or scalars, which is precisely what they are in this context. (3) The 16-digit hexadecimal alphabet makes individual finite-field elements short and easier to recognize than long strings of ones and zeroes. For example, $(u^7+u^5+u^3+u^2+u, u^7+u^6+u^5+u^3+u+1, u^4+u^3+u^2)$, looks much like $(u^7+u^5+u^3+u^2+u, u^7+u^6+u^5+u^2+u+1, u^4+u^3+u^2)$, as do $(10101110, 11101011, 00011100)$ and $(10101110, 11100111, 00011100)$, but $(\mathrm{ae}, \mathrm{eb}, 1\mathrm{c})$ and $(\mathrm{ae}, \mathrm{e}7, 1\mathrm{c})$ are visibly distinct. (All three represent the same pair of elements of the vector space $\mathbb{F}_{256}^3$.) (4) Since 16 is a power of 2, the individual bits are readily recovered for computations as needed.

It may be verified by trial factorization that the polynomials listed in table 2 are monic irreducibles over $\mathbb{F}_2$. Unless otherwise noted, it is assumed in this paper that the fields $\mathbb{F}_{2^r}$, for $r = 1, \ldots, 6$, are defined using these moduli. These particular moduli are chosen for the following two reasons. First, they are the lexically smallest monic irreducibles of each degree in $\mathbb{F}_2[x]$. Second, they are *primitive* in the sense that their roots generate the multiplicative group of $\mathbb{F}_p[x]/\langle m(x)\rangle$. It is possible for the lexically smallest monic irreducible of a given degree to be imprimitive, but the first such occurrence is at $r = 8$.

r	1	2	3	4	5	6
$m(x)$	$x+1$	x^2+x+1	x^3+x+1	x^4+x+1	x^5+x^2+1	x^6+x+1

Table 2. Selected monic irreducible polynomials over $\mathbb{F}_2$

2.2. Tables

Arithmetic in $\mathbb{F}_q$ is readily performed using the construction $\mathbb{F}_q \cong \mathbb{F}_p[x]/\langle m(x)\rangle$: addition and subtraction are performed coefficientwise mod p; multiplication is done in $\mathbb{F}_p[x]$, then reduced mod $m(x)$. Division may be done using the extended Euclidean algorithm in $\mathbb{F}_p[x]$: let $a(x) + \langle m(x)\rangle$ be non-zero in $\mathbb{F}_p[x]/\langle m(x)\rangle$. Since $m(x)$ is assumed to be irreducible, $a(x)$ is relatively prime to $m(x)$, with $\gcd(a(x), m(x)) = 1$. Therefore, there are $g(x), h(x) \in \mathbb{F}_p[x]$ such that $g(x)a(x) + h(x)m(x) = 1$. This means that $g(x)a(x) \equiv 1 \pmod{m(x)}$, and thus $g(x) + \langle m(x)\rangle$ is the reciprocal of $a(x) + \langle m(x)\rangle$. Tables 3 through 5 are provided as a convenience, to accelerate computation for a few small fields.

$+$	0	1	$\cdot$	0	1
0	0	1	0	0	0
1	1	0	1	0	1

Table 3. Addition and multiplication for $\mathbb{F}_2$, using $m(x) = x + 1$.

$+$	0	1	2	3	$\cdot$	0	1	2	3
0	0	1	2	3	0	0	0	0	0
1	1	0	3	2	1	0	1	2	3
2	2	3	0	1	2	0	2	3	1
3	3	2	1	0	3	0	3	1	2

Table 4. Addition and multiplication for $\mathbb{F}_4$, using $m(x) = x^2 + x + 1$.

$+$	0	1	2	3	4	5	6	7	$\cdot$	0	1	2	3	4	5	6	7
0	0	1	2	3	4	5	6	7	0	0	0	0	0	0	0	0	0
1	1	0	3	2	5	4	7	6	1	0	1	2	3	4	5	6	7
2	2	3	0	1	6	7	4	5	2	0	2	4	6	3	1	7	5
3	3	2	1	0	7	6	5	4	3	0	3	6	5	7	4	1	2
4	4	5	6	7	0	1	2	3	4	0	4	3	7	6	2	5	1
5	5	4	7	6	1	0	3	2	5	0	5	1	4	2	7	3	6
6	6	7	4	5	2	3	0	1	6	0	6	7	1	5	3	2	4
7	7	6	5	4	3	2	1	0	7	0	7	5	2	1	6	4	3

Table 5. Addition and multiplication for $\mathbb{F}_8$, using $m(x) = x^3 + x + 1$.

The Galois group of $\mathbb{F}_q$ over $\mathbb{F}_p$ is cyclic of order r, generated by the p-power Frobenius map $\alpha \mapsto \alpha^p$. Elements of $\mathbb{F}_q$ cluster into conjugates via the action of the Frobenius map: two field elements are in the same cluster, or orbit, if and only if they share the same minimal polynomial over

$\mathbb{F}_p$. Table 6 shows *root charts* for $\mathbb{F}_{2^r}$, with $r = 1, 2, 3, 4$. Each element of $\mathbb{F}_{2^r}$ is listed along with its conjugates, preceded by their common minimal polynomial. Furthermore, elements are listed in Frobenius order. For example, in $\mathbb{F}_{16}$, $8^2 =$ c, $\text{c}^2 =$ f, $\text{f}^2 =$ a, and $\text{a}^2 = 8$.

Min. poly.	Orbit
x	0
$x+1$	1

Min. poly.	Orbit	
x	0	
$x+1$	1	
x^2+x+1	2	3

Min. poly.	Orbit		
x	0		
$x+1$	1		
x^3+x+1	2	4	6
x^3+x^2+1	3	5	7

Min. poly.	Orbit			
x	0			
$x+1$	1			
x^2+x+1	6	7		
x^4+x+1	2	4	3	5
x^4+x^3+1	b	9	d	e
$x^4+x^3+x^2+x+1$	8	c	f	a

Table 6. Root charts for $\mathbb{F}_2$ through $\mathbb{F}_{16}$.

Remark 2.2.1. A finite field of order p^r has a unique subfield of order p^d for each $d \mid r$, and these are its only subfields. Looking at the root chart for a field, it is easy to see which elements comprise these various subfields: the elements of a subfield of order p^d are exactly those whose minimal polynomials have degree dividing d. For example, $\mathbb{F}_4$ and $\mathbb{F}_8$ have no proper non-trivial subfields; $\mathbb{F}_{16}$ has subfield $\mathbb{F}_4$, with elements visibly equal to 0, 1, 6, and 7.

CHAPTER 3

Coding Theory

The essential definitions for coding theory are given here. Engineering motivations, channel models, error probabilities, examples of non-AG codes, etc. are not discussed. See any of [MS], [Ber], [PW], [VvO] for a thorough introduction. Coding theory is a broad subject; here, discussion is limited to linear block codes over field alphabets.

3.1. Linear Codes

A *block code* (or simply a *code*) is any subset C of the vector space $\mathbb{F}_q^n$. If C is not just a subset of $\mathbb{F}_q^n$ but a subspace as well, then C is said to be a *linear code*. The vector-space dimension $k = \dim_{\mathbb{F}_q}(C)$ is called the *dimension* of the linear code C; n is called the *length* of C.

The *encoding problem* is that of *embedding* the smaller vector space $\mathbb{F}_q^k$ into the larger vector space $\mathbb{F}_q^n$, in a maximal way as is to be discussed below. A vector $\boldsymbol{m}$ in $\mathbb{F}_q^k$ is called a *message word*; its image $\boldsymbol{u}$ in C is called a *code word*. During transmission, a code word may be turned into any element (say $\boldsymbol{v}$) of $\mathbb{F}_q^n$. This is called a *received word*.

Notation 3.1.1. For brevity, n-tuples are written in this chapter in the form 111 rather than $(1, 1, 1)$. There is no ambiguity as long as each coordinate takes only a single digit, which is certainly the case over $\mathbb{F}_2$.

Example 3.1.2. The three-bit *repetition code* embeds $\mathbb{F}_2$ into $\mathbb{F}_2^3$ by the map which sends 0 to 000 and 1 to 111. Here, $k = 1$ and $n = 3$. Note that there are $2^3 = 8$ elements of $\mathbb{F}_2^3$, but only two of them are code words. More generally, one obtains a *family* of n-bit repetition codes, embedding $\mathbb{F}_2$ into $\mathbb{F}_2^n$: 0 maps to the vector consisting of n zeroes, and 1 maps to n ones. Clearly, these are linear codes.

Example 3.1.3. The family of n-bit *parity codes* embed $\mathbb{F}_2^{n-1}$ into $\mathbb{F}_2^n$ via the following: the additional bit at the end is the sum of the previous $n-1$, taken mod 2. For example, 1101 encodes to 11011. Here, $k = n-1$. These are also clearly linear codes.

3.2. Minimum Distance

Definition 3.2.1. The *Hamming weight* of a vector $\boldsymbol{v}$ in $\mathbb{F}_q^n$ is given by the number of non-zero entries in $\boldsymbol{v}$. This is a function $\mathrm{wt} : \mathbb{F}_q^n \to \mathbb{Z}$.

Definition 3.2.2. The *Hamming distance* between vectors $\boldsymbol{u}$ and $\boldsymbol{v}$ in $\mathbb{F}_q^n$ is given by the number of non-zero entries in their difference. That is, $\mathrm{dist} : \mathbb{F}_q^n \times \mathbb{F}_q^n \to \mathbb{Z}$ is given by $\mathrm{dist}(\boldsymbol{u}, \boldsymbol{v}) = \mathrm{wt}(\boldsymbol{u} - \boldsymbol{v})$.

For example, $\mathrm{wt}(101) = 2$ and $\mathrm{dist}(101, 110) = \mathrm{wt}(010) = 1$.

Definition 3.2.3. The *minimum distance* of a code C, written $d(C)$ or simply d, is the smallest distance between distinct pairs of vectors of C.

If C is linear, then the difference of $\boldsymbol{u}$ and $\boldsymbol{v}$ is also in C, so the minimum distance is then the *minimum weight* over all non-zero vectors in C. For example, the three-bit repetition code has minimum distance 3.

Definition 3.2.4. The *weight distribution* of a code C is the function mapping from the integers w such that $0 \le w \le n$ to the number of code words in C having weight w.

For example, consider the three-bit repetition code, with code words 000 and 111, and the three-bit parity-check code, with code words 000, 110, 101, and 011. Their weight distributions are shown in table 7.

Definition 3.2.5. The values n, k, d, and q are called the *code parameters* for a given code.

3-bit rep. code					3-bit par. code				
w	0	1	2	3	w	0	1	2	3
#	1	0	0	1	#	1	0	3	0

Table 7. Weight distributions for three-bit repetition and parity-check codes

Notation 3.2.6. If a linear code C over $\mathbb{F}_q$ has length n, dimensions k, and minimum distance d, then C is said to be an $[n, k, d]_q$ code.

For example, the n-bit binary repetition code is an $[n, 1, n]_2$ code; the n-bit binary parity-check code is $[n, n - 1, 2]_2$.

A linear code C contains q^k code words among the q^n possible n-tuples over $\mathbb{F}_q$. After transmission, errors may result in a code word being received as an arbitrary element of $\mathbb{F}_q^n$. In order that the receiver may correct the highest number of errors, the code words should be placed at maximum distance from one another in n-dimensional space. The minimum distance is related to error control as shown in the next section.

3.3. Error Detection and Error Correction

The relationship between a code's minimum distance and its ability to detect and/or correct errors is illustrated by example. Suppose single 0's and 1's are transmitted using a three-bit repetition code. The receiver may trust the sender to encode only 0 or 1, as 000 or 111, respectively, but due to noise any of 000, 001, 010, 011, 100, 101, 110 or 111 might be received. If the block 111 were received, then the receiver may assume that either 111 was sent and all bits are intact, or 000 was sent and there was a triple bit error. The *maximum-likelihood assumption* is made in this paper (see [MS] for the statistical basis for this assumption) that the former conclusion is the more likely, namely, that fewer errors are more likely than more errors. Now suppose 101 was received: either 000 was sent and two bits were flipped, or 111 was sent and the middle bit was flipped. The latter case is the more likely. The receiver cannot distinguish the two cases, and so would make a decoding error in the former case.

In figure 1, code words are marked with an open circle. Maximum-likelihood decoding involves finding the code word which is nearest to a given received word. For this three-bit repetition code, any one-bit error can be correctly detected. The receiver cannot detect a triple-bit error at all; a double-bit error looks like a single-bit error instead. These latter two cases are referred to as *decoding errors*.

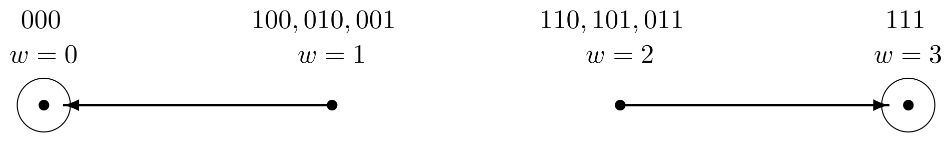

Figure 1. Maximum-likelihood decoding for the three-bit repetition code

Now suppose a four-bit repetition code is used (figure 2). Then 0 is encoded as 0000 and 1 is encoded as 1111. If a vector of weight 0 or 1 is received, it is decoded to 0; if a vector of weight 3 or 4, it is decoded to 1. However, if a vector with two zero bits and two one bits is received, that vector is clearly not a code word (the only code words are 0000 and 1111), but the receiver cannot tell whether two bits got set by error, or two bits got cleared by error. For this four-bit repetition code, 1-bit errors can be corrected, but 2-bit errors can only be detected. More generally, one sees intuitively that if the minimum distance d of a code C is odd, then C can detect and correct up to $\frac{d-1}{2}$ errors per block. If d is even, then C can correct up to $\frac{d}{2} - 1$ errors per block, and can detect up to $\frac{d}{2}$ errors per block. Whether d is even or odd, one may then say that a linear code C can correct at most $\lfloor \frac{d-1}{2} \rfloor$ errors per block.

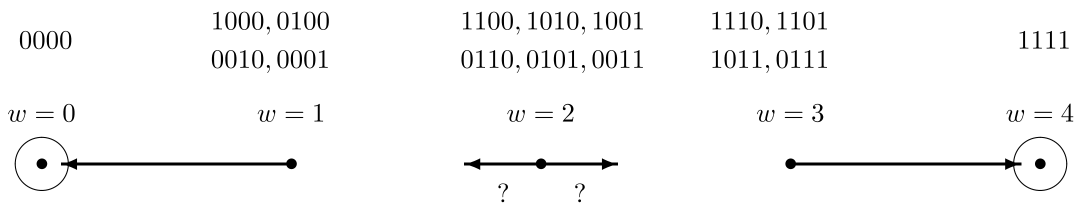

Figure 2. Maximum-likelihood decoding for the four-bit repetition code

Remark 3.3.1. Note that error-correction ability is counted in terms of elements of $\mathbb{F}_q$. For engineering applications, q is typically a power of 2. All of the example codes in this chapter use $q = 2$, but the codes discussed in later chapters are over larger fields. Transmission errors typically affect bits, which in this case are distinct from field elements. For example, the error pattern $(0, 0, 1, 1, 0, 1, 0, 0)$ is a 3-bit error in $\mathbb{F}_2^8$. If the same data were treated as a 4-tuple over $\mathbb{F}_4$, i.e. $((0, 0), (1, 1), (0, 1), (0, 0))$, then the error pattern has weight 2, not 3.

3.4. Maximization and Minimization

The example of the previous section motivates the following description of the fundamental problems of constructing codes:

- Given k and n, d should be maximized in order to achieve high error correction.
- Given d and k, n should be minimized in order to achieve high data rate through the communications channel.
- Given n and d, k should be maximized in order to put as much end-user data as possible in each encoded block.

Here, q is assumed to be given, although this need not be the case [Sud]. Once a code has been constructed, efficient encoding and decoding are separate problems.

Figure 3 shows selected embeddings of $\mathbb{F}_2$ into $\mathbb{F}_2^3$. The one-dimensional vector space $\mathbb{F}_2$ is represented by the ends of a line segment. The three-dimensional vector space $\mathbb{F}_2^3$ is represented by the vertices of a cube, where the horizontal axis represents the first coordinate, the vertical axis represents the second coordinate, and the receding axis represents the third coordinate. Heavy dots represent the image of $\mathbb{F}_2$. In the first embedding, $\mathbb{F}_2$ is sent to the front left edge, with 0 mapping to 000 and 1 mapping to 010. This code visibly has minimum distance 1. In the second embedding of figure 3, $\mathbb{F}_2$ is sent to opposite corners of the left face, with 0 mapping to 000 and 1 mapping to 011. This code has minimum distance 2. The third embedding shows 0 mapping to 000 and 1 mapping to 111, where the elements of the image of $\mathbb{F}_2$ are as far apart as possible, namely, with minimum distance 3. This is the three-bit repetition code.

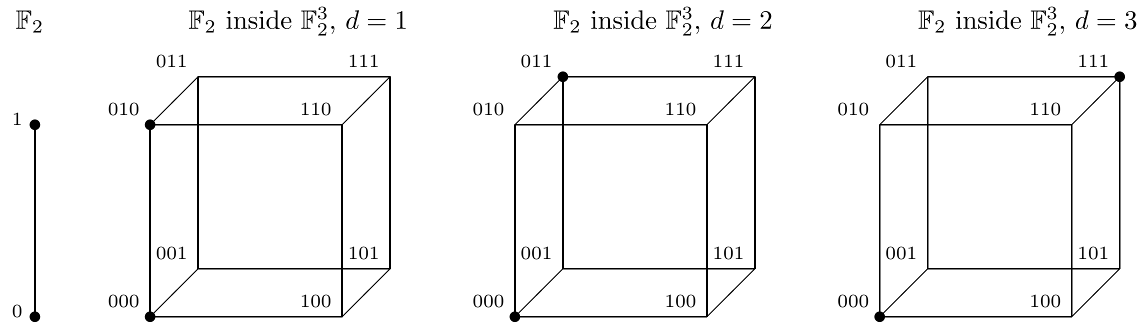
$\mathbb{F}_2$
$\mathbb{F}_2$ inside $\mathbb{F}_2^3$, $d = 1$
$\mathbb{F}_2$ inside $\mathbb{F}_2^3$, $d = 2$
$\mathbb{F}_2$ inside $\mathbb{F}_2^3$, $d = 3$
1
0
011
111
010
110
001
101
000
100

Figure 3. Selected embeddings of $\mathbb{F}_2$ into $\mathbb{F}_2^3$

Figure 4 shows selected embeddings of $\mathbb{F}_2^2$ into $\mathbb{F}_2^3$. Much as in figure 3, the two-dimensional vector space $\mathbb{F}_2^2$ is represented by the vertices of a square and the three-dimensional vector space $\mathbb{F}_2^3$ is represented by the vertices of a cube. Heavy dots represent the image of $\mathbb{F}_2^2$ in $\mathbb{F}_2^3$. In the first embedding, $\mathbb{F}_2^2$ is sent to a slant plane, with $00 \mapsto 000$, $01 \mapsto 011$, $10 \mapsto 100$, and $11 \mapsto 111$. This code has minimum distance 1. In the second embedding of figure 4, the image of $\mathbb{F}_2^2$ looks like a tetrahedron, but algebraically it is a plane. This code, the three-bit parity-check code, has minimum distance 2.

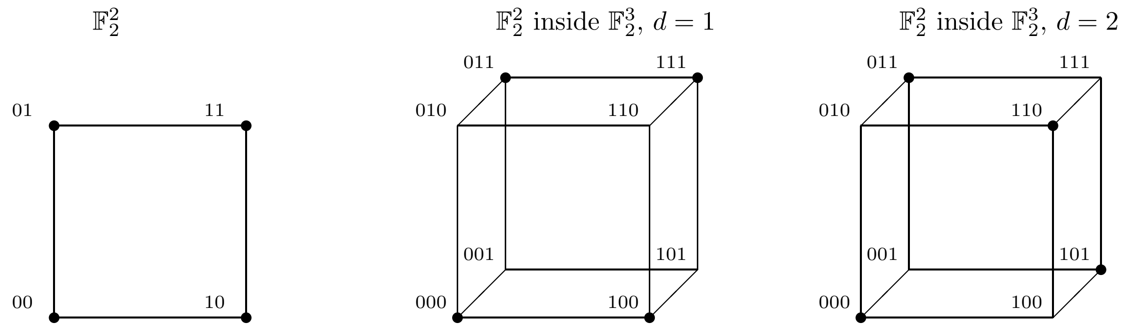
$\mathbb{F}_2^2$
$\mathbb{F}_2^2$ inside $\mathbb{F}_2^3$, $d = 1$
$\mathbb{F}_2^2$ inside $\mathbb{F}_2^3$, $d = 2$
01
11
00
10
011
111
010
110
001
101
000
100

Figure 4. Selected embeddings of $\mathbb{F}_2^2$ into $\mathbb{F}_2^3$

The rightmost embeddings in figures 3 and 4 are visibly the highest-distance 1-dimensional and 2-dimensional subspaces, respectively, of $\mathbb{F}_2^3$. Here $q = 2$, $n = 3$ and $k = 1$ or 2. For higher n, k and q, though, it is not immediately obvious how to spread out code words in this maximum-distance manner. The encoding problem consists in large part of finding a way of constructing such embeddings such that all code words are as far apart from one another as possible. This problem clearly is combinatorial in nature. Another technique for constructing such embeddings, using algebraic geometry, is explored starting in chapter 5.

3.5. Bounds on the Minimum Distance

Several upper and lower bounds are available in the literature ([Wal], [MS]). In particular, the following is easy to prove, though not sharp: some but certainly not all codes meet this bound.

Proposition 3.5.1 (Singleton bound). *Let C be a linear code of length n, dimension k, and minimum distance d over $\mathbb{F}_q$. Then*

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

Proof. Following [Wal], § 2.1, let W be the subspace of $\mathbb{F}_q^n$ consisting of vectors for which all but the first $d-1$ elements are zero. Clearly, $\dim(W) = d - 1$. Since all vectors in W have weight at most $d-1$, $W \cap C = \{\mathbf{0}\}$. From linear algebra,

$$\dim(W + C) = \dim(W) + \dim(C) = d - 1 + k \leq n.$$

□

3.6. Rate, Relative Minimum Distance, and Asymptotics

Definition 3.6.1. The *rate* of a code is the ratio $R = k/n$. For example, the n-bit repetition codes have rate $R = 1/n$: as n increases, R approaches zero.

Definition 3.6.2. The *relative minimum distance* of a code is the ratio $\delta = d/n$. For example, the repetition codes have relative minimum distance $\delta = n/n = 1$.

Definition 3.6.3. The values R, δ, and q are called the *asymptotic code parameters* for a family of codes.

For example, the parity-check codes have rate $R = (n-1)/n$, which approaches 1 as n increases, and relative minimum distance $2/n$, which approaches 0 as n increases. Of course, R and δ are both confined to the unit interval. One says that *asymptotically* (as n gets big) the repetition-code family has $R = 0$ and $\delta = 1$; asymptotically the parity-check family has $R = 1$ and $\delta = 0$.

The repetition codes have good error-correcting ability. However, the drawback is that most of the transmitted data is redundant: only one of every n bits is actual data. The parity-check codes, on the other hand, add just a single redundant bit, but tolerate few errors. These extreme cases motivate the following definition.

Definition 3.6.4. A *good code* (really, a good family, but "good code" is standard in the literature) is one for which R and δ are bounded away from 0 and 1.

For engineering reasons, it is also desirable for codes to have large block length [HvLP]. Since blocks are n-tuples over $\mathbb{F}_q$, long blocks may be obtained by increasing either q or n. The former requires more complicated circuitry to do arithmetic over larger finite fields, so the latter is preferred.

Definition 3.6.5. A *long code* is one for which n is large relative to q.

Code length is one key advantage of the algebraic-geometry codes discussed starting in chapter 5.

3.7. Encoding and the Generator Matrix

It has been assumed up to this point that a k-dimensional linear code C is a subspace of $\mathbb{F}_q^n$. Furthermore, it is now assumed that $\mathbb{F}_q^k$ is mapped to C by an injective linear transformation. The advantage of using a linear transformation is that, instead of needing a list of q^k images for the elements of $\mathbb{F}_q^k$, one needs only the images of k basis vectors to fully specify the map from $\mathbb{F}_q^k$ into C.

Such a linear transformation exists for any linear code: since $\mathbb{F}_q^k$ and C are vector spaces of the same dimension over the same field, an isomorphism exists. To obtain it explicitly if only C is given, form a tall matrix the rows of which are all the vectors of C, then row-reduce and discard zero rows. The result is a basis for C. Then, send the ith standard basis vector in $\mathbb{F}_q^k$ to the ith basis vector of C. Regardless of how it is obtained, the result is a *generator matrix*

$$G : \mathbb{F}_q^k \to \mathbb{F}_q^n$$

where C is the image of G in $\mathbb{F}_q^n$. For convenience later on, as well as consistency with the literature, G is written as a $k \times n$ matrix. To encode the message word $\boldsymbol{m}$, one writes $\boldsymbol{m}G$ rather than $G\boldsymbol{m}$. No explicit notational distinction is made in this paper between column and row vectors: the form is clear from the context.

For example, for the 5-bit repetition code one obtains

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

For the 5-bit parity-check code, one wants

$$\begin{bmatrix} a, & b, & c, & d, & a+b+c+d \end{bmatrix} = \begin{bmatrix} a, & b, & c, & d \end{bmatrix} G$$

where

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

Generator matrices are in general non-unique. For example, given a generator matrix with two or more rows, one may permute the rows of G to obtain a different matrix G' such that $\mathbb{F}_q^k G' = \mathbb{F}_q^k G$.

3.8. The Parity-Check Matrix

For a linear code C, encoding is easy: it is simply matrix multiplication. Decoding, and doing so efficiently, is a harder problem. In fact, there have been codes which were published before any decoding algorithm was known, and one area of current research is to develop improved decoding algorithms for existing codes.

Below, it will be useful to find a so-called *parity-check matrix*, H, such that C is precisely the kernel of H. (The terminology originally comes from parity-check codes, but it is a poor choice of words: all linear codes, not just the parity-check ones, have a parity-check matrix.) That is, $H\boldsymbol{v}$ should be zero if and only if $\boldsymbol{v} \in C$. By the rank-nullity theorem, H may be written as an $(n-k)\times n$ matrix, of rank $n-k$. Unlike with G, H conventionally is taken to operate by pre-multiplication: one writes $H\boldsymbol{v}$, not $\boldsymbol{v}H$. Before a technique to construct such a matrix is presented, some terminology is defined.

Definition 3.8.1. The *dual code* of C, written $C^\perp$, is the set of vectors in $\mathbb{F}_q^n$ which are orthogonal to all vectors of C, using the standard dot product. That is,

$$C^\perp = \{\boldsymbol{v} \in \mathbb{F}_q^n : \boldsymbol{u} \cdot \boldsymbol{v} = 0 \text{ for all } \boldsymbol{u} \in C\}.$$

Remark 3.8.2. The term *dual code* here has nothing to do with the term *dual space* from linear algebra. A dual code is usually referred to in linear algebra as a *perpendicular space* or an *orthogonal complement.*

Remark 3.8.3. The Hamming weight is a vector-space norm, if one defines $|c|$ on $\mathbb{F}_q$ to have value 0 when $c = 0$, 1 otherwise. If the standard dot product is used, then $\mathbb{F}_q^n$ satisfies all the axioms for an inner product space except for the positive-definiteness of the dot product. For example, if $\mathbb{F}_q$ has characteristic 2, the non-zero vector $(1, 1)$ dotted with itself is $1 + 1 = 0$. Note that the Hamming weight is computed in $\mathbb{Z}$: it is the number of non-zero coordinates in a vector. However, the dot product is computed in $\mathbb{F}_q$. Thus the Hamming weight and Hamming distance are positive definite, while the dot product is not. This means that inner-product-space results such as $\mathbb{F}_q^n = C \oplus C^\perp$ do not apply: the intersection of a subspace and its perpendicular space may contain more than just the zero vector. In fact, a code may be *self dual*, i.e. $C = C^{\perp}$. For example, $\{00, 11\}$ is a self-dual subspace of $\mathbb{F}_2^2$. As is shown in proposition 3.8.4, a self-dual code must have even n, and k must be $n/2$.

Proposition 3.8.4. *If C is a k-dimensional subspace of $V = \mathbb{F}_q^n$, then $\dim(C^{\perp}) = n - k$.*

Proof. The proof uses systematic and equivalent matrices, two concepts which are not developed in this paper since they are not needed outside this proof. See [VvO], theorem 3.3. □

Proposition 3.8.5. *If C is a k-dimensional subspace of $V = \mathbb{F}_q^n$, then $(C^{\perp})^{\perp} = C$.*

Proof. Let C be a k-dimensional subspace of $V = \mathbb{F}_q^n$. First, let $\boldsymbol{u} \in C$. For $\boldsymbol{u}$ to be in $(C^{\perp})^{\perp}$, $\boldsymbol{u} \cdot \boldsymbol{v}$ must be 0 for all $\boldsymbol{v} \in C^{\perp}$. Let $\boldsymbol{v}$ be arbitrary in $C^{\perp}$. Then $\boldsymbol{v} \cdot \boldsymbol{c} = 0$ for all $\boldsymbol{c} \in C$. In particular, this holds for $\boldsymbol{c} = \boldsymbol{u}$. Thus $\boldsymbol{u} \in (C^{\perp})^{\perp}$ and therefore $C \subseteq (C^{\perp})^{\perp}$.

For the reverse inclusion, since $\dim(C) = k$, $\dim(C^{\perp}) = n - k$ by proposition 3.8.4, and likewise $\dim((C^{\perp})^{\perp}) = k$. Since the vector space C is contained in $(C^{\perp})^{\perp}$ and they both have the same dimension, they are equal. □

Since G is already obtained, it remains to actually compute a matrix for H. Suppose that the problem were reversed, i.e. if H were already obtained, how would G be computed? Since the kernel of H is the image of G, which is C, one could just compute the kernel basis of H. This is a standard elementary linear algebra problem: G would have rows equal to the elements of that basis.

The following proposition shows that the generator matrix of $C^{\perp}$ is H and the parity-check matrix of $C^{\perp}$ is G. That is, $C^{\perp}$'s G and H are swapped from C's. Also $(C^{\perp})^{\perp}$ is just C by proposition 3.8.5. G is given, which is C's generator matrix as well as $C^{\perp}$'s parity-check matrix. The kernel basis of G is the generator matrix for $C^{\perp}$, which is also the parity-check matrix for C. This means that not only can G be obtained by computing a kernel basis of an H, but vice versa as well.

Proposition 3.8.6 (GH-perp). *Let C have generator matrix G and parity-check matrix H. Then $C^{\perp}$ has generator matrix H and parity-check matrix G.*

Proof. Recall the convention that a generator matrix acts by post-multiplication and that a parity-check matrix acts by pre-multiplication. So in this role, H maps $\mathbb{F}_q^{n-k}$ to $\mathbb{F}_q^n$ by sending $\boldsymbol{z}$ to $\boldsymbol{z}H$, and G maps $\mathbb{F}_q^n$ to $\mathbb{F}_q^k$ by sending $\boldsymbol{v}$ to $G\boldsymbol{v}$. To avoid confusion (only for the duration of this proof) the notation $\cdot G$ is used for the linear transformation $G : \mathbb{F}_q^k \to \mathbb{F}_q^n$ acting by post-multiplication and $G\cdot$ for $G : \mathbb{F}_q^n \to \mathbb{F}_q^k$ acting by pre-multiplication. Likewise, $H\cdot$ represents $H : \mathbb{F}_q^n \to \mathbb{F}_q^{n-k}$ and $\cdot H$ for $H : \mathbb{F}_q^{n-k} \to \mathbb{F}_q^n$. Plain G and H refer to the matrices without respect to a linear transformation.

The following short exact sequences are desired:

$$\begin{tikzcd} 0 \arrow[r] & \mathbb{F}_q^k \arrow[r, "\cdot G"] & \mathbb{F}_q^n \arrow[r, "H\cdot"] & \mathbb{F}_q^{n-k} \arrow[r] & 0 \end{tikzcd}$$

$$\begin{tikzcd} 0 & \mathbb{F}_q^k \arrow[l] & \mathbb{F}_q^n \arrow[l, "G\cdot"'] & \mathbb{F}_q^{n-k} \arrow[l, "\cdot H"'] & 0 \arrow[l] \end{tikzcd}$$

with $\mathrm{im}(\cdot G) = C = \ker(H\cdot)$ and $\mathrm{im}(\cdot H) = C^\perp = \ker(G\cdot)$. The short exactness means that $\cdot G$ and $\cdot H$ are 1-1, while $H\cdot$ and $G\cdot$ are onto. Thus, it suffices to show: (1) $\mathrm{im}(\cdot H) = C^\perp$; (2) $\cdot H$ is 1-1, (3) $\ker(G\cdot) = C^\perp$, and (4) $G\cdot$ is onto. Now, it has already been seen that the matrix G has rank k and H has rank $n - k$. Since row rank equals column rank, (4) follows from (3) by the rank-nullity theorem. Likewise, (2) follows from (1) since $C^\perp$ has dimension $n - k$.

To prove (3), first let $\boldsymbol{v} \in C^\perp$. The rows of G form a basis for C; let $\boldsymbol{g}_i$ be the ith row of G, for $i = 1, \ldots, k$, where each $\boldsymbol{g}_i$ is a vector of length n (since it is in $\mathbb{F}_q^n$). Also let $\boldsymbol{v} = (v_1, \ldots, v_n)$. The matrix-times-vector multiplication $G \cdot \boldsymbol{v}$ consists of dot products of $\boldsymbol{v}$ with the rows of G:

$$G \cdot \boldsymbol{v} = \begin{bmatrix} \boldsymbol{g}_1 \\ \vdots \\ \boldsymbol{g}_k \end{bmatrix} \begin{bmatrix} v_1 \\ \vdots \\ v_n \end{bmatrix} = \begin{bmatrix} \boldsymbol{g}_1 \cdot \boldsymbol{v} \\ \vdots \\ \boldsymbol{g}_k \cdot \boldsymbol{v} \end{bmatrix}.$$

Since each $\boldsymbol{g}_i$ is in C and since $\boldsymbol{v}$ is in $C^\perp$, all the dot products are zero and so $G \cdot \boldsymbol{v} = 0$.

Conversely, let $\boldsymbol{v} \in \ker(G\cdot)$. Then $G \cdot \boldsymbol{v} = 0$. Again, this product consists of dot products of rows of G with $\boldsymbol{v}$, so $\boldsymbol{g}_i \cdot \boldsymbol{v} = 0$ for all $\boldsymbol{g}_i$'s. Let $\boldsymbol{c}$ be an arbitrary element of C. Since the $\boldsymbol{g}_i$'s are a basis for C, $\boldsymbol{c} = \sum_{i=1}^k c_i \boldsymbol{g}_i$ for some c_i's in $\mathbb{F}_q$. Then

$$\boldsymbol{v} \cdot \boldsymbol{c} = \boldsymbol{v} \cdot \sum_{i=1}^k c_i \boldsymbol{g}_i = \sum_{i=1}^k c_i (\boldsymbol{v} \cdot \boldsymbol{g}_i) = 0.$$

Therefore $\boldsymbol{v} \in C^\perp$.

To prove that $\operatorname{im}(\cdot H) = C^{\perp}$, notice in general that when a matrix X acts on a standard basis by $X\boldsymbol{\varepsilon}_i$, the image of that basis consists of the columns of X. Likewise, when X acts on a standard basis by $\boldsymbol{\varepsilon}_i X$, the image of that basis consists of the rows of X. It suffices to show that the rows of H are a basis for $C^{\perp}$. Remember that H was set up to check the elements of C, and since G has rows forming a basis for C, necessarily

$$HG^t = 0.$$

This means that the rows of H are orthogonal to the rows of G, which shows that the rows of H are in $C^{\perp}$. Since H has rank $n-k$, the image of the standard basis for $\mathbb{F}_q^{n-k}$ under $\cdot H$ is linearly independent, and $\operatorname{im}(\cdot H)$ must be all of $C^{\perp}$. $\square$

Example 3.8.7. The 5-bit repetition code has generator matrix

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

Then the kernel basis, in row-echelon form, is computed to be

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

Intuitively, this makes sense: recalling that arithmetic here is being done mod 2, this means that $H\boldsymbol{v}$ is 0 only when $\boldsymbol{v}$ has all coordinates the same. The two possible cases are 00000 and 11111, which are precisely the code words of the 5-bit repetition codes.

Example 3.8.8. The 5-bit parity-check code has generator matrix

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

One then computes

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

Intuitively, this also makes sense: pre-multiplying $\boldsymbol{v}$ by H just adds up the bits of $\boldsymbol{v}$ mod 2. The result is zero precisely when $\boldsymbol{v}$ has even parity, which is the case iff $\boldsymbol{v}$ is in C.

As an added bonus, since the first G is the same as the second H and vice versa, the repetition and parity-check families are now seen by example to be duals of one another.

3.9. Decoding

One may think of a transmission error as a vector $\boldsymbol{e}$ which is added to the code word $\boldsymbol{u}$, resulting in a received word $\boldsymbol{v}$. It is the task of the decoding algorithm to compute an estimate $\hat{\boldsymbol{e}}$ for $\boldsymbol{e}$, as illustrated in figure 5 (taken from [MS]). Once $\hat{\boldsymbol{e}}$ has been computed, one then computes $\hat{\boldsymbol{u}} = \hat{\boldsymbol{v}} - \hat{\boldsymbol{e}}$. Then, $\hat{\boldsymbol{m}}$ may be found by solving the linear system of equations $\hat{\boldsymbol{u}} = \hat{\boldsymbol{m}}G$ for $\hat{\boldsymbol{m}}$.

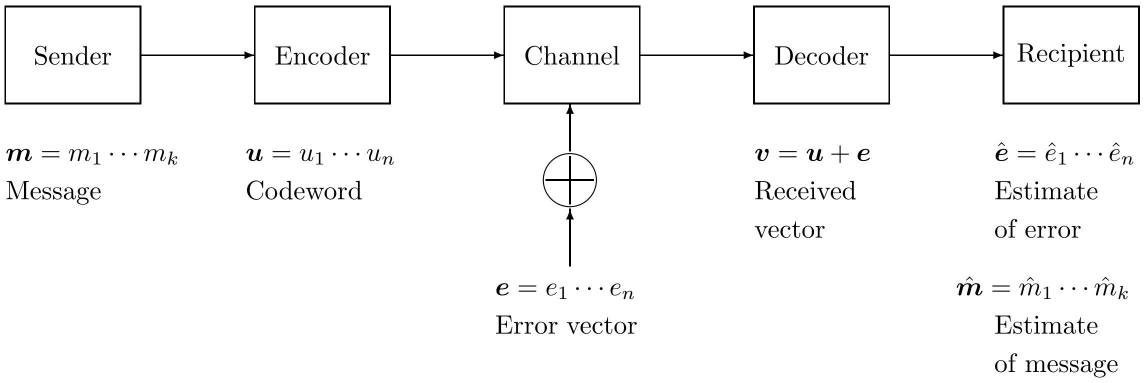
Sender
Encoder
Channel
Decoder
Recipient
$\boldsymbol{m} = m_1 \cdots m_k$
Message
$\boldsymbol{u} = u_1 \cdots u_n$
Codeword
$\boldsymbol{v} = \boldsymbol{u} + \boldsymbol{e}$
Received vector
$\hat{\boldsymbol{e}} = \hat{e}_1 \cdots \hat{e}_n$
Estimate of error
$\boldsymbol{e} = e_1 \cdots e_n$
Error vector
$\hat{\boldsymbol{m}} = \hat{m}_1 \cdots \hat{m}_k$
Estimate of message

Figure 5. The coding theorist's coat of arms

Definition 3.9.1. A *decoding error* occurs when the decoder computes $\boldsymbol{e} \neq \hat{\boldsymbol{e}}$. A *decoding failure* occurs when the decoder fails to compute $\hat{\boldsymbol{e}}$.

The simplest decoding algorithm is to tabulate, as in figures 1 and 2, the nearest code word to each possible message word. Since the amount of storage space required is proportional to the number of code words, this technique is practical only for very small codes. Other simple algorithms include the standard-array and step-by-step decoding algorithms. See [MS], [VvO] for more information.

CHAPTER 4

Algebraic Geometry

Algebraic geometry is a large subject. Here, the definitions necessary for chapters 5-8 are collected. For more information see [Wal], [Pre], [Sch], appendix A of [ST], and chapters I-II of [Sil]. Note that the latter three have no particular emphasis on finite fields.

4.1. Algebraic Closure of $\mathbb{F}_q$

Algebraic geometry is largely concerned with algebraically closed fields. The algebraic closure of $\mathbb{F}_q$ is ([DF], § 14.3)

$$\overline{\mathbb{F}}_q = \bigcup_{i=1}^{\infty} \mathbb{F}_{q^i}.$$

Computation in this infinite field requires embedding finite smaller fields into larger ones. For example, given $\alpha \in \mathbb{F}_q^r$ and $\beta \in \mathbb{F}_q^s$, one may compute $t = \operatorname{lcm}(r, s)$ and find the images of α, β in $\mathbb{F}_q^t$. Then, $\alpha + \beta$, $\alpha\beta$, etc. may be computed in $\mathbb{F}_q^t$ as described in section 2.2. The embeddings may be quickly done as described in remark 2.2.1, using root charts such as those in table 6. However, $\overline{\mathbb{F}}_q$ is used only for theoretical reasons in this paper; explicit computations in this paper are done using only finite extensions of $\mathbb{F}_q$.

4.2. Affine and Projective Spaces

Let K be a field with algebraic closure $\overline{K}$. For most of this paper, $K = \mathbb{F}_q$ and $\overline{K} = \overline{\mathbb{F}}_q$. However, some motivating examples are given using $K = \overline{K} = \mathbb{C}$, or $K = \mathbb{Q}, \overline{K} = \overline{\mathbb{Q}}$.

Definition 4.2.1. Affine n-space, written $\mathbb{A}^n(K)$, consists of all ordered n-tuples of elements of K.

In the literature, a plain $\mathbb{A}^n$ stands for $\mathbb{A}^n(\overline{K})$; this paper always uses the explicit latter notation.

Definition 4.2.2. Let $S = \mathbb{A}^{n+1}(K) \setminus \{(0, \ldots, 0)\}$ consist of all ordered $n+1$-tuples of elements of K with not all elements zero. Form the equivalence relation $\sim$ by $(a_0, \ldots, a_n) \sim (b_0, \ldots, b_n)$ iff $(a_0, \ldots, a_n) = \lambda(b_0, \ldots, b_n)$ for some $\lambda \in \overline{K}$. Projective n-space, written $\mathbb{P}^n(K)$, is $S/\sim$.

Intuitively, $\mathbb{P}^n(K)$ consists of all distinct straight lines through the origin in $(n+1)$-dimensional affine space.

Notation 4.2.3. The equivalence class of a point $(a_0, \ldots, a_n)$ in $\mathbb{P}^n(K)$ is written $[a_0, \ldots, a_n]$.

Note that λ-scaling is up to elements of $\overline{K}$, not just K. For example, $[1, 1]$ and $[\sqrt{2}, \sqrt{2}]$ are both elements of $\mathbb{P}^1(\mathbb{Q})$. For this paper, however, it is assumed that elements of $\mathbb{P}^n(K)$ are written with all coordinates in K. This is always possible for the following reason.

Convention 4.2.4. Since there may be more than one representative for each equivalence class in $\mathbb{P}^n(K)$, unique representations are obtained by forcing the rightmost non-zero coordinate to be a 1. For example, $[2, 3, 5]$ in $\mathbb{P}^2(\mathbb{F}_8)$ is written, dividing through by 5, as $[4, 6, 1]$.

Example 4.2.5. $\mathbb{A}^2(\mathbb{F}_2)$ consists of the following four elements:

$$(0,0), \quad (0,1), \quad (1,0), \quad (1,1).$$

$\mathbb{P}^2(\mathbb{F}_2)$ consists of the following seven elements:

$$\begin{aligned}&[0,0,1], \quad [0,1,1], \quad [1,0,1], \quad [1,1,1],\\ &[0,1,0], \quad [1,1,0],\\ &[1,0,0].\end{aligned}$$

In general, $\mathbb{A}^n(\mathbb{F}_q)$ has q^n elements, while $\mathbb{P}^n(\mathbb{F}_q)$ has an additional $q^{n-1} + \ldots + q + 1$ elements. Observe that $\mathbb{P}^n(K)$ contains a copy of $\mathbb{A}^n(K)$ along with a copy of $\mathbb{P}^{n-1}(K)$. In the above example, the former are the elements for which a 1 appears in the final coordinate; the latter are the elements for which a 0 appears in the final coordinate. The same criterion could be applied, however, to any coordinate. For example, the elements of $\mathbb{P}^2(\mathbb{F}_2)$ might be written

$$\begin{aligned}
&[0,1,1], \quad [0,1,0], \quad [1,1,1], \quad [1,1,0],\\
&[0,0,1], \quad [1,0,1],\\
&[1,0,0].
\end{aligned}$$

This motivates the following definition.

Definition 4.2.6. Fix a coordinate position i of $\mathbb{P}^n(K)$. The elements in the copy of $\mathbb{A}^n(K)$ with non-zero ith coordinate are called *affine points*; the elements in the copy of $\mathbb{P}^{n-1}(K)$ with zero ith coorindate are called *points at infinity.*

Unless otherwise specified, it is assumed that the last coordinate position is the one used to distinguish the affine points from the points at infinity.

Definition 4.2.7. The $n+1$ copies of $\mathbb{A}^n(K)$ obtained by dehomogenizing with respect to each coordinate are called the *affine components* of $\mathbb{P}^n(K)$.

Remark 4.2.8. Let $P = [a_0, \ldots, a_n] \in \mathbb{P}^n(K)$. For any $a_i = 0$, P is a point at infinity with respect to dehomogenization at the ith coordinate. For any $a_i \neq 0$, of which there is at least one since projective points have at least one non-zero coordinate, P is an element of the affine component with respect to dehomogenization at the ith coordinate. In particular, if all a_i's are non-zero, then P appears in all affine components.

4.3. Polynomials and Rational Functions

Definition 4.3.1. Let $f(x_0, \ldots, x_{n-1}) \in K[x_0, \ldots, x_{n-1}]$. The *homogenization* of f in $K[x_0, \ldots, x_n]$ is $F(X_0, \ldots, X_n) = X_n^d f(X_0/X_n, \ldots, X_{n-1}/X_n)$ where $d = \deg(f)$.

Intuitively, F is obtained from f by capitalizing, then inserting X_n's to bring each monomial up to the degree of f.

Example 4.3.2. If $f(x,y) = x + xy + y^3$, then $F(X,Y,Z) = XZ^2 + XYZ + Y^3$.

Definition 4.3.3. A polynomial $F(X_0, \ldots, X_n) \in K[X_0, \ldots, X_n]$ is said to be *homogeneous* if all its terms have the same degree.

Note that this implies $F(\lambda X_0, \ldots, \lambda X_n) = 0$ iff $\lambda F(X_0, \ldots, X_n) = 0$. Observe that the homogenization of a polynomial is homogeneous. Likewise, a homogeneous rational function is the quotient of homogeneous polynomials, of the same degree. The numerator and denominator must have the same degree in order to preserve the *scaling property* for projective points, i.e. to ensure that $F([a_0, \ldots, a_n]) = F([\lambda a_0, \ldots, \lambda a_n])$ for all non-zero $\lambda \in \overline{K}$.

Definition 4.3.4. Given a polynomial $F(X_0, \ldots, X_n) \in K[X_0, \ldots, X_n]$, the *dehomogenization* of F with respect to X_i is the polynomial $f(x_0, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n)$ obtained by setting X_i to 1 and replacing the remaining X_j's with x_j's.

For example, if $F(X, Y, Z) = XZ^2 + XYZ + Y^3$, then the dehomogenization of F with respect to Z is $f(x, y) = x + xy + y^3$. The dehomogenization of F with respect to Y is $f(x, z) = xz^2 + xz + 1$.

Remark 4.3.5. The zeroes of f are the same as those of F on the corresponding affine component. The zeroes at infinity are missing from the dehomogenization, but these may be obtained by setting $X_i = 0$ in F.

Definition 4.3.6. An ideal I of $K[X_0, \ldots, X_n]$ is said to be a *homogeneous ideal* if it is generated by homogeneous polynomials.

By the Hilbert basis theorem ([Sil]), any ideal of $K[X_0, \ldots, X_n]$ is finitely generated. A rational expression $F(X_0, \ldots, X_n) = G(X_0, \ldots, X_n)/H(X_0, \ldots, X_n)$ is simply an element of the field $K(X_0, \ldots, X_n)$. However, $F(X_0, \ldots, X_n)$ may also be thought of as a *rational function* from $\mathbb{P}^n(K)$ to $\mathbb{P}^1(K) \cup \{0\}$, via the evaluation homomorphism.

4.4. Curves

Much of this chapter is owed to [Sil], but this section in particular is almost verbatim from [Sil].

Definition 4.4.1. Given a homogeneous ideal I of $\overline{K}[X_0, \ldots, X_n]$, V_I is the set of all points of $\mathbb{P}^n(\overline{K})$ which are zeroes of all polynomials in I. Any such V_I is called a *algebraic set.*

Definition 4.4.2. The *homogeneous ideal* $I(V)$ of an algebraic set V is the set of all polynomials in $\overline{K}[X_0, \ldots, X_n]$ which have value zero on all points of V. Additionally, one writes $I(V/K) = I(V) \cap K[X_0, \ldots, X_n]$.

Definition 4.4.3. An algebraic set V is said to be *defined over* K if $I(V)$ may be generated by homogeneous polynomials in $K[X_0, \ldots, X_n]$.

Definition 4.4.4. If V is defined over K, then the set of K*-rational points* of V, written $V(K)$, consists of those points in V with coordinates in K. That is, $V(K) = V(\overline{K}) \cap \mathbb{P}^n(K)$.

Definition 4.4.5. A *variety* is an algebraic set whose homogeneous ideal $I(V)$ is a prime ideal in $\overline{K}[X_0, \ldots, X_n]$.

This is technically the definition of a projective variety; affine varieties may be defined similarly. For this paper, the projective kind is used everywhere unless stated otherwise, so the terminology is shortened.

Informally, this means the following: algebraically, the ideal $I(V)$ does not factor non-trivally. In particular, when a variety is defined by a single polynomial F, as is the case in this paper, F does not factor non-trivially over $\overline{K}$. Geometrically, the graph of zeroes is not the union of two curves. For example $(x^2 + y^2 - 1)yx$ factors over $\mathbb{R}$ into the product of three polynomials; the graph of zeroes of this polynomial consists of the union of the unit circle, the x axis, and the y axis. Thus, the zeroes of this polynomial do not form a variety.

Recall that $\overline{K}[X_0, \ldots, X_n]$ is a unique factorization domain, hence its polynomials are prime iff they are irreducible. In the case where $I(V)$ is generated by a single polynomial F (in which case V may be written as V_F), in the literature F is often said to be *absolutely irreducible.* The "absolute" part of this term arises since F is irreducible not only in $K[X_0, \ldots, X_n]$, but also in $\overline{K}[X_0, \ldots, X_n]$. A sufficient condition for absolute irreducibility is shown in proposition 4.10.3.

Remark 4.4.6. In [Sil], the *dimension* of a variety V is defined to be the transcendence degree of the function field $\overline{K}(V)$, defined below, over $\overline{K}$. In particular, if V consists of the zeroes of a single non-constant homogeneous polynomial $F(X_0, \ldots, X_n)$, then the dimension of V is $n-2$.

Definition 4.4.7. A *curve* is a variety of dimension one.

Definition 4.4.8. A *plane curve* is a curve whose homogeneous ideal is an ideal of $\overline{K}[X, Y, Z]$.

From remark 4.4.6, a plane curve is necessarily defined by a single equation. The curves considered in this paper satisfy a final property, namely, smoothness.

Definition 4.4.9. A curve V is *non-singular* at a point P if the Jacobian matrix $(\partial F_i/\partial X_j)$, evaluated at P, has rank $n - \dim V$, where $F_1, \ldots, F_m$ is a finite generating set for $I(V)$. If V is non-singular at every point P of $\mathbb{P}^n(\overline{K})$, then V is said to be *non-singular* or *smooth*. In the case when V is defined by a single polynomial F (which is the case for this paper), then a point P of V is singular iff all its partials vanish at P.

For example, the circle in $\mathbb{A}^2(\mathbb{R})$ given by $x^2 + y^2 - 1 = 0$ homogenizes to $X^2 + Y^2 - Z^2 = 0$ in $\mathbb{P}^2(\mathbb{R})$. This has dimension 1, and is a curve. The partial derivatives are $2X$, $2Y$, and $-2Z$, respectively. These can be all zero only when $X = Y = Z = 0$, which is not a projective point. Thus, the circle is smooth over $\mathbb{R}$. Over a field with characteristic 2, it is not a variety since the defining polynomial is reducible: $x^2 + y^2 - 1 = x^2 + y^2 + 1 = (x + y + 1)^2$. On the other hand, the ball in $\mathbb{A}^3(\mathbb{R})$ given by $x^2 + y^2 + z^2 - 1 = 0$ homogenizes to $X^2 + Y^2 + Z^2 - W^2 = 0$. This is a variety of dimension 2 as long as $\text{char}(K) \neq 2$, but it is not a curve.

The *genus* of a curve is a non-negative integer associated to the curve, as defined in [Sil]. Computation of the genus is in general non-trivial; however, the following formula suffices for this paper. Let V be a smooth curve defined by a single polynomial F of degree d. The genus g of V satisfies the *Plücker formula*:

$$g = \frac{(d-1)(d-2)}{2}.$$

4.5. Enumeration of Points

Throughout this paper it is necessary to obtain a list of points on a curve over $\mathbb{F}_2$ or some extension field thereof. This enumeration may be done analytically, as is shown by the following example.

Example 4.5.1. Let E be defined by $f(x,y) = y^2 + y + x^3 + x + 1$ over $\mathbb{F}_2$, with homogenization $F(X,Y,Z) = Y^2Z + YZ^2 + X^3 + XZ^2 + Z^3$. Then $\frac{\partial F}{\partial X} = X^2 + Z^2$, $\frac{\partial F}{\partial Y} = Z^2$, and $\frac{\partial F}{\partial Z} = Y^2 + Z^2$. For $\frac{\partial F}{\partial Y}$ to be zero forces $Z = 0$, which in turn forces $X = Y = 0$. This remains true for any extension field of $\mathbb{F}_2$. Since no projective point has all coordinates equal to zero, E is non-singular. By the Plücker formula, E has genus 1. Since it visibly has the $\mathbb{F}_2$-rational point $[0,1,0]$, E is in fact an elliptic curve.

The points of $E(\mathbb{F}_2)$ are computed as follows. As discussed in remark 4.3.5, the points at infinity are found by setting $Z = 0$; the affine points are found by setting $Z = 1$. The former yields $X^3 = 0$, regardless of the degree of the extension $\mathbb{F}_q$ of $\mathbb{F}_2$. It remains to compute the affine points, given the dehomogenization $f(x,y) = y^2 + y + x^3 + x + 1$. The y part $y^2 + y = y(y+1)$ has value 0 at $y = 0,1$, while the x part $x^3 + x + 1$ has value 1 at $x = 0,1$. Therefore, $E(\mathbb{F}_2)$ has no affine points.

x	x^3	1	x^3+x+1	y	$y+1$	$y(y+1)$
0	0	1	1	0	1	0
1	1	1	1	1	0	0
2	1	1	2	2	3	1
3	1	1	3	3	2	1

Table 8. x and y parts for affine points of $E(\mathbb{F}_4)$

For curves in a small number of variables over a small finite field, instead of analyzing the curve one may simply test each point of $\mathbb{P}^n(\mathbb{F}_q)$. What this approach lacks in elegance it makes up for in simplicity, and in particular it is easily automated. This method is used to list points of $E(\mathbb{F}_4)$ and $E(\mathbb{F}_8)$. It was shown in the previous paragraph that $[0,1,0]$ is the only point at infinity on $E(\mathbb{F}_{2^r})$ for all positive r, so in fact one needs only to check the affine points. For affine points of

x	x^3	1	x^3+x+1	y	$y+1$	$y(y+1)$
0	0	1	1	0	1	0
1	1	1	1	1	0	0
2	3	1	0	2	3	6
3	4	1	6	3	2	6
4	5	1	0	4	5	2
5	6	1	2	5	4	2
6	7	1	0	6	7	4
7	2	1	4	7	6	4

Table 9. x and y parts for affine points of $E(\mathbb{F}_8)$

$\mathbb{F}_4$, one computes the x and y parts as shown in table 8, using the tables in section 2.2. For each value of x, one may simply tick off the values of y, if any, for which the x part equals the y part. For $x = 0$, $y = 2, 3$; for $x = 1$, $y = 2, 3$. These result in the projective points

$$[0, 2, 1], [0, 3, 1], [1, 2, 1], [1, 3, 1], [0, 1, 0].$$

For affine points of $\mathbb{F}_8$, one proceeds similarly, as shown in table 9. The projective points are

$$[2, 0, 1], [2, 1, 1], [3, 2, 1], [3, 3, 1], [4, 0, 1], [4, 1, 1],$$
$$[5, 4, 1], [5, 5, 1], [6, 0, 1], [6, 1, 1], [7, 6, 1], [7, 7, 1], [0, 1, 0].$$

4.6. Bounds on Point Counts

As discussed in section 3.6, better codes are obtained for curves with large numbers of rational points, which is possible only for curves of higher genus. The Hasse-Weil theorem ([Wal], chapter 7) in fact gives the following: Let V be a smooth curve of genus g over $\mathbb{F}_q$, and write $N = \#(\mathbb{F}_q)$. Then

$$|N - q - 1| \leq 2g\sqrt{q}.$$

A result of Serre [Ser] improves this slightly to

$$|N - q - 1| \leq g\lfloor 2\sqrt{q}\rfloor$$

where $\lfloor x \rfloor$, the *floor function* of x, denotes the largest integer less than or equal to x. Note in particular that the bounds of Hasse-Weil and Serre provide an exact point count, namely $q + 1$, for curves of genus 0. In general, one may view the Hasse-Weil and Serre bounds in the following way: The number of points on a curve over $\mathbb{F}_q$ is approximately $q+1$, plus or minus an amount which is small when g is small; the amount of uncertainty allowed by these bounds grows with the genus. Further discussion of bounds for the number of points on curves over finite fields may be found in [Iha] and [Lau].

Example 4.6.1. The curve given by the zeroes of $Y^2Z + YZ^2 + X^3 + XZ^2 + Z^3$ over $\mathbb{F}_2$ was seen to have genus g. The actual point counts are compared to the Serre bound in table 10.

L	$\#\mathbb{P}^2(L)$	$\#E(L)$	Serre minimum	Serre maximum
$\mathbb{F}_2$	7	1	1	5
$\mathbb{F}_4$	21	5	1	9
$\mathbb{F}_8$	73	13	4	14
$\mathbb{F}_{16}$	273	25	9	25
$\mathbb{F}_{32}$	1057	41	22	44
$\mathbb{F}_{64}$	4161	65	49	81
$\mathbb{F}_{128}$	16513	113	107	151
$\mathbb{F}_{256}$	65793	225	225	289
$\mathbb{F}_{512}$	262657	481	468	558
$\mathbb{F}_{1024}$	1049601	1025	961	1089

Table 10. Point counts vs. Serre bounds for E

4.7. Point Classes

Let L be a Galois extension of K, and let σ be an element of $G = \mathrm{Gal}(L/K)$. Let V be a curve with coefficients in a field K. Then $V(L)$ may have more points than does $V(K)$. For example, the circle $x^2 + y^2 + 1 = 0$ has no points at all over $K = \mathbb{R}$, but it has infinitely many points over $L = \mathbb{C}$.

Notation 4.7.1. Let $P = [a_0, \ldots, a_n] \in \mathbb{P}^n(L)$. Then $\sigma(P)$ denotes $[\sigma(a_0), \ldots, \sigma(a_n)]$.

Note that if $P = [\alpha_0, \ldots, \alpha_n]$ is a zero of F, then $\sigma(P) = [\sigma(\alpha_0), \ldots, \sigma(\alpha_n)]$ is also a zero of F.

Definition 4.7.2. The orbit of P under the action of G is called a *point class*. The cardinality of a point class is called the *degree* of the point class.

Example 4.7.3. The points on $E(\mathbb{F}_4)$ as shown in section 4.5 cluster into point classes

$$\{[0,2,1],[0,3,1]\}, \quad \{[1,2,1],[1,3,1]\}, \quad \{[0,1,0]\}.$$

The points on $E(\mathbb{F}_8)$ cluster into point classes

$$\{[2,0,1],[4,0,1],[6,0,1]\}, \quad \{[2,1,1],[4,1,1],[6,1,1]\},$$

$$\{[3,2,1],[5,4,1],[7,6,1]\}, \quad \{[3,3,1],[5,5,1],[7,7,1]\}, \quad \{[0,1,0]\}.$$

For future reference, some of these point classes are named as follows:

$$\mathcal{Q}_2 = \{[0,2,1],[0,3,1]\}, \quad \mathcal{Q}_3 = \{[2,0,1],[4,0,1],[6,0,1]\}.$$

4.8. Function Fields

Definition 4.8.1. Let V be a variety defined over a field K. The *coordinate ring* of V/K is

$$K[V] = \frac{K[X_0,\ldots,X_n]}{I(V/K)}.$$

This is simply the set of all polynomials, modulo the polynomial(s) defining V. Note that the coordinate ring is an integral domain since $I(V/K)$ is assumed to be prime. Thus it is possible to form a field of quotients.

Definition 4.8.2. The *function field* $K(V)$ of a curve V over K is the field of quotients of its coordinate ring.

The definition means that $K(V)$ is the subfield of $K(X_0,\ldots,X_n)$ consisting of $G(x)/H(x)$ such that:

(i) G, H are homogeneous of the same degree, so that the λ-scaling property applies.

(ii) Division by zero mod $I(V/K)$ is disallowed: $H \notin I(V/K)$.

(iii) Equality of fractions mod $I(V/K)$: $G/H \sim G'/H' \iff GH' - G'H \in I(V)$.

4.9. Intersection Multiplicity

Bezout's theorem is presented in section 4.10. This key result shows that two non-overlapping projective plane curves over an algebraically closed field intersect in exactly de points, where d and e are the degrees of the polynomials defining the curves. However, this only works correctly when intersection points are counted with multiplicity. For example, one would expect the parabola $y = x^2$ and the line $y = 0$, which are $YZ = X^2$ and $Y = 0$ in projective coordinates, to have a double intersection point at the origin. Intersection multiplicity may be defined quite generally [Har], but here all that is needed is intersection multiplicity for plane projective curves, one of which is a line. In contrast to the rest of this paper, the affine approach is taken here: intersection multiplicity for projective plane curves is accomplished by examining each affine component. This is done in order to facilitate a particular computation shown below. By remark 4.2.8, all points of the projective plane are contained in some affine component.

Affine varieties are defined over affine spaces in a manner analogous to projective varieties (section 4.4). Let V and W be projective plane curves, defined by polynomials $\Phi[X, Y, Z]$ and $\Psi[X, Y, Z]$. Let v, defined by $\phi(x, y)$, and w, defined by $\psi(x, y)$, be the affine components of V and W after dehomogenization with respect to one variable, which without loss of generality is taken to be Z.

Definition 4.9.1. Let p be a point on v. Define

$$\mathcal{O}_p = \left\{ f = \frac{g}{h} \in K(x, y) : h(p) \neq 0 \right\}.$$

This is a local ring, the localization of $K[x, y]$ at p, which contains all of $K(x, y)$ except those functions whose denominator is zero at p.

Definition 4.9.2. The *intersection multiplicity* of v and w at a point p is defined ([ST], appendix A) to be

$$\mathrm{IM}(v \cap w, p) = \left(\dim_{\overline{K}} \left(\frac{\mathcal{O}_p}{\langle \phi, \psi \rangle_p} \right) \right)$$

where the subscript on $\langle \phi, \psi \rangle_p$ indicates an ideal of $\mathcal{O}_p$.

This is computed by first considering $K[x,y]/\langle \phi, \psi \rangle$, then localizing at p afterward. The computation is particularly simple if w is a line. For the moment, this special case is all that is needed. Let ℓ be a line, defined by $\lambda(x,y)$. Since ϕ is a polynomial in two variables and λ contains a term of degree 1, one variable, say y, may be eliminated. The ideal $\langle \phi, \lambda \rangle$ is then singly generated as $\langle u(x) \rangle$. Over a splitting field S of K, where $S \subset \overline{K}$, $u(x)$ factors as

$$u(x) = \prod_{j=1}^{n} (x - \alpha_j)^{e_j}.$$

For each α_j, the other coordinate β_j may be computed using λ, yielding a point $p = (\alpha_j, \beta_j)$. In the localization at p, the $x - \alpha_i$ factors become units wherever $i \neq j$, and

$$\frac{\mathcal{O}_p}{\langle \phi, \lambda \rangle_p} = \frac{\mathcal{O}_p}{\langle (x - \alpha_j)^{e_j} \rangle}$$

which is a vector space of dimension e_j over $\overline{K}$. Intersection multiplicities are thus reduced to multiplicities of roots of univariate polynomials.

Note that a projective line may not become an affine line when dehomogenized. For example, the projective line $X = 0$ becomes the empty set upon dehomogenization at X. More generally, [Sil], there are only two possibilities for a projective variety: (1) dehomogenization produces an empty affine variety, or (2) dehomogenization followed by rehomogenization recovers the original projective variety entirely.

Definition 4.9.3. Let V be a projective plane curve and L be a projective plane line. If the dehomogenization of L fails to remain a plane line upon dehomogenization with respect to a variable, then that affine component is said to be *inadmissible* for L.

Example 4.9.4. Let $K = \mathbb{Q}$, let V defined by $YZ^2 = X^3 - XZ^2$, i.e. the homogenization of $y = x^3 - x$, and let L be defined by $Y = 2X + 2Z$, i.e. the homogenization of $y = 2x + 2$. By elementary calculus, L is seen to be tangent to V at $[-1, 0, 1]$, and intersects V additionally at $[2, 6, 1]$. One would expect multiplicity two at the former point and multiplicity one at the latter point. Both points appear in dehomogenization with respect to Z since they have final coordinate 1, and so one does not need to dehomogenize with respect to X or Y. The affine plane curve v is defined by $\phi(x,y) = y - x^3 + x$ and ℓ is defined by $\lambda(x,y) = y - 2x - 2$. Then

$$\langle \phi, \lambda \rangle = \langle y - x^3 + x, y - 2x - 2 \rangle = \langle x^3 - 3x - 2 \rangle = \langle (x+1)^2(x-2) \rangle.$$

Zeroes of this last polynomial are $x = -1, 2$, corresponding to $y = 0, 6$ on v. In the localization at $p = (-1, 0)$, $x - 2$ is permissible in the denominator of elements of $\mathcal{O}_p$ but $x + 1$ is not, and so $x - 2$ is a unit. Thus, $\langle \phi, \lambda \rangle_p = \langle (x+1)^2 \rangle$. In the localization at $q = (2, 6)$, $x + 1$ is a unit and so $\langle \phi, \lambda \rangle_q = \langle x - 2 \rangle$. Then $\mathcal{O}_p / \langle (x+1)^2 \rangle_p$ is generated as a $\overline{K}$-vector space by $\{1, x\}$ and $\mathcal{O}_q / \langle x - 2 \rangle_q$ is generated as a $\overline{K}$-vector space by $\{1\}$. Thus the intersection multiplicities are in fact 2 and 1, respectively, as expected.

Example 4.9.5. Let V be defined by $YZ - X^2$ and L by X. Dehomogenization with respect to Z gives $\phi(x,y) = y - x^2$ and $\lambda(x,y) = x$, leading to the intersection point $[0,0,1]$. Likewise, dehomogenization with respect to Y yields the other intersection point $[0,1,0]$. However, dehomogenization with respect to X gives $\phi(y,z) = yz - 1$ and $\lambda(y,z) = 1$. The latter has no zeroes at all, so dehomogenization with respect to X is inadmissible for L.

The above discussion is summarized in the following pair of algorithms.

Algorithm 4.9.6. Let v be an affine plane curve and ℓ be an affine plane line, defined by $\phi(x,y)$ and $\lambda(x,y)$, respectively. To compute the intersection points and multiplicities for v and ℓ:

- Use λ to eliminate one variable. Since λ has degree 1, it may be solved for one variable or the other, say y. Then a polynomial $u(x)$ is obtained.
- Split $u(x)$ into linear factors over some extension field of K, i.e. $u(x) = \prod_{j=1}^{n} (x - \alpha_j)^{e_j}$.
- For each linear factor α_j, compute β_j using $\lambda(\alpha_j, \beta_j)$. Then (α_j, β_j) is an intersection point with multiplicity e_j.

Algorithm 4.9.7. Let V be a projective plane curve and L be a projective plane line, defined by $\Phi(X,Y,Z)$ and $\Lambda(X,Y,Z)$, respectively. To compute the intersection points and multiplicities for V and L:

- Dehomogenize with respect to each admissible affine component.
- Compute the intersection points and multiplicities for each affine component, as described in algorithm 4.9.6.
- If the dehomogenization was done with respect to Z, homogenize each affine intersection point (α_j, β_j) to obtain the projective point $[A_j, B_j, 1]$, and likewise for the other possible dehomogenizations. Take the multiplicity e_j from the affine intersection point.
- As discussed in remark 4.2.8, a given projective intersection point may be found via more than one affine component. Retain only one copy of each projective intersection point. Use the unique representation described in convention 4.2.4 to facilitate comparison of points.

4.10. Bezout's Theorem

The curves considered in this paper are plane curves, due in large part to the following key theorem which provides insight into their behavior.

Theorem 4.10.1 (Bezout). *If F and G are relatively prime homogeneous polynomials in $\overline{K}[X, Y, Z]$, then F and G intersect in exactly* $\deg(F)\deg(G)$ *points of* $\mathbb{P}^2(\overline{K})$, *when intersections are counted with multiplicity.*

Proof. See [ST], § A.4. □

Remark 4.10.2. Algorithm 4.9.7 calls for dehomogenizing at all variables. This often finds a given projective intersection point in more than one affine component. Using Bezout's theorem, one may reduce the amount of computation by instead trying one affine component at a time, stopping when the full number of intersection points is found.

The following provides a sufficient condition for absolute irreducibility.

Proposition 4.10.3. *A smooth projective plane curve defined by a single equation is absolutely irreducible.*

Proof. The proof is by contrapositive. Let C be a projective plane curve defined by $F(X,Y,Z) \in \overline{K}[X,Y,Z]$. First suppose $F(X,Y,Z) = G(X,Y,Z)H(X,Y,Z)$ for relatively prime G,H; let $d = \deg F$ and $e = \deg G$. Then $d, e \geq 1$ since the factorization is non-trivial. By Bezout's theorem (4.10.1), G and H intersect in at least one point P. By the chain rule,

$$\begin{aligned}
\frac{\partial F}{\partial X}(P) &= \frac{\partial G}{\partial X}(P)H(P) + G(P)\frac{\partial H}{\partial X}(P) = 0\\
\frac{\partial F}{\partial Y}(P) &= \frac{\partial G}{\partial Y}(P)H(P) + G(P)\frac{\partial H}{\partial Y}(P) = 0\\
\frac{\partial F}{\partial Z}(P) &= \frac{\partial G}{\partial Z}(P)H(P) + G(P)\frac{\partial H}{\partial Z}(P) = 0
\end{aligned}$$

showing that C is not smooth at P.

Now suppose F factors non-trivially, but that there is no relatively prime factorization. Thus $F(X,Y,Z) = G^m(X,Y,Z)$ for some irreducible $G(X,Y,Z)$ and $m > 1$. Let P be a point on the curve defined by F. Then $F(P) = 0 = G(P)^m$ implies $G(P) = 0$. Again using the chain rule,

$$\begin{aligned}
\frac{\partial F}{\partial X}(P) &= mG(P)^{m-1}\frac{\partial G}{\partial X}(P)\\
\frac{\partial F}{\partial Y}(P) &= mG(P)^{m-1}\frac{\partial G}{\partial Y}(P)\\
\frac{\partial F}{\partial Z}(P) &= mG(P)^{m-1}\frac{\partial G}{\partial Z}(P)
\end{aligned}$$

If the characteristic of K divides m, then all the partials are zero. Otherwise, the factor $G(P)^{m-1}$, where $m > 1$, makes them all zero at P. Thus in this case, the curve is singular at all points. $\square$

4.11. Zeroes and Poles

Definition 4.11.1. Let p be a point on an affine variety v. Define

$$\begin{aligned}
\mathcal{O}_{p,v} &= \left\{\frac{g}{h} \in K(v) : h(p) \neq 0\right\},\\
\mathcal{M}_{p,v} &= \left\{\frac{g}{h} \in K(v) : h(p) \neq 0, g(p) = 0\right\}.
\end{aligned}$$

Define $\nu_{p,v} : \mathcal{O}_{p,v} \to \mathbb{Z}_{\geq 0} \cup \{\infty\}$ by

$$\nu_{p,v}(\psi) = \max\{d : \psi \in \mathcal{M}^d_{p,v}\}.$$

It is shown in [HvLP] that $\nu_{p,v}$ is a *discrete valuation*, and that $\mathcal{O}_{p,v}$ is a *discrete valuation ring* with $\mathcal{M}_{p,v}$ as its unique maximal ideal. The property of a discrete valuation which is key to the current discussion is that

$$\nu_{p,v}(\psi_1\psi_2) = \nu_{p,v}(\psi_1) + \nu_{p,v}(\psi_2).$$

Computations may be done using the intersection multiplicity defined above.

Proposition 4.11.2. *Let p be a point on an affine plane curve v defined by a polynomial ϕ, and let ℓ be an affine line defined by a polynomial λ. Then $\nu_{p,v}(\lambda) = \mathrm{IM}(v \cap \ell, p)$.*

Proof. Let $e = \nu_{p,v}(\lambda)$. Then $\lambda \in \langle x-a\rangle^e$, $\lambda \notin \langle x-a\rangle^{e+1}$, which implies $e = \dim_{\overline{K}}(\mathcal{O}_{p,v}/\langle\lambda\rangle_{p,v})$, which is the same as $\dim_{\overline{K}}(\mathcal{O}_p/\langle\phi,\lambda\rangle_p)$. $\square$

Finally, extend discrete valuations to projective curves by $\nu_{P,V}(\Psi) = \nu_{p,v}(\psi)$ using some admissible affine component for Ψ. One may now easily compute the intersection multiplicity of a projective plane curve with a product or quotient of lines.

Definition 4.11.3. Let $\Psi \in K(V)$ and $P \in V$; let $d = \nu_{P,V}(\Psi)$. Then P is said to be a *zero of multiplicity* d of Ψ if $d > 0$; P is said to be a *pole of order* $-d$ of Ψ if $d < 0$.

4.12. Divisors

Definition 4.12.1. Let V be a variety defined over a field K. The *divisor group* $\mathrm{Div}_{\overline{K}}(V)$, or simply $\mathrm{Div}(V)$, is the free abelian group on points of $V(\overline{K})$. A *divisor* on V is any element of the divisor group, and is written as $\sum_{P\in V} n_P P$. The integer n_P is called the *coefficient* of P in D. The identity element of the divisor group is written 0. A divisor is said to be K-*rational* if it is invariant under the action of $\mathrm{Gal}(\overline{K}/K)$, where the Galois group acts by $\sigma(D) = \sigma(\sum n_P P) = \sum n_P \sigma(P)$. The set of K-rational divisors of V forms the subgroup $\mathrm{Div}_K(V)$ of $\mathrm{Div}(V)$.

For example, $3P_\infty - 2[5,5,1]$ is a divisor on the sample curve $E(\mathbb{F}_8)$. Recall that from the definition of free abelian group, only finitely many n_P's are non-zero. Also, note that the n_P's are nothing more than integer prefixes; in particular, they are not to be reduced mod p where p is the characteristic of $\mathbb{F}_q$.

Let L be a Galois extension of K. As discussed in section 4.7, points of $V(K)$ that are contained in $\mathbb{P}^n(L)$ but not contained in $\mathbb{P}^n(K)$ cluster into point classes. In this case, a point $\mathcal{P}$ in a divisor D of $V(L)$ is taken to include all the points in the class. For this paper, non-singleton point classes are usually written $\mathcal{P}$; singleton point classes are written P. In this paper, in nearly all cases the point classes discussed are taken to be singletons by extension of the base field. For example, as shown in section 4.7, the points $[2, 0, 1], [4, 0, 1], [6, 0, 1]$ form a single class over $\mathbb{F}_2$. Over $\mathbb{F}_8$, they form three singleton classes.

Definition 4.12.2. Let $\mathcal{P}$ be a point class. The notation $\deg(\mathcal{P})$ represents the cardinality of $\mathcal{P}$.

Definition 4.12.3. Let $D = \sum_{P \in V} n_P P$. The *degree* of D, written $\deg D$, is $\sum_{P \in V} n_P \deg P$, where $\deg P = 1$ if P is a point, or $\deg(P)$ as defined above if P is a point class.

Note that the degree function is a group homomorphism from $\mathrm{Div}(V)$ to $\mathbb{Z}$.

Definition 4.12.4. Let $D = \sum_{P \in V} n_P P$. If $n_P \geq 0$ for all $P \in V$, then D is said to be an *effective* divisor, written $D \succcurlyeq 0$.

For example, since -2 is negative, $3P_\infty - 2[5, 5, 1]$ is a non-effective divisor on $E(\mathbb{F}_8)$. It has degree 1.

Definition 4.12.5. Let $D = \sum_{P \in V} n_P P$. The *support* of D, written $\mathrm{supp}(D)$, is $\{P \in V : n_P \neq 0\}$. Since by definition only finitely many n_P's are non-zero for any divisor D, $\mathrm{supp}(D)$ is always a finite set.

Definition 4.12.6. A *one-point divisor* is one whose support consists of a single point P. Such a divisor is often written in the form rP.

Divisors are a handy way to track points of intersection of curves, counting multiplicity. This motivates the following definition.

Definition 4.12.7. Let $F \in K(V)$. The *intersection divisor* of F with respect to the variety V is $\sum_{P \in V} \nu_{P,V}(F)P$.

Remark 4.12.8. If $G(P) = 0$ and $H(P) = 0$, and if P is a smooth point of V, and if V is one-dimensional, then [Sil] there is another rational function $F' \sim F$ in $K(V)$ such that either F' is defined at P, or P is a pole of F'. That is, $0/0$ situations are removable at smooth points.

Proposition 4.12.9 (Zeroes and poles). *Suppose that V is defined by a single polynomial M and that G/H is non-zero. Then* $\deg \operatorname{div}(G/H) = 0$.

Remark 4.12.10. Informally, G/H has as many zeroes as poles. The condition that V be defined by a single polynomial is in fact the case for curves studied in this paper. Also note that Bezout's theorem applies to polynomials, while this proposition applies to rational functions in a function field.

Proof. Since V is a variety, M is prime. Also, since $K[X_0, \ldots, X_n]$ is a unique factorization domain, G and H may be assumed to be relatively prime. Since H in the denominator is non-zero mod M, it is not a multiple of M and so is relatively prime to M. Also, G is not a multiple of M, since if it were, G/H would be zero mod M. Thus, M is relatively prime to both G and H. Since G and H are assumed to be of the same degree, by Bezout's theorem M and G intersect in as many points as M and H do. Therefore, $\sum \deg P = \sum \deg Q$, i.e. $\deg \operatorname{div}(G/H) = 0$. ☐

4.13. Associated Vector Spaces

Definition 4.13.1. For each K-rational divisor D of $V(K)$, one associates a vector space $\mathcal{L}(D)$ over K:

$$\mathcal{L}(D) = \{F \in K(V) : \operatorname{div}(F) + D \succcurlyeq 0\} \cup \{0\}.$$

One may form an analogous vector space over $\overline{K}$ rather than K, as discussed in section 2.7 of [HvLP]. The key property of $\mathcal{L}(D)$ is that all rational functions in $\mathcal{L}(D)$ have poles confined to the points with positive coefficients in D. This property is used starting in chapter 5 to construct sets of rational functions and sets of points where division by zero does not occur.

The following proposition is for use in section 5.2 of chapter 5.

Proposition 4.13.2. *If* $\deg(D) < 0$*, then* $\mathcal{L}(D) = \{0\}$.

Proof. Let F be non-zero in $K(V)$. From the zeroes-and-poles proposition (4.12.9), $\deg \operatorname{div}(F) = 0$. Thus

$$\deg \operatorname{div}(F) + \deg(D) = \deg(\operatorname{div}(F) + D) < 0 \implies \operatorname{div}(F) + D \not\geq 0 \implies F \notin \mathcal{L}(D).$$

□

It can be shown ([Sil], [HvLP]) that $\mathcal{L}(D)$ is in fact finite-dimensional over K. The dimension of $\mathcal{L}(D)$ is written $\ell(D)$. This is the code dimension for algebraic-geometry codes as defined below. The Riemann-Roch theorem [Sil] in some cases permits quick computation of $\ell(D)$. It also plays a central role in obtaining information about the parameters of algebraic-geometry codes, as discussed in chapter 5.

Theorem 4.13.3 (Riemann-Roch). *Let V be a smooth curve of genus g, defined over K, and let D be a divisor on V. Then*

$$\ell(D) \geq \deg D - g + 1.$$

Furthermore, if $\deg D > 2g - 2$*, then*

$$\ell(D) = \deg D - g + 1.$$

Proof. See [Pre], chapter 11. □

The case $\deg D > 2g-2$ is referred to as the *exact case* of the Riemann-Roch theorem, since it exactly specifies $\ell(D)$.

4.14. Bases for One-Point Divisors

The goal of this section is to provide a technique to compute a basis for divisors of the form $\mathcal{L}(rP)$, $r \geq 0$.

Definition 4.14.1. For a point P on V, a *gap* of P is a non-negative integer i such that $\ell(iP) = \ell(i-1)P)$. A *non-gap* of P is a non-negative integer such that $\ell(iP) \neq \ell((i-1)P)$.

See [HvLP] for justification of the following assertions: A non-negative integer i is a non-gap of P iff there is an $F \in K(V)$ with a pole of order i in P, and poles at no other point of V. The number of gaps is g. The sum of non-gaps m_1, m_2 is another non-gap $m_1 + m_2$. The non-gaps form the *Weierstrass semigroup.*

Proposition 4.14.2. *Let $(\gamma_i : i \in \mathbb{Z}_+)$ be an enumeration of the non-gaps of P, with $0 = \gamma_1 < \gamma_2 < \ldots$. Let $F_i \in \mathcal{L}(\gamma_i P)$ be such that $\nu_P(F) = -\gamma_i$. Then $\{F_1 \ldots, F_r\}$ is a basis for $\mathcal{L}(\gamma_r P)$.*

Proof. [Pre], proposition 4.6; [HvLP], section 2.6. ☐

Proposition 4.14.2 permits computation of a basis for a one-point divisor as follows.

- Let $D = rP$.
- All gaps are at $0 \leq r \leq 2g-2$. Of those $2g-1$ values of r, g are gaps and $g-1$ are non-gaps.
- Obtain functions with various pole orders.
- If $g-1$ different functions are found with pole orders at different r, $0 \leq r \leq 2g-2$, then the gaps are at the remaining r values.
- For $r > 2g-2$, a function may be found with pole order r at P.

4.15. Example Bases

Consider the curve E as shown above, and let $D = rP_\infty$ for some positive integer r. Then $\deg D = r$.

The intersection divisor $\operatorname{div}(X)$ consists of the points of E, i.e. zeroes of $Y^2Z + YZ^2 + X^3 + XZ^2 + Z^3$, where additionally $X = 0$. Thus $Y^2Z + YZ^2 + Z^3 = Z(Y^2 + YZ + Z^2)$. This splits into two cases, $Z = 0$ or $Z \neq 0$. If $Z = 0$, then $X = 0$ and the only infinite point is $P_\infty = [0, 1, 0]$. If $Z \neq 0$, then let $Z = 1$. This is possible since each projective point has one arbitrary coordinate via

λ-scaling. Then $Y^2 + Y + 1 = 0$. This quadratic is irreducible over $\mathbb{F}_2$, so it has two single roots in $\mathbb{F}_4$. Using the notation of section 2.2, $Y = 2, 3$, and so the remaining points are $[0, 2, 1]$ and $[0, 3, 1]$. Using the notation of section 4.7,

$$\mathrm{div}(X) = P_\infty + \mathcal{Q}_2.$$

Note that X is not required to have as many poles as zeroes, since it is not a rational function.

The intersection divisor $\mathrm{div}(Y)$ consists of the points of E where $Y = 0$. Thus $X^3+XZ^2+Z^3$. If $Z = 0$, then $X = 0$. However, $[0, 0, 0]$ is not a projective point. If $Z \neq 0$, then again let $Z = 1$. Points are roots of $X^3 + X + 1$, which is a cubic irreducible over $\mathbb{F}_2$. Using the notation of section 2.2, the three distinct roots are $X = 2, 4, 6$. Then

$$\mathrm{div}(Y) = \mathcal{Q}_3$$

The intersection divisor $\mathrm{div}(Z)$ consists of the points of E where $Z = 0$. Thus $X^3 = 0$. This forces $Y = 1$, but here with multiplicity three. Then

$$\mathrm{div}(Z) = 3P_\infty.$$

Now that these three intersection divisors have been computed, it is possible to write down a basis for $\mathcal{L}(rP_\infty)$. Note that

$$\begin{aligned}
\mathrm{div}(X^iY^j/Z^{i+j}) &= i(P_\infty + \mathcal{Q}_2) + j(\mathcal{Q}_3) - (i + j)(3P_\infty) \\
&= iP_\infty + i\mathcal{Q}_2 + j\mathcal{Q}_3 - 3iP_\infty - 3jP_\infty \\
&= (-2i - 3j)P_\infty + i\mathcal{Q}_2 + j\mathcal{Q}_3.
\end{aligned}$$

For $r = 0$, trivially the rational function 1 forms a linearly independent set. Since E has genus 1, $2g - 2 = 0$ and thus the exact case of the Riemann-Roch theorem (4.13.3) applies whenever $\deg(rP_\infty) = r \geq 1$. In particular, for $r = 1$, $\ell(P_\infty) = 1$. However, $\mathcal{L}(P_\infty)$ already contains $F = 1$. Therefore, $\mathcal{L}(1P_\infty) = \mathcal{L}(0P_\infty)$, and 0 is a gap of E. Since $g(E) = 1$, this is the only gap.

For $r = 2$, take $i = 1$ and $j = 0$. Then $\mathrm{div}(X/Z) + 2P_\infty = \mathcal{Q}_2$, which is in $\mathcal{L}(2P_\infty)$ but not $\mathcal{L}(P_\infty)$, hence linearly independent from 1. For higher r, equate r with $-2i - 3j$ as shown in table 11. As in the case $r = 2$, one obtains for each r an element of $\mathcal{L}(rP_\infty)$ which was not in

$\mathcal{L}((r-1)P_\infty)$, hence linearly independent from the previous set. To summarize, bases for the first few $\mathcal{L}(rP_\infty)$ are shown in table 12.

i	j	$F = X^iY^j/Z^{i+j}$	$\mathrm{div}(F)$	r	$\mathrm{div}(F) + rP_\infty$
0	0	1	$0P_\infty + 0\mathcal{Q}_2 + 0\mathcal{Q}_3$	$0, 1$	0
1	0	X/Z	$-2P_\infty + 1\mathcal{Q}_2 + 0\mathcal{Q}_3$	2	$\mathcal{Q}_2$
0	1	Y/Z	$-3P_\infty + 0\mathcal{Q}_2 + 1\mathcal{Q}_3$	3	$\mathcal{Q}_3$
2	0	X^2/Z^2	$-4P_\infty + 2\mathcal{Q}_2 + 0\mathcal{Q}_3$	4	$2\mathcal{Q}_2$
1	1	XY/Z^2	$-5P_\infty + 1\mathcal{Q}_2 + 1\mathcal{Q}_3$	5	$\mathcal{Q}_2 + \mathcal{Q}_3$
0	2	Y^2/Z^2	$-6P_\infty + 0\mathcal{Q}_2 + 2\mathcal{Q}_3$	6	$2\mathcal{Q}_3$
2	1	X^2Y/Z^3	$-7P_\infty + 2\mathcal{Q}_2 + 1\mathcal{Q}_3$	7	$2\mathcal{Q}_2 + \mathcal{Q}_3$
1	2	XY^2/Z^3	$-8P_\infty + 1\mathcal{Q}_2 + 2\mathcal{Q}_3$	8	$\mathcal{Q}_2 + 2\mathcal{Q}_3$
0	3	Y^3/Z^3	$-9P_\infty + 0\mathcal{Q}_2 + 3\mathcal{Q}_3$	9	$3\mathcal{Q}_3$
2	2	X^2Y^2/Z^4	$-10P_\infty + 2\mathcal{Q}_2 + 2\mathcal{Q}_3$	10	$2\mathcal{Q}_2 + 2\mathcal{Q}_3$
1	3	XY^3/Z^4	$-11P_\infty + 1\mathcal{Q}_2 + 3\mathcal{Q}_3$	11	$\mathcal{Q}_2 + 3\mathcal{Q}_3$
0	4	Y^4/Z^4	$-12P_\infty + 0\mathcal{Q}_2 + 4\mathcal{Q}_3$	12	$4\mathcal{Q}_3$
2	3	X^2Y^3/Z^5	$-13P_\infty + 2\mathcal{Q}_2 + 3\mathcal{Q}_3$	13	$2\mathcal{Q}_2 + 3\mathcal{Q}_3$
1	4	XY^4/Z^5	$-14P_\infty + 1\mathcal{Q}_2 + 4\mathcal{Q}_3$	14	$\mathcal{Q}_2 + 4\mathcal{Q}_3$
0	5	Y^5/Z^5	$-15P_\infty + 0\mathcal{Q}_2 + 5\mathcal{Q}_3$	15	$5\mathcal{Q}_3$
2	4	X^2Y^4/Z^6	$-16P_\infty + 2\mathcal{Q}_2 + 4\mathcal{Q}_3$	16	$2\mathcal{Q}_2 + 4\mathcal{Q}_3$
1	5	XY^5/Z^6	$-17P_\infty + 1\mathcal{Q}_2 + 5\mathcal{Q}_3$	17	$\mathcal{Q}_2 + 5\mathcal{Q}_3$
0	6	Y^6/Z^6	$-18P_\infty + 0\mathcal{Q}_2 + 6\mathcal{Q}_3$	18	$6\mathcal{Q}_3$

Table 11. Selected function divisors for E

r	Basis for $\mathcal{L}(rP_\infty)$
1	$\{1\}$
2	$\{1,\ X/Z\}$
3	$\{1,\ X/Z,\ Y/Z\}$
4	$\{1,\ X/Z,\ Y/Z,\ X^2/Z^2\}$
5	$\{1,\ X/Z,\ Y/Z,\ X^2/Z^2,\ XY/Z^2\}$
6	$\{1,\ X/Z,\ Y/Z,\ X^2/Z^2,\ XY/Z^2,\ Y^2/Z^2\}$
7	$\{1,\ X/Z,\ Y/Z,\ X^2/Z^2,\ XY/Z^2,\ Y^2/Z^2\ X^2Y/Z^3\}$
8	$\{1,\ X/Z,\ Y/Z,\ X^2/Z^2,\ XY/Z^2,\ Y^2/Z^2,\ X^2Y/Z^3,\ XY^2/Z^3\}$

Table 12. Bases for $\mathcal{L}(rP_\infty)$, $r = 1, \ldots, 8$

CHAPTER 5

Construction and Encoding of Goppa Codes

Let V be a smooth curve defined over $\mathbb{F}_q$. Let $\boldsymbol{P} = (P_1, \ldots, P_n) \in \mathbb{F}_q^n$ where the P_j's are n distinct $\mathbb{F}_q$-rational points of V. Let D be a divisor on V, with $0 < \deg D < n$. Further, assume that none of the P_j's appear in the support of D. This is so that no P_j is a pole of any $F \in \mathcal{L}(D)$: $F(P_j) \in \mathbb{F}_q$ for all P_j's and for all $F \in \mathcal{L}(D)$.

Typically, D is a one-point divisor on a particular point selected from the curve, while $\boldsymbol{P}$ includes some or all of the remaining points on the curve. Making $\boldsymbol{P}$ larger or smaller permits variation of the code length n.

5.1. The Goppa Primary Code

Definition 5.1.1. Define

$$F(\boldsymbol{P}) = (F(P_1), \ldots, F(P_n))$$

to be the componentwise application of F.

Definition 5.1.2. Given $\boldsymbol{v} \in \mathbb{F}_q^n$, the *syndrome* of $\boldsymbol{v}$ with respect to F and $\boldsymbol{P}$ is the dot product

$$F(\boldsymbol{P}) \cdot \boldsymbol{v} = \sum_{j=1}^{n} F(P_j) v_j.$$

Definition 5.1.3. The *Goppa primary code* (or Goppa residue code) for V, $\boldsymbol{P}$ and D is

$$C_p(V, \boldsymbol{P}, D) = \{\boldsymbol{v} \in \mathbb{F}_q^n : F(\boldsymbol{P}) \cdot \boldsymbol{v} = 0 \text{ for all } F \in \mathcal{L}(D)\}.$$

The parameters for these codes are as follows. By construction, the length of C_p is n, and q is a given from $\mathbb{F}_q$.

Proposition 5.1.4. *The dimension k of C_p is*

$$k = n - \deg(D) + g - 1$$

Proof. Proposition 5.2.2 in the following section shows that the dual code to C_p has dimension $\deg(D) - g + 1$. From the GH-perp proposition (3.8.6), the dimension of C_p is n minus the dimension of its dual, which is the desired result. □

Proposition 5.1.5. *Let d be the minimum distance of the Goppa primary code $C_p(V, \boldsymbol{P}, D)$; let g be the genus of V. If $\deg(D) > 2g - 2$, then $d \geq \deg(D) - 2g + 2$.*

Proof. Since C_p is a linear code, from the observation in definition 3.2.3 it suffices to show the result for the minimum weight over all non-zero code words of C. Following [Pre], let $\boldsymbol{u}$ be a non-zero code word of C_p, of weight $w > 0$. Without loss of generality, suppose the P_j's of $\boldsymbol{P}$ and the u_j's are numbered such that $u_j \neq 0$ for $j = 1, \ldots, w$ and $u_j = 0$ for $j = w+1, \ldots, n$. Seeking a contradiction, suppose $1 \leq w < \deg(D) - 2g + 2$. Let

$$D_w = D - P_1 - \ldots - P_w, \quad D_{w-1} = D - P_1 - \ldots - P_{w-1}.$$

Then

$$w < \deg(D) - 2g + 2 \quad \Longrightarrow \quad \deg(D) - w = \deg(D_w) > 2g - 2$$

and thus $\deg(D_{w-1}) > 2g - 2$ as well. By the Riemann-Roch theorem,

$$\ell(D_w) = \deg(D) - w - g + 1, \quad \ell(D_{w-1}) = \deg(D) - w - g + 2.$$

Thus there exists an F in $\mathcal{L}(D_{w-1})$ which is not in $\mathcal{L}(D_w)$. This implies $F(P_j) = 0$ for $j = 1, \ldots, w - 1$, and $F(P_w) \neq 0$. Since $D_{w-1} \preccurlyeq D$, $F \in \mathcal{L}(D)$ and $F(\boldsymbol{P}) \cdot \boldsymbol{u} = F(P_w)u_w \neq 0$, contradicting the assumption that $\boldsymbol{u} \in C$. □

Definition 5.1.6. The lower bound $\deg(D) - 2g + 2$ is called the *designed minimum distance* of a Goppa code. A code's actual minimum distance is called the *true minimum distance.*

In general, to compute the true minimum distance of a code requires a brute-force iteration over all the code words. For large codes, this is impractical. By constrast, the designed minimum distance for Goppa codes is easily computed using the above formula. Goppa codes are not the only family of codes for which this is the case [MS].

For linear codes in general, one has

$$0 < k < n$$

as discussed in chapter 3. As a consequence of proposition 5.1.5, however, there do not exist Goppa codes over curves of higher genus with dimension too close to 0 or n.

Corollary 5.1.7. *The dimension k of a Goppa code is related to the genus by*

$$g - 1 < k < n - g + 1.$$

Proof. For the lower bound,

$$\begin{aligned} k &= n - \deg D + g - 1 \\ \deg D &= n - k + g - 1 \\ n - k + g - 1 &< n \quad \text{since } \deg D < n \\ k - g + 1 &> 0 \\ k &> g - 1. \end{aligned}$$

For the upper bound, we need $\deg D > 2g - 2$ in order to guarantee a minimum distance d. Thus

$$\begin{aligned} k &= n - \deg D + g - 1 \\ \deg D &= n - k + g - 1 > 2g - 2 \\ k &< n - g + 1. \end{aligned}$$

□

Proposition 5.1.5 provides a lower bound on d; the Singleton bound (3.5.1) provides an easily computed upper bound on d. Tighter upper bounds are available [MS], but are not necessary for the short codes defined in this paper.

The question of why algebraic-geometry codes are of interest to the coding-theory community is not addressed here. It is possible to use algebraic geometry to produce sequences of codes which are good in the sense of section 3.6. See, for example, chapter 5 of [Wal] and section 1 of [HvLP] for a discussion of connections between AG codes and the Gilbert-Varshamov bound. See also [Gop] for the original presentation of Goppa codes.

5.2. The Goppa Dual Code

Definition 5.2.1. Let V, $\boldsymbol{P}$ and D be as in section 5.1 and let $F \in \mathcal{L}(D)$. The *Goppa dual code* (or Goppa function code) for V, $\boldsymbol{P}$ and D is

$$C_d(V, \boldsymbol{P}, D) = \{F(\boldsymbol{P}) : F \in \mathcal{L}(D)\} \subset \mathbb{F}_q^n.$$

That is, C_d is the image of the evaluation map

$$\varepsilon : \mathcal{L}(D) \to \mathbb{F}_q^n \text{ given by } F \mapsto F(\boldsymbol{P})$$

and therefore C_d is a linear code over $\mathbb{F}_q$. Thus, the Goppa primary code $C_p(V, \boldsymbol{P}, D)$ is now seen to be

$$\begin{aligned} C_p(V, \boldsymbol{P}, D) &= \{\boldsymbol{v} \in \mathbb{F}_q^n : F(\boldsymbol{P}) \cdot \boldsymbol{v} = 0 \text{ for all } F \in \mathcal{L}(D)\} \\ &= \{\boldsymbol{v} \in \mathbb{F}_q^n : F(\boldsymbol{P}) \cdot \boldsymbol{v} = 0 \text{ for all } F(\boldsymbol{P}) \in C_d\} \\ &= \{\boldsymbol{v} \in \mathbb{F}_q^n : \boldsymbol{u} \cdot \boldsymbol{v} = 0 \text{ for all } \boldsymbol{u} \in C_d\} \end{aligned}$$

which is precisely the dual code of C_d. The primaries get their name because they are easier to decode, and decoding is where most of the work is. The dual codes are used because it is easy to compute the generator matrix for a dual code. Then, one may use the GH-perp proposition (3.8.6) to obtain a generator matrix for the corresponding primary code.

The code parameters for Goppa dual codes are as follows. By construction, the length of C_d is n, and q is a given from $\mathbb{F}_q$. It remains to find the dimension $n - k$ and the minimum distance d. Once the dimension $n - k$ of the dual code is obtained, the GH-perp proposition (3.8.6) shows the dimension of the primary code to be k.

Proposition 5.2.2. *Let V, $\boldsymbol{P}$, and D be defined as above, and let g be the genus of V. If* $\deg(D) > 2g-2$*, then $C_d(V, \boldsymbol{P}, D)$ has dimension* $\deg D - g + 1$.

Proof. Since $\deg(D) > 2g - 2$, the Riemann-Roch theorem (theorem 4.13.3) gives $\ell(D) = \deg D - g + 1$. Since C_d is the image of $\mathcal{L}(D)$ under ε, it suffices to show that ε is one-to-one, for which it suffices in turn to show that ε has trivial kernel. Following [Wal], suppose $\varepsilon(F) = 0$ for some $F \in \mathcal{L}(D)$. This means $F(P_1) = \ldots = F(P_n) = 0$. Since each P_j is a zero of F, the coefficient n_{P_j} of each P_j in the divisor $\text{div}(F)$ is positive. Since the P_j's were chosen to lie outside $\text{supp}(D)$, $\text{div}(F) + D - P_1 - \ldots - P_n \not\geq 0$. Since it is assumed that $\deg(D) < n$, $\deg(D - P_1 - \ldots - P_n) < 0$. By proposition 4.13.2, $\mathcal{L}(D - P_1 - \ldots - P_n) = \{0\}$. This forces $F = 0$ which proves the claim. □

Proposition 5.2.3. *Let V, $\boldsymbol{P}$, and D be defined as above. If* $2g-2 < \deg D < n$*, then $C_d(V, \boldsymbol{P}, D)$ has minimum distance* $d \geq n - \deg D$.

Proof. Following [Wal], let $\varepsilon(F) = F(\boldsymbol{P}) \in C_d$ be a code word of minimum non-zero weight d. Then exactly d coordinates of $\varepsilon(F)$ are non-zero. Without loss of generality, assume $F(P_{d+1}) = \ldots = F(P_n) = 0$. This implies that $\text{div}(F) + D - P_{d+1} - \ldots - P_n \geq 0$. By the contrapositive to proposition 4.13.2, the divisor $D - P_{d+1} - \ldots - P_n$ has non-negative degree, i.e. $\deg(D) - (n - d) \geq 0$, which is to say $d \geq n - \deg D$. □

5.3. Encoding

Let $k = n - \ell(D)$, that is, $n - k = \ell(D)$. Using notation as above, let $\{F_1, \ldots, F_{n-k}\}$ be a basis for $\mathcal{L}(D)$. From linear algebra it is well known that $\{\varepsilon(F_1), \ldots, \varepsilon(F_{n-k})\}$ is a basis for C_d. Thus a generator matrix for C_d is given by the matrix having entries

$$F_i(P_j)$$

for $i = 1, \ldots, n-k$ and $j = 1, \ldots, n$. Here the matrix is used to map from $\mathbb{F}_q^{n-k}$ to $\mathbb{F}_q^n$, not from $\mathcal{L}(D)$. The $(n-k)$-dimensional $\mathbb{F}_q$-vector space $\mathcal{L}(D)$ is identified with the $(n-k)$-dimensional $\mathbb{F}_q$-vector space $\mathbb{F}_q^{n-k}$. Given a generator matrix for C_d, one computes a basis for its kernel to obtain a parity-check matrix for C_d. This is a generator matrix G for C_p by the GH-perp proposition (3.8.6), and thus one encodes a message word $\boldsymbol{m} \in \mathbb{F}_q^k$ by forming the product $\boldsymbol{m}G$.

Example 5.3.1. The sample curve E is defined over $K = \mathbb{F}_2$, but one may work over $L = \mathbb{F}_8$. Let $V = E(\mathbb{F}_8)$ and $D = 8P_\infty$. Let $\boldsymbol{P}$ be taken from $E(\mathbb{F}_8) \setminus \{P_\infty\}$ as shown in example 4.5.1, namely,

$$\begin{aligned} \boldsymbol{P} &= ([2,0,1],[4,0,1],[6,0,1],[2,1,1],[4,1,1],[6,1,1], \\ &\quad [3,2,1],[5,4,1],[7,6,1],[3,3,1],[5,5,1],[7,7,1]). \end{aligned}$$

Then for $k = 7$, $n = 12$, a generator matrix for C_d, hence a parity-check matrix H for C_p, is

$$H = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 2 & 4 & 6 & 2 & 4 & 6 & 3 & 5 & 7 & 3 & 5 & 7 \\ 0 & 0 & 0 & 1 & 1 & 1 & 2 & 4 & 6 & 3 & 5 & 7 \\ 4 & 6 & 2 & 4 & 6 & 2 & 5 & 7 & 3 & 5 & 7 & 3 \\ 0 & 0 & 0 & 2 & 4 & 6 & 6 & 2 & 4 & 5 & 7 & 3 \end{bmatrix}.$$

If k were smaller, one would use the first k rows of this matrix. For the case $k = 1$, one obtains the n-bit repetition code (example 3.1.2) as a special case.

Computing a kernel basis for H gives a parity-check matrix for C_d, hence a generator matrix G for C_p:

$$G = \begin{bmatrix} 6 & 7 & 1 & 6 & 7 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 3 & 4 & 4 & 4 & 6 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 4 & 7 & 3 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 5 & 7 & 5 & 3 & 5 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 7 & 4 & 6 & 5 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 2 & 2 & 4 & 4 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 3 & 5 & 6 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

CHAPTER 6

SV Decoding

Using the notation from section 3.9 and chapter 5, let $C_p(V, \boldsymbol{P}, D)$ be a Goppa primary code with dimension k, length n, and minimum distance d over $\mathbb{F}_q$. Suppose that a message word $\boldsymbol{m} \in \mathbb{F}_q^k$ has been encoded as a code word $\boldsymbol{u} \in \mathbb{F}_q^n$, then corrupted with error word $\boldsymbol{e}$ to form a received word $\boldsymbol{v} = \boldsymbol{u} + \boldsymbol{e}$. The task of a decoding algorithm is to estimate the error as $\hat{\boldsymbol{e}}$. Once this is done, an estimation of the code word $\hat{\boldsymbol{u}}$ is simple: since $\boldsymbol{v} = \boldsymbol{u} + \boldsymbol{e} = \hat{\boldsymbol{u}} + \hat{e}$, $\hat{\boldsymbol{u}} = \boldsymbol{v} - \hat{\boldsymbol{e}}$. The estimated message word $\hat{\boldsymbol{m}}$ may then be obtained by solving the linear system of equations $\hat{\boldsymbol{u}} = \hat{\boldsymbol{m}}G$ for $\hat{\boldsymbol{m}}$.

The Skorobogatov-Vlăduţ (SV) decoding algorithm originates in [SV], but is presented here as in [Pre]. The algorithm proceeds in four steps: (1) Calculate a syndrome. (2) Obtain a rational function λ in $K(V)$ which may be used to find the locations of the non-zero elements of the vector $\hat{\boldsymbol{e}}$. (3) Use λ to find the error locations. (4) Find the values $\hat{e}_j$ at the error locations of $\hat{\boldsymbol{e}}$. Throughout this section, it is assumed that the error word $\boldsymbol{e}$ is decodable, so $\boldsymbol{e}$ is written in place of $\hat{\boldsymbol{e}}$.

6.1. Error Locators

Definition 6.1.1. Write $\boldsymbol{e} = (e_1, \ldots, e_n)$ and $\boldsymbol{P} = (P_1, \ldots, P_n)$. The point P_j is called an *error location* of $\boldsymbol{e}$ if $e_j \neq 0$.

Notation 6.1.2. Given n-tuples $\boldsymbol{u} = (u_1, \ldots, u_n)$ and $\boldsymbol{v} = (v_1, \ldots, v_n)$, let $\boldsymbol{u} * \boldsymbol{v}$ denote the elementwise vector product $(u_1v_1, \ldots, u_nv_n)$.

Definition 6.1.3. A non-zero $\lambda \in K(V)$ is called an *error locator* for $\boldsymbol{e}$ when (1) $\lambda(P_j) = 0$ for all error locations of $\boldsymbol{e}$, and (2) λ has no poles among $P_1, \ldots, P_n$.

Notation 6.1.4. Given $\boldsymbol{P}$ and $\boldsymbol{e}$ as above, let $\boldsymbol{Z}$ be the vector consisting of the elements P_j of $\boldsymbol{P}$ such that e_j is non-zero. Likewise, given an error locator λ for $\boldsymbol{e}$, let $\boldsymbol{z}$ be the vector consisting of elements e_j of $\boldsymbol{e}$ such that $\lambda(P_j) = 0$.

Given this terminology, equivalent conditions for an error locator λ are that $\lambda(\boldsymbol{P}) * \boldsymbol{e} = \boldsymbol{0}$, and that λ is pole-free on $\boldsymbol{P}$ and $\lambda(\boldsymbol{Z}) = \boldsymbol{0}$.

First, existence of error locators is established.

Proposition 6.1.5. *Let $\boldsymbol{e}$ be an error word of weight at most t. Let A be a divisor of V such that (1) $\operatorname{supp}(A)$ contains none of the P_j's in $\boldsymbol{P}$ and (2) $\ell(A) > t$. Then $\mathcal{L}(A)$ contains an error locator for $\boldsymbol{e}$.*

Remark. Since the divisor A is needed for SV decoding but is not needed to construct the code itself, it is called an *auxiliary divisor.*

Proof. Since A is a divisor on V, $\mathcal{L}(A)$ is finite dimensional. Choose a basis $\lambda_1, \ldots, \lambda_{\ell(A)}$ of $\mathcal{L}(A)$. As discussed in the proof of proposition 5.2.3, the evaluation homomorphism $\lambda_i \mapsto \lambda_i(\boldsymbol{P})$ is 1-1. Thus $\{\lambda_1(\boldsymbol{P}), \ldots, \lambda_{\ell(A)}(\boldsymbol{P})\}$ is a linearly independent set. Then

$$\sum_{i=1}^{\ell(A)} c_i \lambda_i(\boldsymbol{Z}) = \boldsymbol{0}$$

is a homogeneous system of at most t independent equations in $\ell(A)$ unknowns. Since $t < \ell(A)$ by hypothesis, a non-zero solution $\boldsymbol{c} = (c_1, \ldots, c_{\ell(A)})$ exists. Choose one such solution and let $\lambda = \sum_{i=1}^{\ell(A)} c_i \lambda_i$. Since $\lambda(\boldsymbol{Z}) = \boldsymbol{0}$, λ is an error locator for $\boldsymbol{e}$. □

Now that existence of an error locator is guaranteed, the following proposition permits the explicit computation of an error locator. It requires another auxiliary divisor.

Lemma 6.1.6. *Let $\boldsymbol{e}$ have weight at most t. Let R be a divisor on V such that (1) the support of R contains none of the P_j's in $\boldsymbol{P}$, and (2) $\deg(R) \geq t + 2g - 1$. Then a rational function $\lambda \in K(V)$ without poles in $\boldsymbol{P}$ is an error locator for $\boldsymbol{e}$ if and only if $(\rho\lambda)(\boldsymbol{P}) \cdot \boldsymbol{e} = 0$ for all $\rho \in \mathcal{L}(R)$.*

Proof, following [Pre]. First suppose λ is an error locator for $\boldsymbol{e}$ without poles in $\boldsymbol{P}$, and let $\rho \in \mathcal{L}(R)$. Since the support of R contains none of the P_j's in $\boldsymbol{P}$, ρ has no poles on the P_j's in $\boldsymbol{P}$. Then

$$\begin{aligned} \mathbf{0} &= \lambda(\boldsymbol{P}) * \boldsymbol{e} \\ \implies (0,\ldots,0) &= (\lambda(P_1)e_1,\ldots,\lambda(P_n)e_n) \\ &= (\rho(P_1)\lambda(P_1)e_1,\ldots,\rho(P_n)\lambda(P_n)e_n) \\ \implies 0 &= \rho(P_1)\lambda(P_1)e_1 + \ldots + \rho(P_n)\lambda(P_n)e_n \\ &= (\rho\lambda)(P_1)e_1 + \ldots + (\rho\lambda)(P_n)e_n \\ &= (\rho\lambda)(\boldsymbol{P}) \cdot \boldsymbol{e}. \end{aligned}$$

Now suppose $(\rho\lambda)(\boldsymbol{P}) \cdot \boldsymbol{e} = 0$ for all $\rho \in \mathcal{L}(R)$. Then

$$\begin{aligned} 0 &= (\rho\lambda)(\boldsymbol{P}) \cdot \boldsymbol{e} \\ &= ((\rho\lambda)(P_1),\ldots,(\rho\lambda)(P_n)) \cdot (e_1,\ldots,e_n) \\ &= (\rho\lambda)(P_1)e_1 + \ldots + (\rho\lambda)(P_n)e_n \\ &= \rho(P_1)\lambda(P_1)e_1 + \ldots + \rho(P_n)\lambda(P_n)e_n \\ &= (\rho(P_1),\ldots,\rho(P_n)) \cdot (\lambda(P_1)e_1,\ldots,\lambda(P_n)e_n \\ &= \rho(\boldsymbol{P}) \cdot (\lambda(\boldsymbol{P}) * \boldsymbol{e}) \end{aligned}$$

Thus the vector $\lambda(\boldsymbol{P}) * \boldsymbol{e}$ is a code word of $C_p(V,\boldsymbol{P},R)$. Since by assumption $\deg(R) \geq t+2g-1$, by proposition 5.1.5 the minimum distance of $C_p(V,\boldsymbol{P},R)$ is greater than or equal to $t+1$. Since the Goppa codes are linear, the minimum distance of $C_p(V,\boldsymbol{P},R)$ is the same as its minimum weight. Since $\boldsymbol{e} = (e_1,\ldots,e_n)$ and λ is pole-free on $\boldsymbol{P}$, $\lambda(\boldsymbol{P}) * \boldsymbol{e}$ has weight at most t. Thus $\lambda(\boldsymbol{P}) * \boldsymbol{e} = \mathbf{0}$ and therefore λ is an error locator for $\boldsymbol{e}$. $\square$

Proposition 6.1.7. *Let R be as above, and suppose A is a divisor on V such that $\mathcal{L}(A)$ contains an error locator for $\boldsymbol{e}$. Let $\{\rho_1,\ldots,\rho_{\ell(R)}\}$ and $\{\lambda_1,\ldots,\lambda_{\ell(A)}\}$ be bases for $\mathcal{L}(R)$ and $\mathcal{L}(A)$, respectively. Let S be the $\ell(R) \times \ell(A)$ matrix of syndromes given by*

$$S_{ij} = (\rho_i\lambda_j)(\boldsymbol{P}) \cdot \boldsymbol{e}.$$

Then $\lambda = \sum_{j=1}^{\ell(A)} c_j\lambda_j$ is an error locator for $\boldsymbol{e}$ in $\mathcal{L}(A)$ if and only if $\boldsymbol{c} = (c_1, \ldots, c_{\ell(A)})$ is a solution of the homogeneous linear system

$$S\boldsymbol{c} = \mathbf{0}.$$

Proof. First, observe that $S\boldsymbol{c} = \mathbf{0}$ means that for $i = 1, \ldots, \ell(R)$,

$$\begin{aligned} 0 &= \sum_{j=1}^{\ell(A)} c_j(\rho_i\lambda_j)(\boldsymbol{P}) \cdot \boldsymbol{e} = \sum_{j=1}^{\ell(A)} c_j \sum_{k=1}^{n} \rho_i(P_k)\lambda_j(P_k)e_k \\ &= \sum_{k=1}^{n} \rho_i(P_k) \sum_{j=1}^{\ell(A)} c_j\lambda_j(P_k)e_k = \sum_{k=1}^{n} \rho_i(P_k)\lambda(P_k)e_k = (\rho_i\lambda)(\boldsymbol{P}) \cdot \boldsymbol{e} \end{aligned}$$

Thus it suffices to show that λ is an error locator for $\boldsymbol{e}$ iff $(\rho_i\lambda)(\boldsymbol{P}) \cdot \boldsymbol{e} = 0$ for all ρ_i in the basis for $\mathcal{L}(R)$. First suppose the former. By the lemma, $(\rho\lambda)(\boldsymbol{P}) \cdot \boldsymbol{e} = 0$ for all $\rho \in \mathcal{L}(R)$, which applies in particular to each ρ_i. Now suppose the latter. Let $\rho = \sum_{i=1}^{\ell(R)} b_i\rho_i \in \mathcal{L}(R)$. Then

$$\begin{aligned} (\rho_i\lambda)(\boldsymbol{P}) \cdot \boldsymbol{e} &= 0, \quad i = 1, \ldots, \ell(R) \quad \Longrightarrow \quad b_i(\rho_i\lambda)(\boldsymbol{P}) \cdot \boldsymbol{e} = 0, \quad i = 1, \ldots, \ell(R) \\ \Longrightarrow 0 &= \sum_{i=1}^{\ell(R)} b_i \left(\rho_i\lambda\right)(\boldsymbol{P}) \cdot \boldsymbol{e} = \sum_{i=1}^{\ell(R)} b_i \sum_{j=1}^{n} \rho_i(P_j)\lambda(P_j)e_j = \sum_{j=1}^{n}\sum_{i=1}^{\ell(R)} b_i\rho_i(P_j)\lambda(P_j)e_j \\ &= \sum_{j=1}^{n} \rho(P_j)\lambda(P_j)e_j = (\rho\lambda)(\boldsymbol{P}) \cdot \boldsymbol{e} \end{aligned}$$

□

6.2. Error Locations and Error Values

Next, error locators are put to use to calculate the errors in a received word. The construction requires a third and final auxiliary divisor.

Lemma 6.2.1. *Let A be a divisor of V such that (1) $\operatorname{supp}(A)$ contains none of the P_j's in $\boldsymbol{P}$, (2) $\ell(A) > t$, and (3) $\deg(A) < \deg(D) - 2g + 2 - t$. Let λ be an error locator in $\mathcal{L}(A)$ for $\boldsymbol{e}$. Let $\hat{\boldsymbol{Z}}$ and $\hat{\boldsymbol{z}}$ consist of the P_j's in $\boldsymbol{P}$ and the e_j's in $\boldsymbol{e}$, respectively, such that $\lambda(P_j) = 0$. (Then $\hat{\boldsymbol{Z}}$ contains all t error locations of $\boldsymbol{e}$, as well as perhaps some non-error locations.) Let M be a divisor of V such that (1) $\operatorname{supp}(M)$ contains none of the P_j's in $\boldsymbol{P}$, and (2) $\deg(M) > \deg(A) + 2g - 2$. Then $\hat{\boldsymbol{z}}$ is uniquely determined by any error locator $\lambda \in \mathcal{L}(A)$ and the syndromes $\mu(\boldsymbol{P}) \cdot \boldsymbol{v}$ with respect to functions $\mu \in \mathcal{L}(M)$.*

Proof, following [Pre]. Since poles of λ are confined to the support of A, which is disjoint from the P_j's in $\boldsymbol{P}$, the zeroes-and-poles theorem (4.12.9) requires that the number of elements of $\hat{\boldsymbol{Z}}$ is at most $\deg(D) - 2g + 2 - t$. By linearity of the dot product when one of the operands is held fixed,

$$\mu(\boldsymbol{P}) \cdot \boldsymbol{v} = \mu(\boldsymbol{P}) \cdot (\boldsymbol{u} + \boldsymbol{e}) = \mu(\boldsymbol{P}) \cdot \boldsymbol{u} + \mu(\boldsymbol{P}) \cdot \boldsymbol{e} = \mu(\boldsymbol{P}) \cdot \boldsymbol{e}.$$

This replaces the unknown quantity $\boldsymbol{e}$ with the known quantity $\boldsymbol{v}$. Thus for any word $\boldsymbol{e}$ with λ as an error locator and any function μ without poles in $\boldsymbol{P}$, e.g. for any $\mu \in \mathcal{L}(M)$,

$$\mu(\boldsymbol{P}) \cdot \boldsymbol{e} = \mu(\boldsymbol{P}) \cdot \boldsymbol{v} = \mu(\hat{\boldsymbol{Z}}) \cdot \hat{\boldsymbol{z}}.$$

For uniqueness, suppose $\hat{\boldsymbol{z}}_1$ and $\hat{\boldsymbol{z}}_2$ are two words such that $\mu(\boldsymbol{P}) \cdot \hat{\boldsymbol{z}}_1 = \mu(\boldsymbol{P}) \cdot \hat{\boldsymbol{z}}_2$ for all $\mu \in \mathcal{L}(M)$. Then $\mu(\boldsymbol{P}) \cdot (\hat{\boldsymbol{z}}_1 - \hat{\boldsymbol{z}}_2) = 0$ for all $\mu \in \mathcal{L}(M)$. Therefore $\hat{\boldsymbol{z}}_1 - \hat{\boldsymbol{z}}_2$ is a codeword of $C_p(V, \hat{\boldsymbol{Z}}, M)$. Let d_M be the minimum distance of $C_p(V, \hat{\boldsymbol{Z}}, M)$. By proposition 5.1.5,

$$\begin{aligned} d_M &\geq \deg(M) - 2g + 2 \\ &> \deg(A) + 2g - 2 - 2g + 2 \\ &= \deg(A) \end{aligned}$$

Then $\hat{\boldsymbol{z}}_1 - \hat{\boldsymbol{z}}_2$ has at most $\deg(D) - 2g + 2 - t$ entries. Thus $\operatorname{dist}(\hat{\boldsymbol{z}}_1, \hat{\boldsymbol{z}}_2) \leq \deg(D) - 2g + 2 - t$. Since $d_M > \deg D - 2g + 2 - t$, $\operatorname{dist}(\hat{\boldsymbol{z}}_1, \hat{\boldsymbol{z}}_2) = 0$, which implies $\hat{\boldsymbol{z}}_1 = \hat{\boldsymbol{z}}_2$. □

Proposition 6.2.2. *Let M be as in the previous lemma and let $\mu_1, \ldots, \mu_{\ell(M)}$ be a basis of $\mathcal{L}(M)$. Then $\hat{\boldsymbol{z}}$ is the unique solution of the system of equations*

$$\mu_i(\hat{\boldsymbol{Z}}) \cdot \hat{\boldsymbol{z}} = \mu_i(\boldsymbol{P}) \cdot \boldsymbol{v}$$

Proof. Let $m = \ell(M)$ and $c = \#\hat{\boldsymbol{Z}}$, and write $\mu = \sum_{i=1}^{m} a_i \mu_i$. Then the system of equations expands to

$$\begin{bmatrix} \mu_1(\hat{Z}_1) & \ldots & \mu_1(\hat{Z}_c) \\ \vdots & & \vdots \\ \mu_m(\hat{Z}_1) & \ldots & \mu_m(\hat{Z}_c) \end{bmatrix} \begin{bmatrix} \hat{z}_1 \\ \vdots \\ \hat{z}_c \end{bmatrix} = \begin{bmatrix} \mu_1(\boldsymbol{P}) \cdot v \\ \vdots \\ \mu_m(\boldsymbol{P}) \cdot v \end{bmatrix}.$$

Premultiplying by the coefficients for μ gives

$$\begin{bmatrix} a_1 & \dots a_m \end{bmatrix} \begin{bmatrix} \mu_1(\hat{Z}_1) & \dots & \mu_1(\hat{Z}_c) \\ \vdots & & \vdots \\ \mu_m(\hat{Z}_1) & \dots & \mu_m(\hat{Z}_c) \end{bmatrix} \begin{bmatrix} \hat{z}_1 \\ \vdots \\ \hat{z}_c \end{bmatrix} = \begin{bmatrix} a_1 & \dots a_m \end{bmatrix} \begin{bmatrix} \mu_1(\boldsymbol{P}) \cdot v \\ \vdots \\ \mu_m(\boldsymbol{P}) \cdot v \end{bmatrix}$$
$$\mu(\hat{\boldsymbol{Z}}) \cdot \hat{\boldsymbol{z}} = \mu(\boldsymbol{P}) \cdot \boldsymbol{v}.$$

Thus if $\hat{\boldsymbol{z}}$ is a solution the system of equations, it is a solution of the equation in the lemma as well. □

Remark 6.2.3. Auxiliary divisors R and M satisfying the hypotheses of lemmas 6.1.6 and 6.2.1 may be obtained by

$$R = D - A, \quad M = D$$

if D, A, M, and R are one-point divisors on a common point.

For the first claim, due to the commonality of D and A, $\deg(D - A) = \deg(D) - \deg(A)$. Then

$$\begin{aligned} \deg(D) - \deg(A) &\geq t \quad \text{(constraint on } A) \\ \deg(D) - \deg(A) &\geq t + 2g - 1 \quad \text{since } g \geq 0 \\ \deg(D - A) &\geq t + 2g - 1 \quad \text{as necessary for } R. \end{aligned}$$

For the second claim, the conditions of lemma 6.2.1 require that

$$\deg(D) > \deg(A) + 2g - 2 + t,$$

which implies

$$\deg(D) > \deg(A) + 2g - 2$$

since t is non-negative.

6.3. Choice of Parameters

Construction of a Goppa code requires the following steps.

1. Choose a field $K = \mathbb{F}_q$, e.g. $\mathbb{F}_2$.

2. Choose a smooth curve V.

3. Compute the genus g of V.

4. Choose an extension field L of $\mathbb{F}_q$, e.g. $\mathbb{F}_{2^r}$.

5. Choose an array of distinct points $\boldsymbol{P}$ on $V(L)$.

6. Choose a divisor D.

7. Compute $\ell(D)$.

8. Compute a basis for $\mathcal{L}(D)$.

9. Compute a generator matrix for the dual code, with matrix elements $F_i(P_j)$. This is a parity-check matrix for the primary code.

10. Compute a basis for the kernel of the parity-check matrix. Form a matrix whose rows are the basis vectors. This is a generator matrix for the primary code.

SV decoding further requires the selection of the auxiliary divisors A, M and R. In practice, these choices must be related to the desired code parameters n, k, and d. Furthermore, since the SV algorithm decodes only up to a minimum error weight t, perhaps less than the maximum possible $\lfloor \frac{d-1}{2} \rfloor$ as discussed in section 3.3, t must here be treated as a fifth code parameter in addition to n, k, d and q.

The relationships between Goppa parameters and code parameters are as follows:

- The code length n is taken from the choice of $\boldsymbol{P}$. Since the elements of $\boldsymbol{P}$ are required to be distinct, n is bounded above by the number of $\mathbb{F}_q$-rational points on V, e.g. the Serre bound as discussed in section 4.6.

- The code dimension k and $\deg(D)$ are related by proposition 5.1.4, namely, $k = n - \deg(D) + g - 1$.

- The minimum distance d is bounded below by the proposition 5.1.5 and above by the Singleton bound (proposition 3.5.1). That is, $\deg(D) - 2g + 2 \leq d \leq n - k + 1$.

- The maximum SV-correctable error weight t is related to the auxiliary divisor A by $t < \ell(A)$ and $\deg(A) < \deg(D) - 2g + 2 - t$.

- The auxiliary divisors R and M may be chosen simply by $R = D - A$ and $M = D$, as shown in remark 6.2.3.

Given the Serre bound discussed in section 4.6, curves of higher genus can yield more rational points, which permits longer codes as discussed in section 3.6. However, the SV algorithm decodes closer to half the minimum distance for curves of lower genus.

For the sample curve $E(\mathbb{F}_8)$ with the 12 points of $\boldsymbol{P}$ taken from $E(\mathbb{F}_8) \setminus \{P_\infty\}$, and with D and $\mathcal{L}(D)$ chosen from table 11, the choices of D, A and t admit the possibilities shown in table 13. The code to be presented in section 6.5 is chosen to be the highest-dimension code in table 13 capable of correcting triple errors. The entries in table 13 corresponding to the sample code are marked in boldface.

Table 13 shows all possible parameter choices permitting construction of a code. For a fixed k, however, t is to be maximimized, so that the code may correct many errors. For each t, $\deg(A)$ is to be minimized, to reduce the operations count in the computation of error locations. These design constraints allow exclusion of most of the possible values of t and $\deg(A)$, resulting in the simpler table 14. Rows where t and $\deg(A)$ are marked with "-" indicate Goppa codes that are constructible, but not SV-decodable.

Figures 6 and 7 give a graphical representation of these optimal parameter choices for curves of degree 5 and 8 respectively. The divisor degrees $\deg(D)$ and $\deg(A)$ are shown, along with t, Goppa-minimum $\lfloor \frac{d-1}{2} \rfloor$, and Singleton-maximum $\lfloor \frac{d-1}{2} \rfloor$, plotted against code dimension k. The code length is held fixed at $n = 200$. Note that the plot depends only on the genus, namely 6 for a smooth quintic and 21 for a smooth octic, and the code length, except for the precise locations of the Weierstrass gaps.

deg(D)	k	$\lfloor \frac{d-1}{2} \rfloor$ bounds	t	deg(A)							
4	8	1-2	1	2							
5	7	2	1	2	3						
6	6	2-3	1	2	3	4					
			2		3						
7	5	3	1	2	3	4	5				
			2		3	4					
8	**4**	**3-4**	1	2	3	4	5	6			
			2		3	4	5				
			3			**4**					
9	3	4	1	2	3	4	5	6	7		
			2		3	4	5	6			
			3			4	5				
10	2	4-5	1	2	3	4	5	6	7	8	
			2		3	4	5	6	7		
			3			4	5	6			
			4				5				
11	1	5	1	2	3	4	5	6	7	8	9
			2		3	4	5	6	7	8	
			3			4	5	6	7		
			4				5	6			

Table 13. Parameter choices for a curve of genus 1, $n = 12$.

n	deg(D)	$\ell(D)$	k	d min.	d max.	$\lfloor \frac{d-1}{2} \rfloor$ min.	$\lfloor \frac{d-1}{2} \rfloor$ max.	t	deg(A)
12	2	2	10	2	3	0	1	-	-
12	3	3	9	3	4	1	1	-	-
12	4	4	8	4	5	1	2	1	2
12	5	5	7	5	6	2	2	1	2
12	6	6	6	6	7	2	3	2	3
12	7	7	5	7	8	3	3	2	3
12	**8**	**8**	**4**	**8**	**9**	**3**	**4**	**3**	**4**
12	9	9	3	9	10	4	4	3	4
12	10	10	2	10	11	4	5	4	5
12	11	11	1	11	12	5	5	4	5

Table 14. Optimal parameter choices for a curve of genus 1, $n = 12$.

The exact case of the Riemann-Roch theorem applies for $k \leq 128$. For $k > 128$, the gaps are visible for the values of $\deg(A)$ in the lower right-hand corner of the plot. These gaps are more easily visible in figure 7 due to the higher genus. Observe that values of k are bounded away from 1 and $n-1$, in accordance with corollary 5.1.7. As well, the spacing between t and the designed minimum distance $\lfloor \frac{d-1}{2} \rfloor$ widens as g increases. This is a limitation of the SV algorithm; see [Pre], [HvLP] for more powerful decoding techniques.

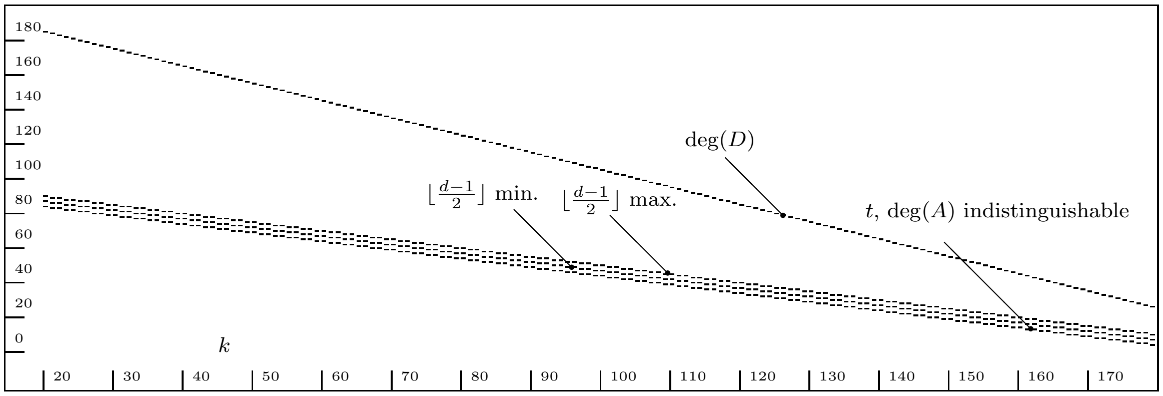

Figure 6. Optimal parameter choices vs. k for a curve of genus 6, $n = 200$.

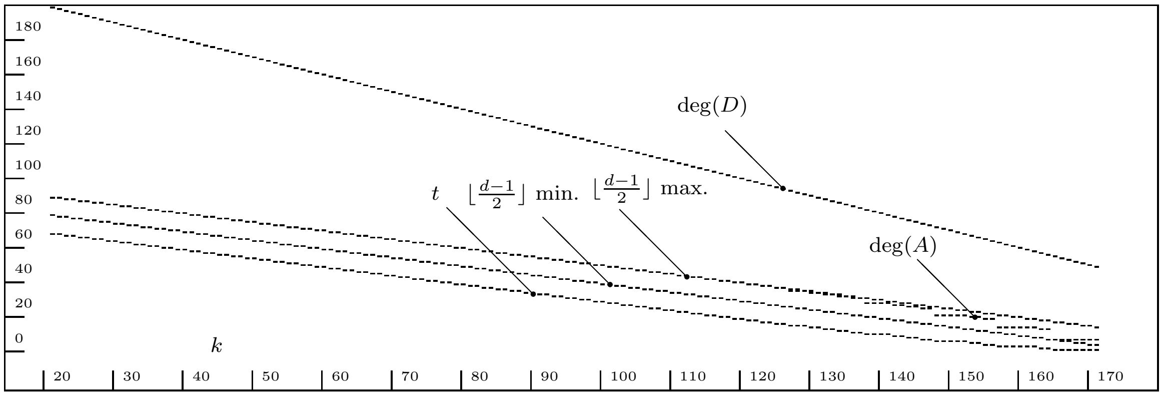

Figure 7. Optimal parameter choices vs. k for a curve of genus 21, $n = 200$.

6.4. The SV Algorithm

The selection of parameters in chapter 6.3 constitutes the *off-line* portion, or design phase, of the SV algorithm. That is, the steps in that section may be performed before any message words are received. In this section the *on-line* portion of the SV algorithm is presented. The steps here simply summarize the propositions already proved.

Algorithm 6.4.1 (On-line portion of SV decoding algorithm).

```
1. Compute $S$:
    For $i = 1, \dots, \ell(R)$
        for $j = 1, \dots, \ell(A)$
            $S_{ij} = (\rho_i(\boldsymbol{P}) * \lambda_j(\boldsymbol{P})) \cdot \boldsymbol{v}$.
    If $S$ is the zero matrix,
        $\hat{\boldsymbol{e}} = \boldsymbol{0}$; stop.
2. Compute $\lambda$:
    Compute a basis for the kernel of $S$.
    If basis is empty
        decoding failure; stop.
    Let $\boldsymbol{c}$ be the first basis vector.
3. Compute the error locations:
    For $j = 1, \dots, n$
        if $\lambda(P_j) = 0$ (i.e. if $\sum_{i=1}^{\ell(A)} c_i \lambda_i(P_j) = 0$)
            append $P_j$ to $\hat{\boldsymbol{Z}}$.
4. Compute the error values:
    For $i = 1, \dots, \ell(M)$
        for $j = 1, \dots, \#\hat{\boldsymbol{Z}}$
            $T_{ij} = \mu_i(P_j)$.
    For $i = 1, \dots, \ell(M)$
```

```
        $b_i = \mu_i(\boldsymbol{P}) \cdot \boldsymbol{v}$.
    Let $\hat{\boldsymbol{z}}$ be the unique solution of $T\hat{\boldsymbol{z}} = \boldsymbol{b}$.
    If no such solution exists,
        decoding failure; stop.
    For each index $j$ of $\hat{\boldsymbol{z}}$
        $e_j = \hat{z}_j$.
    For each non-index $j$
        $e_j = 0$.
```

Note the $\rho_i(\boldsymbol{P}_j)$'s, $\lambda_i(\boldsymbol{P}_j)$'s, and $\mu_i(\boldsymbol{P})$'s may be precomputed. In fact, given the common one-point construction described in remark 6.2.3, these three matrices may be obtained by simply taking the first $\ell(R)$, $\ell(A)$, and $\ell(M)$ rows of H, respectively. Thus the decoding circuitry does not need to know about $\boldsymbol{P}$, and does not need to be able to evaluate rational functions.

Let $\mathcal{R}$ be the $\ell(R) \times n$ matrix $(\rho_i(\boldsymbol{P}_j))_{i,j}$, let $\mathcal{A}$ be the $\ell(A) \times n$ matrix $(\lambda_i(\boldsymbol{P}_j))_{i,j}$, and let $\mathcal{M}$ be the $\ell(M) \times n$ matrix $(\mu_i(\boldsymbol{P}_j))_{i,j}$. Also, given a matrix $\mathcal{B}$, let the notation $\mathcal{B}_i$ denote the ith row of $\mathcal{B}$. Then the decoding algorithm is as follows.

Algorithm 6.4.2 (Optimized on-line portion of SV decoding algorithm).

```
1. Compute $S$:
    For $i = 1, \ldots, \ell(R)$
        for $j = 1, \ldots, \ell(A)$
            $S_{ij} = (\mathcal{R}_i * \mathcal{A}_j) \cdot \boldsymbol{v}$.
    If $S$ is the zero matrix,
        $\hat{\boldsymbol{e}} = \boldsymbol{0}$; stop.
2. Compute $\lambda$:
    Compute a basis for the kernel of $S$.
    If basis is empty
```

```
        decoding failure; stop.
    Let $\boldsymbol{c}$ be the first basis vector.
3. Compute the error locations:
    Compute the length-$n$ vector $\boldsymbol{y} = c\mathcal{A}$.
    For $j = 1, \ldots, \#\boldsymbol{Z}$
        if $y_j = 0$
            $j$ is an error-location index.
4. Compute the error values:
    Form $T$ by taking the $j$th column of $\mathcal{M}$ for each error-location index $j$.
    Let $\boldsymbol{b} = \mathcal{M}\boldsymbol{v}$.
    Let $\hat{\boldsymbol{z}}$ be the unique solution of $T\hat{\boldsymbol{z}} = \boldsymbol{b}$.
    If no such solution exists,
        decoding failure; stop.
    For each index $j$ of $\hat{\boldsymbol{z}}$
        $e_j = \hat{z}_j$.
    For each non-index $j$
        $e_j = 0$.
```

6.5. Example

Define a linear code C_1 over $\mathbb{F}_8$ as follows. As discussed in section 6.3, let $V = E(\mathbb{F}_8)$ which was found in example 4.5.1 to have genus 1. Let $\boldsymbol{P}$ be the twelve affine points from example 4.5.1, reproduced for convenience in table 15. Let $D = 8P_\infty$ as chosen in section 6.3, with basis shown in table 12. Let $A = 4P_\infty$ as chosen in section 6.3; choose $R = D - A = 4P_\infty$ and $M = D = 8P_\infty$. Then D and the auxiliary divisors R, A, and M have bases as shown in table 17.

This code has length $n = 12$, from the length of $\boldsymbol{P}$. By proposition 5.1.4, the code has dimension $k = n - \deg(D) + g = 1 = 12 - 8 = 4$. As discussed in section 6.3, the maximum

P_1	$[2,0,1]$
P_2	$[4,0,1]$
P_3	$[6,0,1]$
P_4	$[2,1,1]$
P_5	$[4,1,1]$
P_6	$[6,1,1]$
P_7	$[3,2,1]$
P_8	$[5,4,1]$
P_9	$[7,6,1]$
P_{10}	$[3,3,1]$
P_{11}	$[5,5,1]$
P_{12}	$[7,7,1]$

Table 15. Curve points for sample $[12,4,8]_8$ code on $E(\mathbb{F}_8)$.

correctable error weight is $t = 3$. Since $\#C_1 = q^k = 8^4 = 4096$, it is feasible to enumerate all code words of C_1 and compute the weight distribution shown in table 16. To perform this computation, loop over all of $\mathbb{F}_q^k = \mathbb{F}_8^4$, encode each vector, and compute the weight of the resulting code word. Then take the minimum non-zero weight. In particular, C_1 has minimum distance $d = 8$, fitting within the Goppa and Singleton bounds $8 - 9$ as shown in table 17. Thus C_1 is a $[12,4,8]_8$ linear code.

w	0	1	2	3	4	5	6	7	8	9	10	11	12
#	1	0	0	0	0	0	0	0	273	448	1176	1344	854

Table 16. Empirical weight distribution for sample $[12,4,8]_8$ code on $E(\mathbb{F}_8)$.

$L(D)$:	1,	X/Z,	Y/Z,	X^2/Z^2,	XY/Z^2,	Y^2/Z^2,	X^2Y/Z^3	XY^2/Z^3
$L(R)$:	1,	X/Z,	Y/Z,	X^2/Z^2				
$L(A)$:	1,	X/Z,	Y/Z,	X^2/Z^2				
$L(M)$:	1,	X/Z,	Y/Z,	X^2/Z^2,	XY/Z^2,	Y^2/Z^2,	X^2Y/Z^3,	XY^2/Z^3

Table 17. Basis functions for sample $[12,4,8]_8$ code on $E(\mathbb{F}_8)$.

The generator and parity-check matrices for C_1 are computed as shown in chapter 5:

$$H = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 2 & 4 & 6 & 2 & 4 & 6 & 3 & 5 & 7 & 3 & 5 & 7 \\ 0 & 0 & 0 & 1 & 1 & 1 & 2 & 4 & 6 & 3 & 5 & 7 \\ 4 & 6 & 2 & 4 & 6 & 2 & 5 & 7 & 3 & 5 & 7 & 3 \\ 0 & 0 & 0 & 2 & 4 & 6 & 6 & 2 & 4 & 5 & 7 & 3 \\ 0 & 0 & 0 & 1 & 1 & 1 & 4 & 6 & 2 & 5 & 7 & 3 \\ 0 & 0 & 0 & 4 & 6 & 2 & 1 & 1 & 1 & 4 & 6 & 2 \\ 0 & 0 & 0 & 2 & 4 & 6 & 7 & 3 & 5 & 4 & 6 & 2 \end{bmatrix}$$

$$G = \begin{bmatrix} 3 & 2 & 6 & 7 & 7 & 7 & 4 & 5 & 1 & 0 & 0 & 0 \\ 7 & 2 & 4 & 7 & 2 & 4 & 1 & 0 & 0 & 1 & 0 & 0 \\ 6 & 3 & 4 & 6 & 3 & 4 & 0 & 1 & 0 & 0 & 1 & 0 \\ 5 & 0 & 3 & 1 & 5 & 2 & 4 & 5 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

The auxiliary matrices $\mathcal{R}$, $\mathcal{A}$, and $\mathcal{M}$ are taken from the first 4, 4, and 8 rows of H, respectively.

Let

$$\boldsymbol{m} = (1, 1, 1, 1).$$

This encodes to

$$\boldsymbol{m}G = \boldsymbol{u} = (7, 3, 5, 7, 3, 5, 1, 1, 1, 1, 1, 1).$$

Let the error be

$$\boldsymbol{e} = (0, 0, 0, 0, 0, 1, 2, 3, 0, 0, 0, 0)$$

such that the received word is

$$\boldsymbol{v} = (7, 3, 5, 7, 3, 4, 3, 2, 1, 1, 1, 1)$$

Then

$$S = \begin{bmatrix} 0 & 4 & 2 & 1 \\ 4 & 1 & 7 & 5 \\ 2 & 7 & 3 & 3 \\ 1 & 5 & 3 & 4 \end{bmatrix}$$

which is

$$\begin{bmatrix} 1 & 0 & 0 & 3 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 7 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

in row-echelon form, with kernel basis

$$\{(3, 1, 7, 1)\}.$$

Thus

$$\lambda(\boldsymbol{P}) = 3\lambda_1(\boldsymbol{P}) + \lambda_2(\boldsymbol{P}) + 7\lambda_3(\boldsymbol{P}) + \lambda_4(\boldsymbol{P})$$

is the row vector $(3, 1, 7, 1)$ times the first four rows of H, namely,

$$\lambda(\boldsymbol{P}) = (5, 1, 7, 2, 6, 0, 0, 0, 3, 7, 7, 4).$$

The error locations are then P_6, P_7, and P_8. Likewise, T is read off from columns 6, 7, and 8 of the first 8 rows of H:

$$T = \begin{bmatrix} 1 & 1 & 1 \\ 6 & 3 & 5 \\ 1 & 2 & 4 \\ 2 & 5 & 7 \\ 6 & 6 & 2 \\ 1 & 4 & 6 \\ 2 & 1 & 1 \\ 6 & 7 & 3 \end{bmatrix}, \quad b = (0, 4, 2, 1, 7, 3, 3, 6).$$

The augmented matrix

$$T \mid b = \begin{bmatrix} 1 & 1 & 1 & 0 \\ 6 & 3 & 5 & 4 \\ 1 & 2 & 4 & 2 \\ 2 & 5 & 7 & 1 \\ 6 & 6 & 2 & 7 \\ 1 & 4 & 6 & 3 \\ 2 & 1 & 1 & 3 \\ 6 & 7 & 3 & 6 \end{bmatrix}$$

row-reduces to

$$\begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 2 \\ 0 & 0 & 1 & 3 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

with solution

$$\hat{\boldsymbol{z}} = (1, 2, 3).$$

Putting these values at positions 6, 7, and 8 of $\hat{\boldsymbol{e}}$ and zeroes elsewhere gives

$$\hat{\boldsymbol{e}} = (0, 0, 0, 0, 0, 1, 2, 3, 0, 0, 0, 0).$$

which exactly matches the original error word $\boldsymbol{e}$.

For another example, choose $\boldsymbol{m}$ as before, but let

$$\boldsymbol{e} = (0, 0, 0, 0, 0, 1, 2, 3, 4, 0, 0, 0),$$

This is an error of weight 4, whereas C_1 is designed to handle only at most $t = 3$ errors. The syndrome matrix is found to be

$$S = \begin{bmatrix} 4 & 5 & 7 & 6 \\ 5 & 6 & 1 & 6 \\ 7 & 1 & 0 & 7 \\ 6 & 6 & 7 & 6 \end{bmatrix}$$

which row-reduces to

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

which has no non-zero solutions. Thus, the algorithm produces a decoding failure.

CHAPTER 7

Data

7.1. Construction of Curves and Code Parameters

In this chapter, parameters are computed and displayed for codes over several curves. The curve E discussed in previous chapters is from an exercise of [Wal]; the Klein quartic (section 7.3) is used in [Pre] and [HvLP]. The Reed-Solomon codes (section 7.4) are well-known to be a special case of Goppa codes, as discussed in [Wal], [HvLP]. The sextic curve (section 7.5) was selected arbitrarily. The Hermitian curve (section 7.6) is discussed in [HvLP].

For each curve V, the following are presented: smoothness, genus, point counts vs. the Serre bounds for $q = 2, 4, \ldots, 1024$, Weierstrass gaps, and a basis for $\mathcal{L}(D)$. For the resulting codes, the code length n is taken from a choice of $\boldsymbol{P}$ which is in turn selected from $V(\mathbb{F}_q)$ for one or more choices of q. The code dimension k is selected within the constraints of section 6.3. The maximum SV-correctable error weight t also follows from section 6.3.

The results of chapter 5 specify precisely the parameters n, k, and q for Goppa codes, but only give bounds on the minimum distance d. If a weight distribution is computed, the true minimum distance d is obtained as a result. Recall that a k-dimensional code over $\mathbb{F}_q$ has q^k elements, and that for linear codes the minimum distance is equal to the minimum non-zero weight. Thus, for small q and k one may simply iterate over all q^k elements of $\mathbb{F}_q^k$, multiplying each by a generator matrix to obtain a complete list of code words, then take the minimum non-zero weight. Likewise, if k is large but q^{n-k} is small, then the MacWilliams identities ([MS], [Sud]) permit the weight distribution of the primary code to be obtained from that of the dual code, which in turn may be done using the technique just described. If q^k and q^{n-k} are both large — say, greater than a million — then this approach is prohibitive. This applies in particular to curves of higher genus, as shown in corollary 5.1.7.

7.2. Automation

Computations were performed using SPFFL, a C++ small-prime finite-field library developed by the author. The off-line portion of the algorithm was partially automated, as is explained in greater detail in the following paragraph; computation of G and H, along with all of the on-line portion of the algorithm including encoding and decoding, is an exercise in simple computational linear algebra and was fully automated.

Smoothness was checked by hand. Computation of the genus is easy for smooth curves using the Plücker formula. However, in general, computation of genus for non-smooth curves of higher degree is one of the harder problems in the theory of AG codes [Pre]. All curves considered in this paper are defined over $\mathbb{F}_2$; codes presented are in some low-degree extension. Given a chosen extension field $\mathbb{F}_q$ of $\mathbb{F}_2$, listing of points was automated as described in section 4.5, although the partition of points between the single point forming D and the points forming $\boldsymbol{P}$ was manual. The intersection multiplicities necessary for divisors was were computed manually. Weight distributions were computed automatically.

7.3. The Klein Quartic

Let V_4 be defined by the Klein quartic,

$$F(X, Y, Z) = X^3Y + Y^3Z + Z^3X$$

Partial derivatives are

$$\frac{\partial F}{\partial X} = X^2Y + Z^3, \quad \frac{\partial F}{\partial Y} = Y^2Z + X^3, \quad \frac{\partial F}{\partial Z} = Z^2X + Y^3.$$

If there is a singular point, it is either infinite or affine. In the former case $Z = 0$ which requires $X^3 = Y^3 = 0$, whence $X = Y = Z = 0$ which is not a projective point. In the latter case $Z = 1$, and for F and its partials to all be zero requires

$$X^3Y + Y^3 + X = 0, \quad X^2Y = 1, \quad Y^2 = X^3, \quad X = Y^3.$$

Substituting $X^2Y = 1$ into the first condition yields $X + Y^3 + X = Y^3 = 0$ which implies $Y = 0$. Then the original equation requires $X = 0$, but $X^2Y = 1$ requires $X \neq 0$. This contradiction shows that the Klein quartic is smooth when defined over $\mathbb{F}_2$. Since the degree of defining polynomial is 4, the genus of the curve is 3. Point-counting results are shown in tables 18 and 19.

$\mathbb{F}_2$	$\mathbb{F}_4$	$\mathbb{F}_8$			$\mathbb{F}_{16}$	
$[1,0,0]$	$[1,0,0]$	$[1,0,0]$	$[6,1,1]$	$[5,5,1]$	$[1,0,0]$	$[6,9,1]$
$[0,1,0]$	$[0,1,0]$	$[0,1,0]$	$[2,4,1]$	$[7,7,1]$	$[0,1,0]$	$[7,\mathrm{d},1]$
$[0,0,1]$	$[0,0,1]$	$[0,0,1]$	$[4,6,1]$	$[3,4,1]$	$[0,0,1]$	$[6,\mathrm{e},1]$
	$[2,3,1]$	$[1,2,1]$	$[6,2,1]$	$[5,6,1]$	$[2,\mathrm{c},1]$	$[7,\mathrm{b},1]$
	$[3,2,1]$	$[1,4,1]$	$[2,5,1]$	$[7,2,1]$	$[4,\mathrm{f},1]$	$[8,6,1]$
		$[1,6,1]$	$[4,7,1]$	$[3,7,1]$	$[3,\mathrm{a},1]$	$[\mathrm{c},7,1]$
		$[2,1,1]$	$[6,3,1]$	$[5,3,1]$	$[5,8,1]$	$[\mathrm{f},6,1]$
		$[4,1,1]$	$[3,3,1]$	$[7,5,1]$	$[6,7,1]$	$[\mathrm{a},7,1]$
					$[7,6,1]$	

Table 18. Points on the Klein quartic for $\mathbb{F}_2$, $\mathbb{F}_4$, $\mathbb{F}_8$, and $\mathbb{F}_{16}$

L	$\#\mathbb{P}^2(L)$	$\#V_4(L)$	Serre minimum	Serre maximum
$\mathbb{F}_2$	7	3	-3	9
$\mathbb{F}_4$	21	5	-7	17
$\mathbb{F}_8$	73	24	-6	24
$\mathbb{F}_{16}$	273	17	-7	41
$\mathbb{F}_{32}$	1057	33	0	66
$\mathbb{F}_{64}$	4161	38	17	113
$\mathbb{F}_{128}$	16513	129	63	195
$\mathbb{F}_{256}$	65793	257	161	353
$\mathbb{F}_{512}$	262657	528	378	648
$\mathbb{F}_{1024}$	1049601	1025	833	1217

Table 19. Point counts vs. Serre bounds for the Klein quartic.

Let

$$P_1 = [1,0,0], \quad P_2 = [0,1,0], \quad P_3 = [0,0,1].$$

The intersection divisor of X is obtained by setting $X = 0$ in F, to obtain $Y^3Z = 0$. This forces either $Y = 0$ or $Z = 0$. In the former case, $P_3 = [0, 0, 1]$ is an intersection point; dehomogenization at Z gives multiplicity 3. Likewise, $P_2 = [0, 1, 0]$ has intersection multiplicity 1. Thus $\operatorname{div}(X) = 3P_3 + P_2$. Similarly, $\operatorname{div}(Y) = 3P_1 + P_3$ and $\operatorname{div}(Z) = 3P_2 + P_1$. Thus

$$\operatorname{div}(X^iY^j/Z^{i+j}) = (-i + 2j)P_1 + (-2i - 3j)P_2 + (3i + j)P_3$$

This permits a single-point divisor on P_2 as long as $-i + 2j \geq 0$, i.e. for $2j \geq i$.

Even though such X^iY^j/Z^{i+j} are pole-free at all points other than P_2, including P_1, an annoyance arises at P_1 which has Z coordinate 0. This 0/0 situation is removable as described in remark 4.12.8, provided one is willing to search for an equivalent function in the function field $\mathbb{F}_q(V_4)$. Alternatively, one may prefer not to make that effort, and instead simply exclude P_1 from $\boldsymbol{P}$. The latter is the approach taken here. That is to say, all points on the curve are included in the intersection-divisor analysis, as is necessary for the Riemann-Roch theory to work correctly. However, not all points other than D need to be included in $\boldsymbol{P}$. In general, when using functions of the form X^iY^j/Z^{i+j}, 0/0 situations only arise when $Z = 0$, which are points at infinity with respect to Z. Such points are intersections of the curve with the line at infinity $Z = 0$, and the number of such points is constrained by Bezout's theorem. Thus, omitting these troublesome points from $\boldsymbol{P}$ does not significantly affect code length.

With the choice of $n = 15$, i.e. with $\boldsymbol{P}$ consisting of all points on $V_4(\mathbb{F}_{16})$ other than the infinite points P_1 and P_2, the formulas of section 6.3 permit code parameters shown in table 21. As before, rows where t and $\deg(A)$ are marked with "-" indicate constructible codes that are not SV-decodable. Selecting the last row gives a code with $n = 15$, $k = 3$, $q = 16$, and d in the range 10 to 13. Since $q^k = 4096$ which is small, the weight distribution is readily computed as shown in table 23. Thus a $[15, 3, 11]_{16}$ code is obtained.

Over $\mathbb{F}_{256}$ there are 255 non-infinite points, so n may be as high as 255. Table 22 shows the optimal code parameters in this case. Even for the smallest k, namely 3, $q^k = 256^3 = 16,777,216$ so brute-force weight enumeration is infeasible: each of these 16,777,216 code words is a vector of length 255, so there are approximately 4 billion weights to compute; also, generation of each code word requires a 3×255 matrix-times-vector multiplication.

r	i	j	$i+j$	ϕ	$-i+2j$	$-2i-3j$	$3i+j$
0,1,2	0	0	0	1	0	0	0
3,4	0	1	1	Y/Z	2	-3	1
5	1	1	2	XY/Z^2	1	-5	4
6	0	2	2	Y^2/Z^2	4	-6	2
7	2	1	3	X^2Y/Z^3	0	-7	7
8	1	2	3	XY^2/Z^3	3	-8	5
9	0	3	3	Y^3/Z^3	6	-9	3
10	2	2	4	X^2Y^2/Z^4	2	-10	8
11	1	3	4	XY^3/Z^4	5	-11	6
12	3	2	5	X^3Y^2/Z^5	1	-12	11
13	2	3	5	X^2Y^3/Z^5	4	-13	9
14	4	2	6	X^4Y^2/Z^6	0	-14	14
15	3	3	6	X^3Y^3/Z^6	3	-15	12

Table 20. Selected function bases for the Klein quartic.

n	$\deg(D)$	$\ell(D)$	k	d min.	d max.	$\lfloor\frac{d-1}{2}\rfloor$ min.	$\lfloor\frac{d-1}{2}\rfloor$ max.	t	$\deg(A)$
15	4	2	13	0	3	0	1	-	-
15	5	3	12	1	4	0	1	-	-
15	6	4	11	2	5	0	2	-	-
15	7	5	10	3	6	1	2	-	-
15	8	6	9	4	7	1	3	-	-
15	9	7	8	5	8	2	3	1	3
15	10	8	7	6	9	2	4	1	3
15	11	9	6	7	10	3	4	1	3
15	12	10	5	8	11	3	5	2	5
15	13	11	4	9	12	4	5	2	5
15	14	12	3	10	13	4	6	3	6

Table 21. Code parameters for the Klein quartic over $\mathbb{F}_{16}$.

7.4. Reed-Solomon Codes

Let V_R be defined by the line $Y = 0$. The partial derivative $\frac{\partial F}{\partial Y}$ is 1, so V_R is smooth; it has genus 0. Points on V_R are the projective line $\mathbb{P}^1(\mathbb{F}_q)$ contained in $\mathbb{P}^2(\mathbb{F}_q)$, namely, the q affine points $[\alpha, 0, 1]$ for all $\alpha \in \mathbb{F}_q$, along with the single point at infinity $P_\infty = [1, 0, 0]$. Thus $\#V_R = q + 1$. This is in agreement with the exact point count prescribed by the Serre bounds as discussed in section 4.6.

n	$\deg(D)$	$\ell(D)$	k	d min.	d max.	$\lfloor \frac{d-1}{2} \rfloor$ min.	$\lfloor \frac{d-1}{2} \rfloor$ max.	t	$\deg(A)$
255	4	2	253	0	3	0	1	-	-
255	5	3	252	1	4	0	1	-	-
255	6	4	251	2	5	0	2	-	-
255	7	5	250	3	6	1	2	-	-
255	8	6	249	4	7	1	3	-	-
255	9	7	248	5	8	2	3	1	3
255	10	8	247	6	9	2	4	1	3
255	11	9	246	7	10	3	4	1	3
255	12	10	245	8	11	3	5	2	5
255	13	11	244	9	12	4	5	2	5
255	14	12	243	10	13	4	6	3	6
255	15	13	242	11	14	5	6	3	6
255	16	14	241	12	15	5	7	4	7
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
255	247	245	10	243	246	121	122	119	122
255	248	246	9	244	247	121	123	120	123
255	249	247	8	245	248	122	123	120	123
255	250	248	7	246	249	122	124	121	124
255	251	249	6	247	250	123	124	121	124
255	252	250	5	248	251	123	125	122	125
255	253	251	4	249	252	124	125	122	125
255	254	252	3	250	253	124	126	123	126

Table 22. Code parameters for the Klein quartic over $\mathbb{F}_{256}$.

wt	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
#	1	0	0	0	0	0	0	0	0	0	0	270	555	1650	1620

Table 23. Weight distribution for the Klein quartic over $\mathbb{F}_{16}$.

The intersection divisor $\operatorname{div}(X)$ is obtained from $X = Y = 0$, namely, $1P_0$ where $P_0 = [0, 0, 1]$. All points on the curve V_R appear in $\operatorname{div}(Y)$; the support of $\operatorname{div}(Z)$ is $Y = Z = 0$, i.e. $\operatorname{div}(Z) = 1P_\infty$. Then

$$\operatorname{div}(X^i/Z^i) = iP_0 - iP_\infty$$

and a basis for $\mathcal{L}(rP_\infty)$ is as shown in table 24. Code parameters are shown in table 25.

Let $\boldsymbol{P}$ be all of $V_R(\mathbb{F}_{16})$ except the point at infinity, which is used for the one-point divisor, and arbitrarily exclude $[0, 0, 1]$ to get $n = 15$ as in the Klein-quartic example. Then take $k = 3$, again matching the Klein-quartic example. The resulting code has weight distribution as shown in table 26. Thus a $[15, 3, 13]_{16}$ code is obtained. Call this a *geometric Reed-Solomon code*. This minimum distance, 13, is better than that of the code constructed using the Klein quartic using the same n and k, namely, $d = 11$. The duals of the geometric Reed-Solomon codes are equivalent to the classical Reed-Solomon codes ([Wal], [MS]). It can be shown [Wal] that in fact the Reed-Solomon codes always have minimum distance meeting the Singleton bound as discussed in section 3.5. Codes with minimum distance meeting this bound are called *maximum distance separable*, or *MDS*. While this is a strength of Reed-Solomon codes, their weakness is that their code length is limited to $q+1$. This motivates one of the reasons for study of Goppa codes, as was mentioned in section 3.6. See also [HvLP] section 2.8 for further comparison of Reed-Solomon and Goppa codes. The code over the Klein quartic presented in section 7.3 is no longer than the Reed-Solomon code presented here; much longer Goppa codes are discussed in section 7.6.

7.5. A Sextic Curve

Let V_6 be defined by $X^6 + XYZ^4 + Y^5Z + Z^6 = 0$. Partial derivatives are

$$\frac{\partial F}{\partial X} = YZ^4, \quad \frac{\partial F}{\partial Y} = XZ^4 + Y^4Z, \quad \frac{\partial F}{\partial Z} = Y^5.$$

The third requires $Y = 0$, so any singular points are common zeroes of $X^6 + Z^6$ and XZ^4. If $X = 0$, then $Z = 0$ and vice versa, but $[0, 0, 0]$ is not a projective point. Therefore V_6 is smooth, and since it is sextic it has genus 10. Point counts are summarized in table 27.

r	i	ϕ
0	0	1
1	1	X/Z
2	2	X^2/Z^2
3	3	X^3/Z^3
4	4	X^4/Z^4
5	5	X^5/Z^5
6	6	X^6/Z^6
7	7	X^7/Z^7
8	8	X^8/Z^8
9	9	X^9/Z^9

Table 24. Selected function bases for the Reed-Solomon curve

n	$\deg(D)$	$\ell(D)$	k	d min.	d max.	$\lfloor \frac{d-1}{2} \rfloor$ min.	$\lfloor \frac{d-1}{2} \rfloor$ max.	t	$\deg(A)$
15	2	3	12	4	4	1	1	1	1
15	3	4	11	5	5	2	2	1	1
15	4	5	10	6	6	2	2	2	2
15	5	6	9	7	7	3	3	2	2
15	6	7	8	8	8	3	3	3	3
15	7	8	7	9	9	4	4	3	3
15	8	9	6	10	10	4	4	4	4
15	9	10	5	11	11	5	5	4	4
15	10	11	4	12	12	5	5	5	5
15	11	12	3	13	13	6	6	5	5
15	12	13	2	14	14	6	6	6	6
15	13	14	1	15	15	7	7	6	6
15	14	15	0	16	16	7	7	7	7

Table 25. Code parameters for the Reed-Solomon curve over $\mathbb{F}_{16}$.

wt	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
#	1	0	0	0	0	0	0	0	0	0	0	0	0	1575	675	1845

Table 26. Weight distribution for the Reed-Solomon curve over $\mathbb{F}_{16}$.

For the divisor $\mathrm{div}(X)$, $X = 0$ gives

$$0 = Y^5Z + Z^6 = Z(Y^5 + Z^5)$$

If $Z = 0$, then $P_2 = [0, 1, 0]$ is a zero; else, there are $[0, 1, 1]$ and the four points $[0, 8, 1]$, $[0, \mathrm{c}, 1]$, $[0, \mathrm{f}, 1]$, and $[0, \mathrm{a}, 1]$ in $\mathbb{F}_{16}$. By Bezout's theorem, the line $X = 0$ intersects V_6 at each of these six points with multiplicity 1. For $\mathrm{div}(Y)$, $Y = 0$ gives

$$0 = X^6 + Z^6 = (X^3 + Z^3)^2$$

with zeroes $[1, 0, 1]$, and the pair $[2, 0, 1]$, $[3, 0, 1]$ in $\mathbb{F}_4$, each with multiplicity 2. For $\mathrm{div}(Z)$, $Z = 0$ gives $X^6 = 0$, with P_2 a zero of multiplicity 6. Then

$$\mathrm{div}(X^iY^j/Z^{i+j}) = (-5i - 6j)P_2 + iQ_1 + iQ_2 + iQ_3 + iQ_4 + iQ_5 + 2jQ_6 + 2jQ_7 + 2jQ_8.$$

Code parameters are shown in table 29. Weight distributions are impractical to compute by corollary 5.1.7.

L	$\#\mathbb{P}^2(L)$	$\#V_4(L)$	Serre minimum	Serre maximum
$\mathbb{F}_2$	7	4	-17	23
$\mathbb{F}_4$	21	8	-35	45
$\mathbb{F}_8$	73	10	-41	59
$\mathbb{F}_{16}$	273	24	-63	97
$\mathbb{F}_{32}$	1057	24	-77	143
$\mathbb{F}_{64}$	4161	68	-95	225
$\mathbb{F}_{128}$	16513	88	-91	349
$\mathbb{F}_{256}$	65793	304	-63	577
$\mathbb{F}_{512}$	262657	424	63	963
$\mathbb{F}_{1024}$	1049601	1008	385	1665

Table 27. Point counts vs. Serre bounds for the sextic.

7.6. A Hermitian Curve

Let V_{17} be defined by $X^{16}Y + XY^{16} + Z^{17}$. Partial derivatives are

$$\frac{\partial F}{\partial X} = Y^{16}, \quad \frac{\partial F}{\partial Y} = X^{16}, \quad \frac{\partial F}{\partial Z} = Z^{17}$$

r	i	j	$i+j$	ϕ	$-i+4j$	$4i-j$	$-i-j$
0,1,2,3,4	0	0	0	1	0	0	0
5	1	0	1	X/Z	-5	1	0
6,7,8,9	0	1	1	Y/Z	-6	0	2
10	2	0	2	X^2/Z^2	-10	2	0
11	1	1	2	XY/Z^2	-11	1	2
12,13,14	0	2	2	Y^2/Z^2	-12	0	4
15	3	0	3	X^3/Z^3	-15	3	0
16	2	1	3	X^2Y/Z^3	-16	2	2
17	1	2	3	XY^2/Z^3	-17	1	4
18,19	0	3	3	Y^3/Z^3	-18	0	6
20	4	0	4	X^4/Z^4	-20	4	0
21	3	1	4	X^3Y/Z^4	-21	3	2
22	2	2	4	X^2Y^2/Z^4	-22	2	4
23	1	3	4	XY^3/Z^4	-23	1	6
24	0	4	4	Y^4/Z^4	-24	0	8
25	5	0	5	X^5/Z^5	-25	5	0
26	4	1	5	X^4Y/Z^5	-26	4	2
27	3	2	5	X^3Y^2/Z^5	-27	3	4

Table 28. Selected function bases for the sextic

with simultaneous solutions only at $[0, 0, 0]$ which is not a projective point. Therefore V_{17} is smooth, and has genus 120.

Point counts are as shown in table 30. Note that due to the high genus, the upper Serre bound is not tight with respect to the Serre lower bound. Yet, the number of points over $\mathbb{F}_{256}$ meets the upper Serre bound. As discussed in more detail in [Wal], chapter 5, for $q = s^2$, a *Hermitian curve* is defined by $X^sY + XY^s + Z^{s+1}$ and has $s^3 + 1$ points over $\mathbb{F}_q$. Thus, the family of Hermitian curves gives rise to long codes in the sense of definition 3.6.5. See also [HvLP], section 2.1, for another parameterized family of codes.

Intersection divisors are as follows. Let $P_1 = [1, 0, 0]$ and $P_2 = [0, 1, 0]$. For $X = 0$, one has $Z^{17} = 0$, so $\operatorname{div}(X) = 17P_2$. Likewise, $\operatorname{div}(Y) = 17P_1$. For $Z = 0$, one has

$$XY(X^{15} + Y^{15}) = 0.$$

The line $Z = 0$ intersects V_{17} if $X = 0$ or $Y = 0$, so P_1 and P_2 are intersection points. When $X, Y \neq 0$, $X^{15} + Y^{15} = 0$, from which $X^{15} = Y^{15}$ since the characteristic is 2. Since $\alpha^{15} = 1$ for all $\alpha \in \mathbb{F}_{16}^\times$, there are 15 more intersection points, of the form $[\alpha, 1, 0]$ for $\alpha \in \mathbb{F}_{16}^\times$. This is a total of 17

n	$\deg(D)$	$\ell(D)$	k	d min.	d max.	$\lfloor\frac{d-1}{2}\rfloor$ min.	$\lfloor\frac{d-1}{2}\rfloor$ max.	t	$\deg(A)$
64	18	9	55	0	10	0	4	-	-
64	19	10	54	1	11	0	5	-	-
64	20	11	53	2	12	0	5	-	-
64	21	12	52	3	13	1	6	-	-
64	22	13	51	4	14	1	6	-	-
64	23	14	50	5	15	2	7	-	-
64	24	15	49	6	16	2	7	-	-
64	25	16	48	7	17	3	8	1	5
64	26	17	47	8	18	3	8	1	5
64	27	18	46	9	19	4	9	2	6
64	28	19	45	10	20	4	9	2	6
64	29	20	44	11	21	5	10	2	6
64	30	21	43	12	22	5	10	2	6
64	31	22	42	13	23	6	11	2	6
64	32	23	41	14	24	6	11	3	10
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
64	56	47	17	38	48	18	23	13	23
64	57	48	16	39	49	19	24	14	24
64	58	49	15	40	50	19	24	14	24
64	59	50	14	41	51	20	25	15	25
64	60	51	13	42	52	20	25	15	25
64	61	52	12	43	53	21	26	16	26
64	62	53	11	44	54	21	26	16	26
64	63	54	10	45	55	22	27	17	27

Table 29. Code parameters for the sextic over $\mathbb{F}_8$.

L	$\#\mathbb{P}^2(L)$	$\#V_{17}(L)$	Serre minimum	Serre maximum
$\mathbb{F}_2$	7	3	-237	243
$\mathbb{F}_4$	21	5	-475	485
$\mathbb{F}_8$	73	9	-591	609
$\mathbb{F}_{16}$	273	17	-943	977
$\mathbb{F}_{32}$	1057	33	-1287	1353
$\mathbb{F}_{64}$	4161	65	-1855	1985
$\mathbb{F}_{128}$	16513	129	-2511	2769
$\mathbb{F}_{256}$	65793	4097	-3583	4097
$\mathbb{F}_{512}$	262657	513	-4887	5913
$\mathbb{F}_{1024}$	1049601	1025	-6655	8705

Table 30. Point counts vs. Serre bounds for the Hermitian curve.

intersection points; by Bezout's theorem, each has multiplicity one. Let Q be shorthand for each of the non-infinite intersection points with $Z = 0$. Then $\text{div}(Z) = P_1 + P_2 + Q$. All of the Q points have non-zero X and Y coordinate; P_1 and P_2 each have one or the other non-zero; all 17 of them have Z coordinate 0. Therefore one may use a single-point divisor on either P_1 or P_2; the former is selected. One then has

$$\text{div}(X^i Z^j / Y^{i+j}) = (-17i - 16j)P_1 + (17i + j)P_2 + jQ.$$

Since $17i + j$ and j assume non-negative values and $-17i - 16j$ assumes non-positive values when $i, j \geq 0$, the entire family $X^i Z^j / Y^{i+j}$ is in $\mathcal{L}(rP_1)$ for various r. Since $\text{div}(Y) = 17P_1$, by Bezout's theorem no other points of V_{17} have Y coordinate 0 and thus there are no $0/0$ situations for any of the rational functions $X^i Z^j / Y^{i+j}$.

Code parameters are shown in table 29. Weight distributions are impractical to compute by corollary 5.1.7. The true minimum distance is loosely bounded by the Goppa and Singleton bounds. If it were to lie closer to the upper Singleton bound, the SV algorithm would not be able to take advantage of it. See [Pre], chapter 7, for the full error-processing algorithm.

n	$\deg(D)$	$\ell(D)$	k	d min.	d max.	$\lfloor\frac{d-1}{2}\rfloor$ min.	$\lfloor\frac{d-1}{2}\rfloor$ max.	t	$\deg(A)$
4096	238	119	3977	0	120	0	59	-	-
4096	239	120	3976	1	121	0	60	-	-
4096	240	121	3975	2	122	0	60	-	-
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
4096	253	134	3962	15	135	7	67	-	-
4096	254	135	3961	16	136	7	67	-	-
4096	255	136	3960	17	137	8	68	-	-
4096	256	137	3959	18	138	8	68	1	16
4096	257	138	3958	19	139	9	69	1	16
4096	258	139	3957	20	140	9	69	2	17
4096	259	140	3956	21	141	10	70	2	17
4096	260	141	3955	22	142	10	70	2	17
4096	261	142	3954	23	143	11	71	2	17
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
4096	4090	3971	125	3852	3972	1925	1985	1865	1985
4096	4091	3972	124	3853	3973	1926	1986	1866	1986
4096	4092	3973	123	3854	3974	1926	1986	1866	1986
4096	4093	3974	122	3855	3975	1927	1987	1867	1987
4096	4094	3975	121	3856	3976	1927	1987	1867	1987
4096	4095	3976	120	3857	3977	1928	1988	1868	1988

Table 31. Code parameters for the Hermitian curve over $\mathbb{F}_{256}$.

CHAPTER 8

Further Directions

This paper has described bivariate Goppa codes with one-point divisors over smooth curves, along with a simple decoding algorithm, while using a minimum amount of graduate abstract algebra. See [HvLP] for a more thorough mathematical treatment; see [Pre], part II, for a function-field approach. The restriction to one-point divisors appears not to be seen as a liability; as shown in section 4.14, they ease the otherwise difficult computation of $\ell(D)$. See [Pre], chapter 15, for more information on non-smooth curves and the difficulty of computing their genuses. The SV algorithm does not decode up to half the minimum distance; see [Pre], [HvLP] for the full error-processing algorithm. Good codes in the sense of definition 3.6.4 are not obtainable using the technique presented in this paper; see [GS] and [TVZ], and see [SAKSD] for an efficient method to construct such codes. The computation of the genuses of the curves in [GS] is an open problem, as is the development of more efficient decoding algorithms.

REFERENCES

[Ber] E. Berlekamp. Algebraic Coding Theory (revised 1984 edition). Aegean Park Press, 1984.

[DF] D.S. Dummit and R.M. Foote. Abstract Algebra (2nd ed.). John Wiley and Sons, 1999.

[HvLP] Høholdt, T., van Lint, J.H. and Pellikaan, R. *Algebraic Geometry Codes.* Handbook of Coding Theory, vol. 1, pp. 871-961 (Pless, V.S., Huffman, W.C. and Brualdi, R.A. Eds.). Elsevier, Amsterdam, 1998.

[GS] Garcia, A. and Stichtenoth, H. (1996). *Asymptotically good towers of function fields over finite fields.* C. R. Acad. Sci. Paris **322 I**, 1067-1070.

[Gop] Goppa, V.D. (1977). *Codes associated with divisors.* Probl. Inform. Transmission, vol. 13, 22-26.

[Har] Hartshorne, R. Algebraic Geometry. Springer, 1977.

[Iha] Ihara, Y. (1983). *Some remarks on the number of rational points of algebraic curves over finite fields.* J. Fac. Sci. Tokyo **28**, pp. 721-724.

[Lau] Lauter, K. (2001). *Geometric Methods for Improving the Upper Bounds on the Number of Rational Points on Algebraic Curves over Finite Fields.* Journal of Algebraic Geometry **10**, pp. 19-36.

[LN] R. Lidl and H. Niederreiter. Finite Fields. Cambridge University Press, 1997.

[MS] MacWilliams, F.J. and Sloane, N.J.A. The Theory of Error-Correcting Codes. Elsevier Science B.V., 1997.

[PW] Peterson, W.W. and Weldon, E.J. Error-Correcting Codes (2nd ed.). MIT Press, 1972.

[Pre] Pretzel, O. Codes and Algebraic Curves. Oxford University Press, 1998.

[Sch] Schenk, H. Computational Algebraic Geometry. Cambridge University Press, 2003.

[Ser] Serre, J.-P. (1983). *Sur le nombre des points rationelles d'une courbe algébrique sur un corps fini.* C.R. Acad. Sc. Paris Sér. I Math. **296**, pp. 397-402.

[SAKSD] Shum, K., Aleshnikov, I., Kumar, P.V., Stichtenoth, H., and Deolalikar, V. *A Low-Complexity Algorithm for the Construction of Algebraic-Geometric Codes Better Than the Gilbert-Varshamov Bound.* IEEE Trans. Inform. Theory, vol. 47, no. 6, 2225-2241, Sep. 2001.

[Sil] Silverman, J. The Arithmetic of Elliptic Curves. Springer-Verlag, 1986.

[ST] Silverman, J. and Tate, J. Rational Points on Elliptic Curves. Springer-Verlag, 1992.

[SV] Skorobogatov, A.N. and Vlăduţ, S.G. *On the decoding of algebraic-geometric codes.* IEEE Trans. Inform. Theory, vol. 36, pp. 1051-1060, Nov. 1990.

[Sud] Sudan, M. *Algebraic Introduction to Coding Theory.* `http://theory.lcs.mit.edu/~madhu/FT01`.

[TVZ] Tsfasman, M.A., Vlăduţ, S.G., and Zink, T. (1982). *Modular curves, Shimura curves and Goppa codes better than the Varshamov-Gilbert bound.* Math. Nachr., **109**, 21-28.

[VvO] Vanstone, S.A. and van Oorschot, P.C. An Introduction to Error Correcting Codes with Applications. Kluwer Academic Publishers, 1989.

[Wal] Walker, J.L. Codes and Curves. American Mathematical Society, 2000.

Index