Chapter 11

Algebraic Geometry Constructions of Convolutional Codes

J.A. Domínguez Pérez, J.M. Muñoz Porras and G. Serrano Sotelo

Department of Mathematics
University of Salamanca
Plaza de la Merced 1-4, 37008 Salamanca, Spain
{jadoming,jmp,laina}@usal.es

Algebraic-geometric techniques to construct linear codes can be applied to construct convolutional codes, using algebraic curves over function fields. In this way we construct convolutional Goppa codes and provide a systematic way for constructing convolutional codes with prescribed properties. We study convolutional Goppa codes defined by the projective line and elliptic curves in detail.

Contents

11.1. Introduction

The notion of convolutional codes was introduced by Peter Elias [1] in 1955, considering the codification as a time-dependent process: the *codified word* at some instant depends not only on the *information word* at that instant, but also on the previous words; the number of the previous words

on which the codifications depends is called the *memory* of the codifier; in this scheme the codification of a word can be interpreted as a certain *convolution* with other words.

The use of convolutional codes was very important after the discovery by Andrew Viterbi [2] in 1967 of the decodification algorithm known by his name. The compatibility of this algorithm depends exponentially on the memory of the codifier.

One of the main applications of convolutional codes is the transmission of information through the *deep-space*, where there are strong limitations of potency, but in general no restrictions to the wide band. The communication systems used in artificial satellites transmit telemetric information: orders from earth stations to satellites and tracking. In the telemetric channel, where the rate of the code is relatively high, convolutional and block codes are used.

For instance, the space missions *Pioneer* 10 and 11 to Jupiter and Saturn in 1972-73 used a convolutional code of rate 1/2 and memory 31. After Viterbi, a *planetary standard* as a convolutional code of rate 1/2 and memory 6 was implemented. This code was used for the first time in the *Voyager 1* mission (1980-81), concatenated with a Reed-Solomon code, and in the *Galileo* (1986) and *Voyager 2* missions (1989), concatenated with other convolutional and block codes.

Convolutional codes are also used in the construction of *turbo-codes*, introduced in 1983 by Berrou, Glavieux and Thitimajshima [3]. These are the codes currently used in wireless communications.

There are different approaches to the study of convolutional codes: they can be studied as sequential circuits, as discrete linear systems, etc. From an algebraic point of view, the fundamental reference is the work of G. David Forney Jr. [4] in 1970. Then came the works of Robert J. McEliece [5] in 1977 and Philippe Piret in 1988 [6]. More recently, McEliece [7] in 1998 again gave an introduction to the algebraic theory of convolutional codes, which clarifies the previous approaches.

The recent work of Joachim Rosenthal, Roxana Smarandache [8] and Heide Gluesing-Luerssen [9] in 1999 and 2001, Vakhtang Lomadze [10] in 2001, and the authors of this chapter [11] in 2004, has shown that the use of techniques of Algebraic Geometry are very useful in the study of convolutional codes.

Our contribution continues with five sections. In §2 we give an introduction to the general theory of convolutional codes. Section §3 is devoted to constructing convolutional Goppa codes in terms of algebraic curves de-

fined over the field of rational functions in one variable over a finite field (see [12]).

In §4 we analyze the notion of weight and free distance for convolutional codes, and their possible geometric interpretation in the case of convolutional Goppa codes.

Finally, §5 and §6 are devoted to studying some explicit examples of convolutional Goppa codes defined by the projective line and elliptic curves.

11.2. Convolutional Codes

Given a finite field $\mathbb{F}_q$, representing the symbols in which an information word $u \in \mathbb{F}_q^k$ is written, a *linear block encoder* is essentially an injective linear map

$$\begin{aligned} G\colon \mathbb{F}_q^k &\hookrightarrow \mathbb{F}_q^n \\ u &\mapsto x = u \cdot G \end{aligned} \quad ,$$

whose image subspace is the *linear code* $\mathcal{C}_k = \operatorname{Im} G \subset \mathbb{F}_q^n$.

The map G is a $k \times n$ matrix with entries in $\mathbb{F}_q$, which is called a *generator matrix* of the code (their rows are a basis of $\mathcal{C}_k$); k is the *dimension* of the code, and n is its *length*. Alternatively, one can define the code $\mathcal{C}_k$ by its implicit equations

$$\mathcal{C}_k = \{x \in \mathbb{F}_q^n \,/\, H \cdot x = 0\}\,,$$

where H is an $(n-k) \times n$ matrix with entries in $\mathbb{F}_q$, called a *parity-check (control) matrix* of the code.

In practical applications, the codification process is not limited to a single word, but a sequence of information words depending on time $u(t) \in \mathbb{F}_q^k$, $t = 0, 1, 2, \ldots$, which after the codification are transformed in the sequence of codified words

$$x(t) = u(t) \cdot G \in \mathcal{C}_k \in \mathbb{F}_q^n\,, \quad t = 0, 1, 2, \ldots$$

The codified word $x(t)$ at the instant t depends only on the information word $u(t)$ at the same instant t.

The basic idea of convolutional codification is to allow $x(t)$ to depend not only on $u(t)$ but also on $u(t-1), \ldots, u(t-m)$ for some positive integer m, which is the *memory* of the code. One can then consider a linear block code as a convolutional code with zero memory.

Let us explain this with greater precision. One can write a sequence of words as a polynomial in one variable z whose coefficients are the sequence,

$$U(z) = \sum_t u(t) z^t \in \mathbb{F}_q[z]^k \,,$$

and the product by z^i can be considered as a *delay operator*

$$z^i U(z) = \sum_t u(t) z^{t+i} = \sum_t u(t-i) z^t \in \mathbb{F}_q(z)^k \,.$$

One can now define a convolutional (polynomial) encoder as an injective homomorphism of $\mathbb{F}_q[z]$-modules

$$\begin{array}{c} G(z) \colon \mathbb{F}_q[z]^k \hookrightarrow \mathbb{F}_q[z]^n \\ U(z) \mapsto X(z) = U(z) \cdot G(z) \end{array} \,,$$

where $G(z)$ is a $k \times n$ matrix with entries in $\mathbb{F}_q[z]$.

If we allow the possibility of performing *feedback*, this means that we can reverse the delay, and define convolutional codification more generally over $\mathbb{F}_q(z)$, the field of fractions of $\mathbb{F}_q[z]$, i.e., the localization of the ring $\mathbb{F}_q[z]$ with respect to the multiplicative system $S = \{Q(z) \in \mathbb{F}_q[z] / Q(z) \neq 0\}$. Thus, a convolutional encoder is an injective $\mathbb{F}_q(z)$-linear map

$$G(z) \colon \mathbb{F}_q(z)^k \hookrightarrow \mathbb{F}_q(z)^n \,,$$

with the entries of $G(z)$ also in $\mathbb{F}_q(z)$.

Definition 11.1. An (n, k) *convolutional code* $\mathcal{C}_k$ over $\mathbb{F}_q$ is a linear subspace of dimension k over $\mathbb{F}_q(z)$ of $\mathbb{F}_q(z)^n$.

The integers (n, k) are called, respectively, the *length* and *dimension* of the convolutional code, and n/k is the *ratio* of $\mathcal{C}_k \subseteq \mathbb{F}_q(z)^n$.

Example 11.1. The subspace of dimension 1 in $\mathbb{F}_2(z)^3$ defined by

$$\mathcal{C}_1 = \langle 1 + z, 1 + z^2, z + z^3 \rangle \,,$$

is a $(3, 1)$-convolutional code over $\mathbb{F}_2$, in which the code word $x(t)$ at the instant t is obtained in terms of the information words $u(t)$, $u(t-1)$, $u(t-2)$ and $u(t-3)$ in the following way:

$$x(t) = (u(t) + u(t-1), u(t) + u(t-2), u(t-1) + u(t-3)) \,.$$

11.2.1. *Convolutional encoders. Linear systems and circuits.*

Definition 11.2. A *convolutional encoder*, or *codifier*, for a convolutional code $\mathcal{C}_k \subseteq \mathbb{F}_q(z)^n$ is a $k \times n$ matrix

$$G(z) = \begin{pmatrix} G_1(z) \\ \vdots \\ G_k(z) \end{pmatrix},$$

where $G_i(z) = (G_{i1}(z), \ldots, G_{in}(z)) \in F(z)^n$ are a basis of $\mathcal{C}_k$.

Equivalently, $G(z)$ is a *generator matrix* that defines an injective linear *encoding map*:

$$\begin{aligned} G(z)\colon F(z)^k &\hookrightarrow F(z)^n \\ U(z) &\mapsto X(z) = U(z) \cdot G(z)\,, \end{aligned}$$

such that $\operatorname{Im} G(z) = \mathcal{C}_k \subset F(z)^n$.

Given two codifiers $G(z)$ and $G'(z)$ of the same convolutional code $\mathcal{C}_k$, there exists an element (the *base change*) $B(z) \in GL(k, \mathbb{F}_q(z))$ of the linear group of dimension k over $\mathbb{F}_q(z)$ such that:

$$G'(z) = B(z) \cdot G(z)\,.$$

Example 11.2. The convolutional code $\mathcal{C}_1$ defined in example 11.1 has the following different encoders

$$\begin{aligned} G(z) &= (\ 1+z, 1+z^2,\ z+z^3\ ) \\ G'(z) &= (\ z,\ z+z^2, z^2+z^3\ ) \\ G''(z) &= (\ 1,\ 1+z,\ z+z^2\ ) \\ G'''(z) &= (\ \tfrac{1}{1+z},\ 1,\ z\ ) \end{aligned}$$

and the following identities are satisfied

$$G(z) = \frac{1+z}{z} \cdot G'(z) = (1+z) \cdot G''(z) = (1+z^2) \cdot G'''(z)\,.$$

Definition 11.3. A *polynomial encoder* is a convolutional encoder $G(z)$ whose entries are in $\mathbb{F}_q[z]$.

If $G_i(z)$ is the i-th row of a polynomial encoder $G(z)$, the *degree* of $G_i(z)$, denoted $e_i = \operatorname{Degree} G_i(z)$, is the highest of the degrees of its components

$$e_i = \max_{1 \le j \le n} (\operatorname{Degree} G_{ij}(z))\,.$$

The *memory* of a polynomial encoder $G(z)$, denoted m_G, is the maximun degree of its rows

$$m_G = \max_{1\leq i\leq k} e_i\,.$$

The *degree* of a polynomial encoder $G(z)$, denoted δ_G, is the sum of the degrees of its rows

$$\delta_G = \sum_{i=1}^{k} e_i\,.$$

Remark 11.1. By reordering the rows of $G(z)$, one can henceforth assume that

$$e_1 \leq \cdots \leq e_k = m_G\,.$$

Every convolutional code has polynomial encoders: if $G(z)$ is an arbitrary encoder and $\mu(z)$ the least common multiple of the denominators of its coefficients, $\mu(z)\,G(z)$ is a polynomial encoder of the same code. Thus, for any encoder $G(z)$ one can consider the *degree* δ_G as the degree of the polynomial encoder $\mu(z)\,G(z)$. In particular, *linear block codes* are convolutional codes for which there are *zero degree* encoders.

Definition 11.4. A convolutional encoder $G(z)$ is called *realizable* if the denominators of its entries are not multiples of z.

The notion of realizable encoder is related to the possibility of describing the encoder as a *linear system* and therefore the possibility of constructing a *physical circuit* which performs the encoding process. Given a $k \times n$ realizable encoder $G(z)$ of degree δ_G, it can be decomposed as

$$G(z) = D + E(z)\cdot C\,,$$

where $D = G(0)$ is a $k \times n$ matrix with entries in $\mathbb{F}_q$; C is a $\delta_G \times n$ matrix with entries in $\mathbb{F}_q$, and $E(z)$ is a $k \times \delta_G$ matrix defining the *morphism of states* with values in the *space of state-variables* $\mathbb{F}_q[z]^{\delta_G}$

$$\begin{aligned} E(z)\colon \mathbb{F}_q[z]^k &\to \mathbb{F}_q[z]^{\delta_G}\\ U(z) &\mapsto S(z) = U(z)\cdot E(z)\,. \end{aligned}$$

With these notations, the encoding morphism can be expressed in terms of the *state-variables* $S(z)$ as:

$$\begin{aligned} X(z) =& U(z)\cdot G(z) = S(z)\cdot C + U(z)\cdot D\,.\\ z^{-1}S(z) =& U(z)\cdot z^{-1}E(z)\,. \end{aligned}$$

Denoting $B = (z^{-1}E(z))|_{z=0}$ and decomposing $z^{-1}E(z) = B + E(z) \cdot A$, where A is a $\delta_G \times \delta_G$ matrix with entries in $\mathbb{F}_q$, one obtains the following identity:

$$z^{-1}S(z) = U(z) \cdot (E(z) \cdot A + B) = S(z) \cdot A + U(z) \cdot B\,.$$

Thus, a convolutional realizable encoder is equivalent to a *time-invariant linear system with a finite number of state variables*. The *state space description* of the encoder is given by

$$\left.\begin{aligned} s(t+1) =& s(t) \cdot A_{\delta_G \times \delta_G} + u(t) \cdot B_{k \times \delta_G} \\ x(t) =& s(t) \cdot C_{\delta_G \times n} + u(t) \cdot D_{k \times n} \end{aligned}\right\}. \tag{11.1}$$

By solving the states in the first equation, one obtains $x(t)$ as a function of a set $u(t), u(t-1), \ldots$. If this set is infinite, we say that the encoder has *infinite memory*, whereas if it is finite $u(t), u(t-1), \ldots, u(t-m_G)$, we call m_G the *memory* of the realizable encoder $G(z)$.

Example 11.3. The encoder $G'''(z) = (\frac{1}{1+z}, 1, z)$ of example 11.2 is realizable and its degree is $\delta_{G'''} = 2$. Applying the above description, one obtains:

$$\begin{aligned} D =& G'''(z)|_{z=0} = (1,1,0) \\ G'''(z) - D =& \left(\frac{z}{1+z}, 0, z\right) = \frac{1}{1+z}\left(z, 0, z+z^2\right) = \\ =& \underbrace{\frac{1}{1+z}\left(z, z^2\right)}_{E(z)} \cdot \underbrace{\begin{pmatrix} 1 & 0 & 1 \\ 0 & 0 & 1 \end{pmatrix}}_{C} \\ B =& (z^{-1}E(z))|_{z=0} = (1,0) \\ z^{-1}E(z) - B =& \frac{1}{1+z}(z,z) = \underbrace{\frac{1}{1+z}(z, z^2)}_{E(z)} \cdot \underbrace{\begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix}}_{A} \end{aligned}$$

and finally the state equations are:

$$\left.\begin{aligned} (s_1(t+1), s_2(t+1)) =& (s_1(t), s_2(t)) \cdot \begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix} + u_1(t) \cdot (1,0)\,, \\ (x_1(t), x_2(t), x_3(t)) =& (s_1(t), s_2(t)) \cdot \begin{pmatrix} 1 & 0 & 1 \\ 0 & 0 & 1 \end{pmatrix} + u_1(t) \cdot (1,1,0)\,. \end{aligned}\right\}.$$

Remark 11.2. Let us consider the equations of the encoder of the above example 11.3

$$\left.\begin{aligned}(s_1(t+1), s_2(t+1)) =& (u_1(t) + s_1(t), s_1(t))\,,\\ (x_1(t), x_2(t), x_3(t)) =& (u_1(t) + s_1(t), u_1(t), s_1(t) + s_2(t))\,.\end{aligned}\right\}$$

The state s_1 depends recurrently on itself; that is, there is *feed-back*, and by substituting the first equation in the second one obtains

$$\begin{aligned}x_1(t) = u_1(t) + s_1(t) =& u_1(t) + u_1(t-1) + s_1(t-1) =\\ =& u_1(t) + u_1(t-1) + u_1(t-2) + s_1(t-2) =\\ =& u_1(t) + u_1(t-1) + u_1(t-2) + u_1(t-3) + \dots\end{aligned}$$

Accordingly, the code word depends indefinitely on the information word, which means that the encoder has infinite memory, although its degree is finite.

The generator matrix of an encoder can be recovered from its expression as a linear system: writing equations (11.1) as

$$\left.\begin{aligned}z^{-1}S(z) =& S(z)\cdot A + U(z)\cdot B\\ X(z) =& S(z)\cdot C + U(z)\cdot D\end{aligned}\right\}\,,$$

and eliminating the state variables, one concludes:

$$G(z) = B\cdot(z^{-1}\,\mathrm{Id}_{\delta_G} - A)^{-1}\cdot C + D\,.$$

Example 11.4. Let us consider the matrices of the linear system defined in example 11.3:

$$A = \begin{pmatrix}1 & 1\\ 0 & 0\end{pmatrix}\,,\quad B = \begin{pmatrix}1 & 0\end{pmatrix}\,,\quad C = \begin{pmatrix}1 & 0 & 1\\ 0 & 0 & 1\end{pmatrix}\,,\quad D = \begin{pmatrix}1 & 1 & 0\end{pmatrix}\,.$$

One has $\left(z^{-1}\,\mathrm{Id}_2 - A\right)^{-1} = \frac{1}{1+z}\begin{pmatrix}z & z^2\\ 0 & z+z^2\end{pmatrix}$, and hence:

$$\underbrace{\frac{1}{1+z}\begin{pmatrix}1 & 0\end{pmatrix}\cdot\begin{pmatrix}z & z^2\\ 0 & z+z^2\end{pmatrix}}_{E(z)}\cdot\begin{pmatrix}1 & 0 & 1\\ 0 & 0 & 1\end{pmatrix} + \begin{pmatrix}1 & 1 & 0\end{pmatrix} = \left(\tfrac{1}{1+z}, 1, z\right) = G'''(z)\,.$$

The description of realizable encoders as linear systems allows us to express them as physical devices called *sequential circuits*, composed of *memory boxes* for delay operations (as many as the memory of the encoder) and *sum boxes* for addition operations.

Example 11.5. Let us consider the encoder of example 11.3. Its expression as linear systems is given by the equations:

$$\left.\begin{aligned} s_1(t+1) &= u_1(t) + s_1(t) \\ s_2(t+1) &= s_1(t) \end{aligned}\right\}, \qquad \left.\begin{aligned} x_1(t) &= u_1(t) + s_1(t) \\ x_2(t) &= u_1(t) \\ x_3(t) &= s_1(t) + s_2(t) \end{aligned}\right\},$$

which correspond to the circuit:

11.2.2. *Basic encoders. Degree of a convolutional code*

Let $\mathcal{C}_k \subseteq \mathbb{F}_q(z)^n$ be a (n,k)-convolutional code. Any polynomial encoder $G(z)$ for $\mathcal{C}_k$ induces a morphism of $\mathbb{F}_q[z]$-modules:

$$\phi_G \colon \mathbb{F}_q[z]^k \hookrightarrow \mathbb{F}_q[z]^n \,,$$

whose localization with respect to the multiplicative system $S = \{Q(z) \in \mathbb{F}_q[z] / Q(z) \neq 0\}$ is the encoding map:

$$G(z) = (\phi_G)_S \colon \mathbb{F}_q(z)^k \hookrightarrow \mathbb{F}_q(z)^n \,.$$

The existence of a *decoding map* $G^{-1}(z) \colon \mathbb{F}_q(z)^n \to \mathbb{F}_q(z)^k$ retract of $G(z)$ (i.e. $G(z) \cdot G^{-1}(z) = \mathrm{Id}_k$) can be deduced from the exact sequence:

$$0 \to \mathbb{F}_q[z]^k \xrightarrow{\phi_G} \mathbb{F}_q[z]^n \to \mathbb{F}_q[z]^n / \operatorname{Im} \phi_G \to 0 \,.$$

For each multiplicative system $S \subset \mathbb{F}_q[z]$, one has the exact sequence:

$$0 \to \mathbb{F}_q[z]_S^k \xrightarrow{(\phi_G)_S} \mathbb{F}_q[z]_S^n \to (\mathbb{F}_q[z]^n / \operatorname{Im} \phi_G)_S \to 0 \,.$$

The existence of a retract of $(\phi_G)_S$ is equivalent to saying that $(\mathbb{F}_q[z]^n/\operatorname{Im}\phi_G)_S$ is a free module isomorphic to $\mathbb{F}_q[z]_S^{n-k}$. Let us consider the decomposition of $\mathbb{F}_q[z]^n/\operatorname{Im}\phi_G$ as:

$$\mathbb{F}_q[z]^n/\operatorname{Im}\phi_G \simeq \mathbb{F}_q[z]^{n-k} \oplus T\,,$$

where T is the torsion submodule of $\mathbb{F}_q[z]^n/\operatorname{Im}\phi_G$. Then, $(\phi_G)_S$ is a retract if and only if $T_S = 0$. On the other hand,

$$T \simeq \mathbb{F}_q[z]/\langle\gamma_i(z)\rangle \oplus \cdots \oplus \mathbb{F}_q[z]/\langle\gamma_k(z)\rangle\,,$$

where $\gamma_i(z) \in \mathbb{F}_q[z]$ are the *invariant factors* of ϕ_G,

$$\gamma_i(z) = \Delta_i(z)/\Delta_{i-1}(z)\,,$$

where $\Delta_i(z)$ is the highest common divisor of the minors of order i of $G(z)$. Thus, the vanishing of T_S is equivalent to the condition of $\Delta_k(z)$ being invertible in $\mathbb{F}_q[z]_S$.

Definition 11.5. A polynomial encoder $G(z)$ is *non-catastrophic* if any of the following equivalent conditions are satisfied:

(1) $G(z)$ has a right inverse in $\mathbb{F}_q[z]_S$, where $S = \{z^l, l \geq 0\}$.
(2) $\Delta_k(z) = z^l$, $l \geq 0$.

The word *catastrophic* is used by Massey and Sain [13] to refer to encoders in which a code word $X(z)$ of finite length can be obtained from an information word $U(z)$ of infinite length, and hence the code word may contain infinite errors (*catastrophic errors*).

Example 11.6. Let us consider the encoder $G'(z) = (z, z+z^2, z^2+z^3)$ of example 11.2. One has that $\Delta_1(z) = z$, and therefore $G'(z)$ is a non-catastrophic encoder, which has a (non-unique) right inverse, such as for instance:

$$(G')^{-1}(z) = \begin{pmatrix} \frac{1}{z} + P(z)(1+z) + Q(z)(1+z^2) \\ P(z) \\ Q(z) \end{pmatrix},$$

where $P(z), Q(z)$ are arbitrary polynomials in $\mathbb{F}_q[z]$.

Definition 11.6. A polynomial encoder $G(z)$ is *basic* if any of the following equivalent conditions are satisfied:

(1) $G(z)$ has a right inverse in $\mathbb{F}_q[z]$.
(2) $\Delta_k(z) = 1$.

(3) $\gamma_1(z) = \cdots = \gamma_k(z) = 1$.

In particular, any basic encoder is non-catastrophic.

Example 11.7. The encoder $G''(z) = (1, 1+z, z+z^2)$ of example 11.2 satisfies the condition $\gamma_1(z) = 1$, and is therefore basic. A right inverse of $G''(z)$ is:

$$G^{-1}(z) = \begin{pmatrix} 1 + P(z)(1+z) + Q(z)(1+z^2) \\ P(z) \\ Q(z) \end{pmatrix} .$$

The existence of basic encoders for all convolutional codes was proved in a constructive way by Forney [4], using the Smith algorithm for the computation of invariant factors.

Theorem 11.1. *All convolutional codes admit basic encoders.*

Proof. Let $G(z)$ be a polynomial encoder for an (n,k)-convolutional code. The Smith algorithm allows us to compute the invariant factors $\gamma_1(z), \ldots, \gamma_k(z)$ of ϕ_G and two unimodular matrices $B(z) \in GL(k, \mathbb{F}_q[z])$, $C(z) \in GL(n, \mathbb{F}_q[z])$ such that:

$$B(z) \cdot G(z) \cdot C(z) = \big(\Gamma(z) \mid 0_{k\times(n-k)}\big) ,$$

where

$$\Gamma(z) = \begin{pmatrix} \gamma_1(z) & & \\ & \ddots & \\ & & \gamma_k(z) \end{pmatrix} .$$

From the above identity, one obtains:

$$\Gamma(z)^{-1} \cdot B(z) \cdot G(z) = (\mathrm{Id}_k \,|0) \cdot C(z)^{-1} ,$$

which has invariant factors equal to 1 and is therefore a basic encoder whose right inverse is the polynomial matrix determined by the first k columns of $C(z)$. □

Example 11.8. If we apply the above algorithm to the polynomial encoder $G(z) = (1+z, 1+z^2, z+z^3)$ introduced in example 11.1, we obtain:

$$\Gamma(z) = (1+z) \quad B(z) = (1) , \quad C(z) = \begin{pmatrix} 1 & 1+z & z+z^2 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} ,$$

and the corresponding basic encoder is

$$\Gamma(z)^{-1} \cdot B(z) \cdot G(z) = \frac{1}{1+z} \cdot (1) \cdot (1+z, 1+z^2, z+z^3) = (1, 1+z, z+z^2)\,.$$

The basic encoders are a *maximal class* in the set of polynomials encoders, in the following sense:

Theorem 11.2. *Let $G(z)$ and $G'(z)$ be two polynomial encoders of an (n,k) convolutional code. One has:*

(1) If $G(z)$ is basic and $\operatorname{Im}\phi_G \subseteq \operatorname{Im}\phi_{G'}$, then $\operatorname{Im}\phi_G = \operatorname{Im}\phi_{G'}$.
(2) If $G(z)$ and $G'(z)$ are basic, then $\operatorname{Im}\phi_G = \operatorname{Im}\phi_{G'}$.

Proof. (1) One has a conmutative diagram:

$$\begin{array}{ccccccccc}
0 & \longrightarrow & \mathbb{F}_q[z]^k & \xrightarrow{\phi_G}_{\sim} & \operatorname{Im}\phi_G & \hookrightarrow & \mathbb{F}_q[z]^n & \longrightarrow & \mathbb{F}_q[z]^n/\operatorname{Im}\phi_G \longrightarrow 0 \\
 & & \| & & \downarrow f & & \| & & \downarrow h \\
0 & \longrightarrow & \mathbb{F}_q[z]^k & \xrightarrow{\phi_{G'}}_{\sim} & \operatorname{Im}\phi_{G'} & \hookrightarrow & \mathbb{F}_q[z]^n & \longrightarrow & \mathbb{F}_q[z]^n/\operatorname{Im}\phi_{G'} \longrightarrow 0
\end{array}$$

and $\ker h = \operatorname{Coker} f$. Since f is injective, one has that $\operatorname{Coker} f = \operatorname{Im}\phi_{G'}/\operatorname{Im}\phi_G$, and $\mathbb{F}_q[z]^n/\operatorname{Im}\phi_G$ is free, since $G(z)$ is basic. Thus, $\ker h$ is torsion-free and hence $\operatorname{Im}\phi_{G'}/\operatorname{Im}\phi_G = 0$.

(2) The submodule $\operatorname{Im}\phi_G + \operatorname{Im}\phi_{G'}$ generates the convolutional code and contains $\operatorname{Im}\phi_G$ and $\operatorname{Im}\phi_{G'}$. Thus, applying (1), one has $\operatorname{Im}\phi_G = \operatorname{Im}\phi_G + \operatorname{Im}\phi_{G'} = \operatorname{Im}\phi_{G'}$. □

Remark 11.3. This theorem implies that basic encoders are *invariant with respect the action of the unimodular group* $GL(k, \mathbb{F}_q[z]) = \operatorname{Aut}_{\mathbb{F}_q[z]} \mathbb{F}_q[z]^k$.

Given two basic encoders $G(z)$ and $G'(z)$ of an (n,k) convolutional code, there exists a $B(z) \in GL(k, \mathbb{F}_q[z])$ such that:

$$G'(z) = B(z) \cdot G(z)\,.$$

Corollary 11.1. *Let $G(z)$ and $G'(z)$ be two polynomial encoders of an (n,k) convolutional code, and let us assume that $G(z)$ is basic. If one denotes*

$$\bar{\delta}_G = \textit{maximum degree of the minors of order } k \textit{ of } G(z)\,,$$

then:

$$\bar{\delta}_G \leq \bar{\delta}_{G'}\,.$$

In particular, if $G'(z)$ is also basic, then $\bar{\delta}_G = \bar{\delta}_{G'}$.

Proof. $G(z)$ and $G'(z)$ define the same convolutional code, and hence there exists a $B(z) \in GL(k, \mathbb{F}_q(z))$ such that $G'(z) = B(z) \cdot G(z)$. Since $G(z)$ is basic, from theorem 11.2 $\operatorname{Im} \phi_{G'} \subseteq \operatorname{Im} \phi_G$,

$$\begin{array}{ccc} \mathbb{F}_q[z]^k & \xrightarrow{\phi_G} & \operatorname{Im} \phi_G \\ {\scriptstyle B(z)}\uparrow & & \uparrow \\ \mathbb{F}_q[z]^k & \xrightarrow{\phi_{G'}} & \operatorname{Im} \phi_{G'} \end{array},$$

and one deduces that $B(z) \in GL(k, \mathbb{F}_q[z])$.

Because the minors of $G'(z)$ are the minors of $G(z)$ multiplied by the determinant of $B(z)$, one concludes that $\bar{\delta}_{G'} \geq \bar{\delta}_G$. If $G'(z)$ is also basic, one deduces the equality. □

In the sense of corollary 11.1, the degree of any basic encoder is an *invariant* of the convolutional code.

Definition 11.7. The *degree* δ *of a convolutional code* $\mathcal{C}_k \subseteq \mathbb{F}_q(z)^n$ is

$$\delta = \bar{\delta}_G \,,$$

where $G(z)$ is any basic encoder of $\mathcal{C}_k$.

Sometimes the degree $\bar{\delta}_G$ of a polynomial encoder is also called *internal degree* (see McEliece [7]) to distinguish it from the so-called *external degree*, used to refer to the degree δ_G (see definition 11.3).

11.2.3. *Minimal basic encoders. Canonical encoders*

In the implementation of convolutional codes as physical devices it is convenient to find *minimal encoders*, in the sense that the corresponding circuit had a minimum quantity of memory boxes. The formalization of the concept of minimality can be expressed in terms of the degree δ of the code.

Theorem 11.3. *(Forney [4]) For each* (n, k)*-convolutional code of degree* δ *there exists at least one basic encoder* $G(z)$ *such that*

$$\delta = \delta_G \leq \delta_{G'} \,,$$

for all realizable encoders $G'(z)$ *of the convolutional code. These basic encoders* $G(z)$ *are called* minimal basic encoders*.*

Definition 11.8. (McEliece [7]) Given an (n,k)-convolutional code $\mathcal{C}_k \subseteq \mathbb{F}_q(z)^n$, a polynomial encoder $G(z)$ is called

- *canonical*, if $\delta_G \leq \delta_{G'}$, for every polynomial encoder $G'(z)$ of $\mathcal{C}_k$.
- *reduced*, if $\delta_G = \bar{\delta}_G$.

Theorem 11.4. *(McEliece [7])A polynomial encoder is canonical if and only if it is basic and reduced. Moreover, the set of row degrees is the same for every canonical encoder,*

$$e_1 \leq \cdots \leq e_k = m_G\,.$$

These invariants *of the convolutional code are called the* Forney indices. *The maximum degree $e_k = m_G$ is called the* memory *of the convolutional code.*

Remark 11.4. A polynomial encoder $G(z) = (G_{ij}(z))$ is reduced if and only if the matrix $\bar{G} = (\bar{g}_{ij})$, $\bar{g}_{ij} \in \mathbb{F}_q$, defined by the coefficients of the terms of highest degree in each row, has rank k (McEliece [7]).

Thus, one has the following method for constructing reduced encoders: if the matrix $\bar{G}$ does not have rank k, there exists a zero linear combination between its rows

$$\sum_{i=1}^{k} \lambda_i \bar{g}_{ij} = 0\,, \quad \text{with } \lambda_i \in \mathbb{F}_q\,, \quad 1 \leq j \leq n\,,$$

from which one can construct a linear combination between the rows of $G(z)$ by eliminating the terms of highest degree,

$$\sum_{i=1}^{k} \lambda_i z^{e_k - e_i} G_{ij}(z) = 0\,,$$

and this allows us to replace a row of $G(z)$ by a new one to obtain a new encoder with the lowest degree in each row. Applying this process several times, we finally obtain a reduced encoder.

Example 11.9. For the encoders of example 11.2 we have:

$G(z)$	Δ_k	Basic	δ_G	$\bar{\delta}_G$	Reduced
$(1+z, 1+z^2, z+z^3)$	$1+z$	No	3	3	Yes
$(z, z+z^2, z^2+z^3)$	z	No	3	3	Yes
$(1, 1+z, z+z^2)$	1	Yes	2	2	Yes

In particular, one deduces that $\mathcal{C}_1 \subset \mathbb{F}_2(z)^3$ is a convolutional code with memory equal to its degree, $\delta = 2$.

11.2.4. *Dual code. Parity-check (control) matrix*

Let us consider, over the $\mathbb{F}_q(z)$-vectorial space $\mathbb{F}_q(z)^n$, the pairing $\langle\ ,\ \rangle$

$$\begin{aligned} &\mathbb{F}_q(z)^n \times \mathbb{F}_q(z)^n \to \mathbb{F}_q(z) \\ &(X(z), Y(z)) \mapsto \langle X(z), Y(z)\rangle = \sum_{i=1}^{n} X_i(z) Y_i(z)\,, \end{aligned} \tag{11.2}$$

where $X(z) = (X_1(z), \ldots, X_n(z)), Y(z) = (X_1(z), \ldots, X_n(z)) \in \mathbb{F}_q(z)^n$.

Definition 11.9. Given an (n,k)-convolutional code $\mathcal{C}_k \subseteq \mathbb{F}_q(z)^n$, the dual code is the $\mathbb{F}_q(z)$-subspace defined by

$$\mathcal{C}_k^{\perp} = \{Y(z) \in \mathbb{F}_q(z)^n \ / \ \langle X(z), Y(z)\rangle = 0 \text{ for every } X(z) \in \mathcal{C}\}\,.$$

Theorem 11.5. *$\mathcal{C}_k^{\perp}$ is an $(n-k,k)$ convolutional code of degree equal to the degree of $\mathcal{C}_k$.*

Proof. Let $G(z)$ be a basic encoder $\mathcal{C}_k$. Since $G(z)$ is basic, then $\mathbb{F}_q[z]^n / \operatorname{Im}\phi_G \simeq \mathbb{F}_q[z]^{n-k}$ is free, and one has an exact sequence

$$0 \to \mathbb{F}_q[z]^k \xrightarrow{\phi_G} \mathbb{F}_q[z]^n \xrightarrow{\pi_G} \mathbb{F}_q[z]^{n-k} \to 0\,,$$

and taking $\operatorname{Hom}_{\mathbb{F}_q[z]}(\ \ , \mathbb{F}_q)$ one obtains the exact sequence of free $\mathbb{F}_q[z]$-modules

$$0 \to \mathbb{F}_q[z]^{n-k} \xrightarrow{\pi^*} \mathbb{F}_q[z]^n \xrightarrow{\phi_G^*} \mathbb{F}_q[z]^k \to 0\,.$$

By construction,

$$\mathcal{C}_k^{\perp} \simeq \operatorname{Im}\pi_G^*\,,$$

from which one concludes that the matrix $H(z)$ defining π_G^* is a basic encoder of $\mathcal{C}_k^{\perp}$ and $\delta_G = \delta_H$. Moreover, one has:

$$H(z) \cdot G(z)^T = 0\,, \tag{11.3}$$

an equality that allows us to compute $H(z)$ from $G(z)$ or viceversa. □

Definition 11.10. A *parity-check (control) matrix* for an (n,k)-convolutional code $\mathcal{C}_k$ is every $(n, n-k)$-generator matrix $H(z)$ of its dual code $\mathcal{C}_k^{\perp}$.

We can easily compute a parity-check matrix $H(z)$ from equation (11.3) when we have a generator matrix $G(z)$ in which the first k columns have rank k, making a base change to turn these columns into the identity matrix of order k.

Definition 11.11. An encoder $G(z)$ of an (n,k)-convolutional code $\mathcal{C}_k \subseteq \mathbb{F}_q(z)^n$ is *systematic* if it is takes the form

$$G(z) = (\mathrm{Id}_{k\times k} \mid \bar{G}(z)_{k\times(n-k)}) .$$

Let $G(z)$ be a systematic encoder and let us decompose the parity-check matrix $H(z)$ as:

$$H(z) = (\bar{H}(z)_{(n-k)\times k} \mid \bar{H}'(z)_{(n-k)\times(n-k)}) .$$

From equation (11.3), one deduces that $\bar{H}(z)+\bar{H}'(z)\cdot\bar{G}(z)^T = 0$. Therefore, $H(z) = \bar{H}'(z)\cdot(-\bar{G}(z)^T \mid \mathrm{Id}_{(n-k)\times(n-k)})$. Thus, we can take simply as a parity-check matrix for $\mathcal{C}_k$

$$H(z) = (-\bar{G}(z)^T \mid \mathrm{Id}) .$$

Example 11.10. In example 11.2, the encoder $G''(z) = (1, 1+z, z+z^2)$ is systematic. A parity check control matrix is:

$$H(z) = \begin{pmatrix} 1+z & 1 & 0 \\ z+z^2 & 0 & 1 \end{pmatrix} .$$

Remark 11.5. There is another possible notion of dual codes (see [14]), according to a *time reversal* $\mathbb{F}_q$-linear pairing

$$\begin{aligned} [\ ,\ ] : \mathbb{F}_q((z))^n \times \mathbb{F}_q((z))^n &\to \mathbb{F}_q \\ (X(z), Y(z)) &\mapsto \sum_t \langle x(t), y(-t)\rangle , \end{aligned}$$

where $X(z) = \sum_t x(t)z^t, Y = \sum_i y(t)z^t \in \mathbb{F}_q((z))^n$, and $\langle\ ,\ \rangle$ is the standard $\mathbb{F}_q$-bilinear form on $\mathbb{F}_q^n$. The duality with respect to this pairing $[\ ,\ ]$ is therefore a duality *over the base field* $\mathbb{F}_q$, whereas the duality with respect to the pairing (11.2) is over the field $\mathbb{F}_q(z)$.

11.3. Convolutional Goppa codes

(For an overview of linear Goppa codes, see [15] or [16], and also chapter ? in this book)

Let $(X, \mathcal{O}_X)$ be a smooth projective curve over $\mathbb{F}_q(z)$ of genus g. Let us denote by Σ_X the field of rational functions of X, and let us assume that $\mathbb{F}_q(z)$ is algebraically closed in Σ_X. Both the Riemann-Roch and the Residue theorems (see for instance [17]) still hold under this hypothesis.

Given a set $p_1, \ldots, p_n$ of n different $\mathbb{F}_q(z)$-rational points of X, if $\mathcal{O}_{p_i}$ denotes the local ring at the point p_i, with maximal ideal $\mathfrak{m}_{p_i}$, and t_i is a local parameter at p_i, one has the exact sequences:

$$\begin{array}{c} 0 \to \mathfrak{m}_{p_i} \to \mathcal{O}_{p_i} \to \mathcal{O}_{p_i}/\mathfrak{m}_{p_i} \simeq \mathbb{F}_q(z) \to 0\,, . \\ s(t_i) \mapsto s(p_i)\,. \end{array} \tag{11.4}$$

Let us consider the divisor $D = p_1 + \cdots + p_n$, with its associated invertible sheaf $\mathcal{O}_X(D)$. One then has an exact sequence of sheaves

$$0 \to \mathcal{O}_X(-D) \to \mathcal{O}_X \to Q \to 0\,, \tag{11.5}$$

where the quotient Q is a sheaf with support at the points p_i.

Let G be a divisor on X of degree r, with support disjoint from D. Tensoring the exact sequence (11.5) by the associated invertible sheaf $\mathcal{O}_X(G)$, one obtains:

$$0 \to \mathcal{O}_X(G-D) \to \mathcal{O}_X(G) \to Q \to 0\,. \tag{11.6}$$

For each divisor F over X, let us denote its $\mathbb{F}_q(z)$-vector space of global sections by

$$L(F) \equiv \Gamma(X, \mathcal{O}_X(F)) = \{s \in \Sigma_X \mid (s) + F \geq 0\}\,,$$

where (s) is the divisor defined by $s \in \Sigma_X$. Taking global sections in (11.6), one obtains

$$\begin{array}{c} 0 \to L(G-D) \to L(G) \stackrel{\alpha}{\to} \mathbb{F}_q(z) \times \overset{n}{\cdots} \times \mathbb{F}_q(z) \to \ldots \\ s \mapsto (s(p_1), \ldots, s(p_n))\,. \end{array}$$

Definition 11.12. The convolutional Goppa code $\mathcal{C}(D,G)$ associated with the pair (D,G) is the image of the $\mathbb{F}_q(z)$-linear map $\alpha\colon L(G) \to \mathbb{F}_q(z)^n$.

Analogously, given a subspace $\Gamma \subseteq L(G)$, one defines the convolutional Goppa code $\mathcal{C}(D,\Gamma)$ as the image of $\alpha_{|\Gamma}$.

Remark 11.6. The above definition is more general than the one given in [11] in terms of families of curves $X \to \mathbb{A}^1_{\mathbb{F}_q}$. In fact, given such a family, the fibre X_η, over the generic point $\eta \in \mathbb{A}^1_{\mathbb{F}_q}$, is a curve over $\mathbb{F}_q(z)$. However, not every curve over $\mathbb{F}_q(z)$ extends to a family over $\mathbb{A}^1_{\mathbb{F}_q}$.

By construction, $\mathcal{C}(D,G)$ is a convolutional code of length n and dimension

$$k \equiv \dim L(G) - \dim L(G-D)\,.$$

Prop 11.1. Let us assume that $2g-2 < r < n$. Accordingly, the evaluation map $\alpha\colon L(G) \hookrightarrow \mathbb{F}_q(z)^n$ is injective, and the dimension of $\mathcal{C}(D,G)$ is

$$k = r + 1 - g\,.$$

Proof. If $r < n$, $\dim L(G-D) = 0$, the map α is injective and $k = \dim L(G)$. If $2g-2 < r$, $\dim L(G) = 1-g+r$ by the Riemann-Roch theorem. □

11.3.1. *Dual convolutional Goppa codes*

Given a convolutional Goppa code $\mathcal{C}(D,G)$, let $\mathcal{C}(D,G)^\perp$ be its dual convolutional code, in the sense of definition 11.9.

Theorem 11.6. *$\mathcal{C}^\perp(D,G)$ is also a convolutional Goppa code, in the following sense: If K denotes the canonical divisor of rational differential forms over X, then $\mathcal{C}^\perp(D,G)$ is the image of the $\mathbb{F}_q(z)$-linear map $\beta\colon L(K+D-G) \to \mathbb{F}_q(z)^n$, given by*

$$\beta(\eta) = (\mathrm{Res}_{p_1}(\eta), \ldots, \mathrm{Res}_{p_n}(\eta))\,.$$

Proof. Following the construction of $\mathcal{C}(D,G)$, we start by tensoring the exact sequence (11.4) by $\mathfrak{m}_{p_i}^* = \mathrm{Hom}_{\mathcal{O}_{p_i}}(\mathfrak{m}_{p_i}, \mathcal{O}_{p_i})$, and we obtain:

$$\begin{aligned} 0 \to \mathcal{O}_{p_i} \to \mathfrak{m}_{p_i}^* \to \mathcal{O}_{p_i}/\mathfrak{m}_{p_i} \otimes_{\mathcal{O}_{p_i}} \mathfrak{m}_{p_i}^* \simeq \mathbb{F}_q(z) \to 0 \\ t_i^{-1} s(t_i) \mapsto s(p_i)\,. \end{aligned} \tag{11.7}$$

Again tensoring (11.7) by $\mathfrak{m}_{p_i}/\mathfrak{m}_{p_i}^2$, the tangent space of differentials at the point p_i, one obtains:

$$\begin{aligned} 0 \to \mathfrak{m}_{p_i}/\mathfrak{m}_{p_i}^2 \to \mathfrak{m}_{p_i}^* \otimes_{\mathcal{O}_{p_i}} \mathfrak{m}_{p_i}/\mathfrak{m}_{p_i}^2 \to \mathbb{F}_q(z) \to 0 \\ t_i^{-1} s(t_i) dt_i \mapsto s(p_i)\,, \end{aligned} \tag{11.8}$$

where $s(p_i) = \mathrm{Res}_{p_i}(t_i^{-1} s(t_i) dt_i)$.

This allows us to define a new convolutional Goppa code associated with the pair of divisors $D = p_1 + \cdots + p_n$ and G; tensoring (11.5) by the line sheaf $\mathcal{O}_X(K+D-G)$, one has:

$$0 \to \mathcal{O}_X(K-G) \to \mathcal{O}_X(K+D-G) \to Q \to 0\,. \tag{11.9}$$

Taking global sections, one has

$$\begin{aligned} 0 \to L(K-G) \to L(K+D-G) \xrightarrow{\beta} \mathbb{F}_q(z) \times \overset{n}{\cdots} \times \mathbb{F}_q(z) \to \ldots \\ \eta \mapsto (\mathrm{Res}_{p_1}(\eta), \ldots, \mathrm{Res}_{p_n}(\eta))\,. \end{aligned}$$

The image of β is a subspace of $\mathbb{F}_q(z)^n$, whose dimension can be calculated by the Riemann-Roch theorem:

$$\begin{aligned}&\dim L(K+D-G)-\dim L(K-G)\\&=\dim L(G-D)-(r-n)-1+g-(\dim L(G)-r-1+g)\\&=n-k\,.\end{aligned}$$

Moreover, $\operatorname{Im}\beta$ is the subspace $\mathcal{C}(D,G)^\perp \subset \mathbb{F}_q(z)^n$, because they have the same dimension, and by the Residue theorem for every $\eta \in L(K+D-G)$ and every $s \in L(G)$ one has

$$\langle \beta(\eta), \alpha(s)\rangle = \sum_{i=1}^{n} s(p_i)\operatorname{Res}_{p_i}(\eta) = \sum_{i=1}^{n} \operatorname{Res}_{p_i}(s\eta) = 0\,. \qquad \square$$

Under the hypothesis $2g-2 < r < n$, the map β is injective, and $\mathcal{C}^\perp(D,G)$ is a convolutional code of length n and dimension

$$\dim L(K+D-G) = n-(1-g+r)\,.$$

Remark 11.7. In the context of duality in the sense of the pairing $[\ ,\ ]$ of remark 11.5,

$$[X(z),Y(z)] = \operatorname{Res}_{z=0}\left(\langle X(z),Y(z)\rangle \frac{dz}{z}\right) = \sum_{i=1}^{n} \operatorname{Res}_{z=0}\left(X_i(z)Y_i(z)\frac{dz}{z}\right)\,.$$

Thus, the duality for convolutional Goppa codes in the sense of Definition 11.9 is related to the residues at the points of X, and the duality with respect to the pairing $[\ ,\ ]$ is related to the residues in the variable of the base field.

11.4. Weights and (free)distance

For vectors $x=(x_1,\dots,x_n)\in\mathbb{F}_q^n$, the *(Hamming) weight* is defined as $hwt(x)=\#\{i \mid x_i \neq 0\}$ and the *(Hamming) distance* between $x,y\in\mathbb{F}_q^n$ can be defined as the weight $hwt(y-x)$. In the setting of linear coding theory, the corresponding notion of *minimum weight (distance)* of the words in the code is one of the most important parameters of the code.

For convolutional codes, one needs an analogous notion for polynomial vectors $X(z)=(X(z)_1,\dots,X_n(z))\in\mathbb{F}_q[z]^n$. However, it is possible to define two kinds of weights. First, one can simply define the *Hamming weight of* $X(z)$ as

$$hwt(X(z)) = \#\{i \mid X_i(z)\neq 0\}\,.$$

Thus, the concept of *minimum Hamming weight of a convolutional code* does not reflect the performance of convolutional codes over noisy channels in convolutional coding theory. Of course the minimum Hamming weight of a convolutional Goppa code $\mathcal{C}(D,G)$ can be bounded using the Riemann-Roch theorem, as in the usual Goppa codes.

However, when one considers $X(z) \in \mathbb{F}_q[z]^n$ as a polynomial with vector coefficients

$$X(z) = \sum_t x(t) z^t \,, \text{ where } x(t) = (x_1(t), \ldots, x_n(t) \in \mathbb{F}_q^n \,,$$

one can define a more natural notion of *weight in convolutional coding theory*:

Definition 11.13. The *weight* of $X(z) \in \mathbb{F}_q[z]^n$ is

$$wt(X(z)) = \sum_t hwt(x(t)) \,.$$

Definition 11.14. The *(free) distance* of a (n,k)-convolutional code $\mathcal{C}_k \subseteq \mathbb{F}_q(z)^n$ is

$$d = \text{Min}\{wt(X(z)) \mid X(z) \in \mathcal{C}_k \cap \mathbb{F}_q[z]^n \,,\ X(z) \neq 0\} \,.$$

In particular, if the degree of the code is zero, $\mathcal{C}_k$ is a linear code and the (free) distance is the (minimum) distance as linear code.

As in the case of linear codes, the distance d is one of the most important parameters in convolutional coding theory.

In particular, an interesting problem is to find upper bounds for d. A possible solution is to link convolutional codes with linear codes, fixing the degree of the words in $\mathcal{C}_k$:

Theorem 11.7. *(McEliece [7]) For an (n,k)-convolutional code $\mathcal{C}_k \subseteq \mathbb{F}_q(z)^n$ of Forney indices $e_1 \leq \cdots \leq e_k$, let*

$$\mathcal{C}_L = \{X(z) \in \mathcal{C} \ / \ \text{Degree}(X(z)) \leq L\} \,.$$

Identifying the set of all possible n-dimensional polynomial vectors of degree $\leq L$ over $\mathbb{F}_q$ with $\mathbb{F}_q^{n(L+1)}$, one can see $\mathcal{C}_L$ as an $\mathbb{F}_q$-linear code of length $n(L+1)$ and a certain dimension k_L. Then,

$$k_L = \sum_{i=1}^{k} \text{Max}(L+1-e_i, 0) \,,$$

and

$$d \leq \text{Min}_{L \geq 0} \left(\text{Max}\{\textit{distance of possible } (n(L+1), k_L) \textit{ linear codes}\}\right) \,.$$

This result can be used to calculate the distance d of a convolutional code $\mathcal{C}_k$ from the distances d_L of its linear (sub)codes $\mathcal{C}_L$.

Also it is possible for convolutional codes to find a bound of d analogous to the Singleton bound for linear codes.

Theorem 11.8. *(Rosenthal, Smarandache [8]) If $\mathcal{C}_k \subseteq \mathbb{F}_q(z)^n$ is an (n,k)-convolutional code of degree δ, its distance is bounded by:*

$$d \leq (n-k)\big(\lfloor \frac{\delta}{k} \rfloor + 1\big) + \delta + 1\,.$$

If d achieves this bound, then $\mathcal{C}_k$ will be called a Maximum Distance Separable (MDS) *convolutional code.*

For linear codes, there is a geometric interpretation of distance, in terms of *balls* in the words of the code. Moreover, for linear Goppa codes the distance can be viewed in terms of the number of zeroes of certain meromorphic functions, which allows us to use the Riemann-Roch theorem to make very precise computations.

In the case of convolutional Goppa codes $\mathcal{C}(D,G)$ of length n over a curve X defined over $\mathbb{F}_q(z)$, the interpretation of the notion of weight in geometric terms is much more difficult. Let us assume that X can be extended to a family of curves X_U over $U = \operatorname{Spec}\mathbb{F}_q[z] = \mathbb{A}^1_{\mathbb{F}_q}$ (as in [11]). Let X_0 be the fibre of X_U over the origin of U. The points $p_1,\ldots,p_n$ of the divisor D define sections $p_i(z)\colon \mathbb{A}^1_{\mathbb{F}_q} \to X_U$ and the polynomial words of the code $\mathcal{C}(D,G)$ are defined by evaluating the sections $s \in L(G)$ along the sections $p_i(z)$.

Let p be one of the points defined by D, C_p the curve of X_U defined as the image of the section $p(z)$, and q_0 the intersection of C_p with X_0; that is, $q_0 = p(0)$. Let us assume that $L(G)$ is a very ample linear series [17], and let us assume that X_U is immersed in $\mathbb{P}^N_{\mathbb{F}_q} \times \mathbb{P}^1_{\mathbb{F}_q}$ using the linear series $L(G)$. Let us denote by $\pi_r(q_0)$ the r-th osculating plane to the curve C_p at the point q_0. One has a sequence of strict inclusions:

$$\pi_0(q_0) = q_0 \subset \pi_1(q_0) \subset \pi_2(q_0) \subset \cdots \subset \pi_r(q_0) \subset \cdots\,.$$

The evaluation of s at p, $s(p)$, can be expressed by:

$$s(p) = s_0 + s_1 z + \cdots + s_n z^n\,,$$

where $s_0 = s(0)$ and s_r, the r-th coefficient, can be interpreted as the r-th jet of $s(z)$ at the point q_0.

With this interpretation in mind, one has that $s_r = 0$ if and only if

$$H_s \cap \pi_r(q_0) \neq \emptyset \text{ and } H_s \cap \pi_{r-1}(q_0) \subsetneqq H_s \cap \pi_r(q_0)\,,$$

where H_s is the hyperplane defined by the section s.

Accordingly, the problem of computing the number $\#\{r \mid s_r = 0\}$ can be translated into a problem of enumerative geometry over finite fields.

The main problem here is to develop the classical theory of osculating planes and all the classical computations in the case of finite base fields. This is not an easy problem, but its solution would allow one to give a geometric interpretation of the distance of convolutional Goppa codes.

11.5. Convolutional Goppa Codes over the projective line

Let $X = \mathbb{P}^1_{\mathbb{F}_q(z)} = \operatorname{Proj} \mathbb{F}_q(z)[x_0, x_1]$ be the projective line over the field $\mathbb{F}_q(z)$, and let us denote by $t = x_1/x_0$ the affine coordinate.

Let $p_0 = (1, 0)$ be the origin point, $p_\infty = (0, 1)$ the point at infinity, and let $p_1, \ldots, p_n$ be different rational points of $\mathbb{P}^1$, $p_i \neq p_0, p_\infty$. Let us define the divisors $D = p_1 + \cdots + p_n$ and $G = rp_\infty - sp_0$, with

$$0 \leq s \leq r < n\,.$$

Since $g = 0$, the evaluation map $\alpha\colon L(G) \to \mathbb{F}_q(z)^n$ is injective, and $\operatorname{Im}\alpha$ defines a convolutional Goppa code $\mathcal{C}(D, G)$ of length n and dimension $k = r - s + 1$.

Let us choose the functions $t^s, t^{s+1}, \ldots, t^r$ as a basis of $L(G)$. If $\alpha_i \in \mathbb{F}_q(z)$ is the local coordinate of the point p_i, $i = 1, \ldots, n$, the matrix of the evaluation map α is the following generator matrix for the code $\mathcal{C}(D, G)$:

$$G = \begin{pmatrix} \alpha_1^s & \alpha_2^s & \ldots & \alpha_n^s \\ \alpha_1^{s+1} & \alpha_2^{s+1} & \ldots & \alpha_n^{s+1} \\ \vdots & \vdots & \ddots & \vdots \\ \alpha_1^r & \alpha_2^r & \ldots & \alpha_n^r \end{pmatrix}. \tag{11.10}$$

The dual convolutional Goppa code $\mathcal{C}^\perp(D, G)$ also has length n, and dimension $n - k = n - r + s - 1$.

To construct $\mathcal{C}^\perp(D, G)$, let us choose in $L(K + D - G)$ the basis of rational differential forms:

$$\left\langle \frac{dt}{t^s \prod_{i=1}^n (t - \alpha_i)}, \frac{t\, dt}{t^s \prod_{i=1}^n (t - \alpha_i)}, \ldots, \frac{t^{n-r+s-2} dt}{t^s \prod_{i=1}^n (t - \alpha_i)} \right\rangle,$$

and let us calculate the residues:

$$\begin{aligned}
&\mathrm{Res}_{p_j}\left(\frac{t^m dt}{t^s \prod_{i=1}^{n}(t-\alpha_i)}\right) = \\
&= \mathrm{Res}_{p_j}\left(\frac{(t-\alpha_j+\alpha_j)^m d(t-\alpha_j)}{(t-\alpha_j)(t-\alpha_j+\alpha_j)^s \prod_{\substack{i=1\\ i\neq j}}^{n}(t-\alpha_j+\alpha_j-\alpha_i)}\right) \\
&= \frac{\alpha_j^m}{\alpha_j^s \prod_{\substack{i=1\\ i\neq j}}^{n}(\alpha_j-\alpha_i)}\,.
\end{aligned}$$

If one sets $h_j = \frac{1}{\alpha_j^s \prod_{\substack{i=1\\ i\neq j}}^{n}(\alpha_j-\alpha_i)}$, then the matrix H of

$\beta\colon L(K+D-G) \to \mathbb{F}_q(z)^n$,

$$H = \begin{pmatrix} h_1 & h_2 & \dots & h_n \\ h_1\alpha_1 & h_2\alpha_2 & \dots & h_n\alpha_n \\ \vdots & \vdots & \ddots & \vdots \\ h_1\alpha_1^{n-r+s-2} & h_2\alpha_2^{n-r+s-2} & \dots & h_n\alpha_n^{n-r+s-2} \end{pmatrix}, \tag{11.11}$$

is a generator matrix for the dual code $\mathcal{C}^{\perp}(D,G)$, and therefore a parity-check matrix for $\mathcal{C}(D,G)$. In fact, one has $H \cdot G^T = 0$.

Remark 11.8. The matrix in (11.11) suggests that $\mathcal{C}^{\perp}(D,G)$ is an alternating code over the field $\mathbb{F}_q(z)$, and we can thus apply to $\mathcal{C}(D,G)$ some kind of Berlekamp-Massey decoding algorithm as a linear code over $\mathbb{F}_q(z)$.

Example 11.11. Let $a, b \in \mathbb{F}_q$ be two different non-zero elements, and

$$\alpha_i = a^{i-1}z + b^{i-1}\,, i = 1, \dots, n\,, \text{ with } n < q\,.$$

We present some examples of convolutional Goppa codes with canonical generator matrices, whose distance d attains the generalized Singleton bound 11.8 (i.e., they are MDS convolutional codes), and we include their encoding equations as linear systems.

- Field $\mathbb{F}_3(z)$, $\mathbb{F}_3 = \{0,1,2\}$:

$$\begin{aligned}
&G = \begin{pmatrix} z+1 & z+2 \end{pmatrix} \\
&H = \begin{pmatrix} \frac{1}{2(z+1)} & \frac{1}{z+2} \end{pmatrix} \\
&A = \begin{pmatrix} 0 \end{pmatrix},\ B = \begin{pmatrix} 1 \end{pmatrix},\ C = \begin{pmatrix} 1 & 1 \end{pmatrix},\ D = \begin{pmatrix} 1 & 2 \end{pmatrix} \\
&(n,k,\delta,d) = (2,1,1,4)\,.
\end{aligned}$$

- Field $\mathbb{F}_4(z)$, $\mathbb{F}_4 = \{0, 1, \alpha, \alpha^2\}$ where $\alpha^2 + \alpha + 1 = 0$:

$$G = \begin{pmatrix} 1 & 1 & 1 \\ z+1 & \alpha z + \alpha^2 & \alpha^2 z + \alpha \end{pmatrix}$$
$$H = \begin{pmatrix} \frac{1}{(\alpha^2 z+\alpha)(\alpha z+\alpha^2)} & \frac{1}{(\alpha^2 z+\alpha)(z+1)} & \frac{1}{(\alpha z+\alpha^2)(z+1)} \end{pmatrix}$$
$$A = \begin{pmatrix} 0 \end{pmatrix},\ B = \begin{pmatrix} 0 \\ 1 \end{pmatrix},\ C = \begin{pmatrix} 1 & \alpha & \alpha^2 \end{pmatrix},\ D = \begin{pmatrix} 1 & 1 & 1 \\ 1 & \alpha^2 & \alpha \end{pmatrix}$$
$$(n, k, \delta, d) = (3, 2, 1, 3)\,.$$

- Field $\mathbb{F}_4(z)$:

$$G = \begin{pmatrix} z+1 & z+\alpha & z+\alpha^2 \end{pmatrix}$$
$$H = \begin{pmatrix} \frac{1}{z+1} & \frac{\alpha}{z+\alpha} & \frac{\alpha^2}{z+\alpha^2} \\ 1 & \alpha & \alpha^2 \end{pmatrix}$$
$$A = \begin{pmatrix} 0 \end{pmatrix},\ B = \begin{pmatrix} 1 \end{pmatrix},\ C = \begin{pmatrix} 1 & 1 & 1 \end{pmatrix},\ D = \begin{pmatrix} 1 & \alpha & \alpha^2 \end{pmatrix}$$
$$(n, k, \delta, d) = (3, 1, 1, 6)\,.$$

- Field $\mathbb{F}_5(z)$, $\mathbb{F}_5 = \{0, 1, 2, 3, 4\}$:

$$G = \begin{pmatrix} (z+1)^2 & (z+2)^2 & (z+4)^2 \end{pmatrix}$$
$$H = \begin{pmatrix} \frac{2}{(z+1)^2} & \frac{2}{(z+2)^2} & \frac{1}{(z+4)^2} \\ \frac{2}{z+1} & \frac{2}{z+2} & \frac{1}{z+4} \end{pmatrix}$$
$$A = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix},\ B = \begin{pmatrix} 1 & 0 \end{pmatrix},\ C = \begin{pmatrix} 2 & 4 & 3 \\ 1 & 1 & 1 \end{pmatrix},\ D = \begin{pmatrix} 1 & 4 & 1 \end{pmatrix}$$
$$(n, k, \delta, d) = (3, 1, 2, 9)\,.$$

- Field $\mathbb{F}_5(z)$:

$$G = \begin{pmatrix} z+1 & 2z+3 & 4z+4 & 3z+2 \\ (z+1)^2 & (2z+3)^2 & (4z+4)^2 & (3z+2)^2 \end{pmatrix}$$
$$H = \begin{pmatrix} \frac{4}{a^2 bc} & \frac{4}{bcd^2} & \frac{4}{a^2 bc} & \frac{4}{bcd^2} \\ \frac{4}{abc} & \frac{3}{bcd} & \frac{1}{abc} & \frac{2}{bcd} \end{pmatrix}$$
where $a = z+1$, $b = z+2$, $c = z+3$ and $d = z+4$,
$$A = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix},\ B = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix},\ C = \begin{pmatrix} 1 & 2 & 4 & 3 \\ 2 & 2 & 2 & 2 \\ 1 & 4 & 1 & 4 \end{pmatrix},\ D = \begin{pmatrix} 1 & 3 & 4 & 2 \\ 1 & 4 & 1 & 4 \end{pmatrix}$$
$$(n, k, \delta, d) = (4, 2, 3, 8)\,.$$

11.6. Convolutional Goppa Codes over elliptic curves

We can obtain convolutional codes from elliptic curves in the same way. Let $X \subset \mathbb{P}^2_{\mathbb{F}_q(z)}$ be a plane elliptic curve over $\mathbb{F}_q(z)$, and let us denote by (x, y) the affine coordinates in $\mathbb{P}^2_{\mathbb{F}_q(z)}$. Let p_∞ be the infinity point, and $p_1, \ldots, p_n$ rational points of X, with $p_i = (x_i(z), y_i(z))$. Let us define $D = p_1 + \cdots + p_n$ and $G = rp_\infty$.

The *canonical* basis of $L(G)$ is $\{1, x, y, \ldots, x^a y^b\}$, with $2a+3b = r$ (and $b = 0, 1$ so that there are no linear combinations). Thus, the evaluation map $\alpha\colon L(G) \to \mathbb{F}_q(z)^n$ is:

$$\alpha(x^i y^j) = (x_1^i(z)y_1^j(z), \ldots, x_n^i(z)y_n^j(z)).$$

The image of a subspace $\Gamma \subseteq L(G)$ under the map α provides a Goppa convolutional code.

We present a couple of examples obtained from elliptic curves that, although not MDS, have distance approaching that bound.

Example 11.12. We consider the curve over $\mathbb{F}_2(z)$

$$y^2 + (1+z)xy + (z+z^2)y = x^3 + (z+z^2)x^2\,,$$

and the points

$$\begin{aligned} p_1 &= (z^2+z, z^3+z^2) \\ p_2 &= (0, z^2+z) \\ p_3 &= (z, z^2)\,. \end{aligned}$$

$L(G)$ is the subspace generated by $\{1, x\}$. Thus, the valuation map α is defined by the matrix

$$\begin{pmatrix} 1 & 1\,1 \\ z^2+z & 0\ z \end{pmatrix}.$$

This code has distance $d = 2$. The maximum distance for its parameters is 3.

Example 11.13. Let us now consider the curve over $\mathbb{F}_2(z)$

$$y^2 + (1+z+z^2)xy + (z^2+z^3)y = x^3 + (z^2+z^3)x^2\,,$$

and the points

$$\begin{aligned} p_1 &= (z^3+z^2, 0) \\ p_2 &= (0, z^3+z^2) \\ p_3 &= (z^3+z^2, z^5+z^3) \\ p_4 &= (z^2+z, z^3+z) \\ p_5 &= (z^2+z, z^4+z^2)\,. \end{aligned}$$

Again we take $L(G)$ as the subspace generated by $\{1, x\}$. Therefore, the valuation map α is defined by the matrix

$$\begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ z^3+z^2 & 0 & z^3+z^2 & z^2+z & z^2+z \end{pmatrix}.$$

This code has distance $d = 4$. The maximum distance for its parameters is 5.

Remark 11.9. Every elliptic curve X over $\mathbb{F}_q(z)$ can be considered the generic fibre of a fibration $\mathcal{X} \to U = \operatorname{Spec} \mathbb{F}_q[z]$, with some fibres singular curves of genus 1. The global structure of this fibration is related to the singular fibres (see [18]); the translation into the language of coding theory of the arithmetic and geometric properties of the fibration is the first step in the program of applying the general construction to the effective construction of good convolutional Goppa codes of genus 1.

References

[1] P. Elias, Coding for noisy channels, *I.R.E. Nat. Conv. Record* **3**, 34–45, (1955).
[2] A.J. Viterbi, Error bounds for convolutional codes and an asymptotically optimum decoding algorithm, *IEEE Trans. Inf. Theory* **13** (2), 260–269, (1967).
[3] C. Berrou, A. Glavieux and P. Thitimajshima, Near Shannon limit error-correcting coding and decoding: Turbo codes (1), *IEEE Int. Conf. on Communications (ICC'93)*, 1064–1070, (1993).
[4] G.D. Forney Jr, Convolutional codes I: Algebraic structure, *IEEE Trans. Inform. Theory*, **16** (3), 720–738, (1970).
[5] R.J. McEliece, The theory of information and coding, (vol. 3 of the Encylcopedia of Mathematics and Its Applications). Addison-Wesley, Reading, MA, (1977).
[6] P. Piret, Convolutional codes: An Algebraic Approach, MIT Press, Cambridge, MA, (1988).
[7] R.J. McEliece, The algebraic theory of convolutional codes, in *Handbook of coding theory, Vol. I*, 1065–1138, North-Holland, Amsterdam, (1998).
[8] J. Rosenthal and R.Smarandache, Maximum distance separable convolutional codes, *Appl. Algebra Engrg. Comm. Comput.*, **10** (1), 15–32, (1999).
[9] R.Smarandache, H. Gluesing-Luersen and J. Rosenthal, Constructions of MDS-Convolutional Codes, *IEEE Trans. Inform. Theory*, **47** (5), 2045–2049, (2001).
[10] V. Lomadze, Convolutional Codes and Coherent Sheaves, *Algebra Engrg. Comm. Comput.*, **12** (4), 273–326, (2001).
[11] J.A. Domínguez Pérez, J.M. Muñoz Porras and G. Serrano Sotelo, Convolutional Codes of Goppa type, *Algebra Engrg. Comm. Comput.*, **15** (1), 51–61, (2004).

[12] J.M. Muñoz Porras, J.A. Domínguez Pérez, J.I. Iglesias Curto and G. Serrano Sotelo, Convolutional Goppa Codes, *IEEE Trans. Inform. Theory*, **52** (1), 340–344, (2006).

[13] J.L. Massey and M.K. Sain, Inverses of linear sequential circuits, *IEEE Trans. Comp.* **19** (5), 330–337, (1968).

[14] J.Rosenthal, Connections between linear systems and convolutional codes, in *Codes, systems, and graphical models (Minneapolis, MN, 1999) IMA Vol. Math. Appl.* 39–66, Springer, New York, (2001)

[15] T. Høholdt, J.H. van Lint and R. Pellikaan, Algebraic Geometric Codes, in *Handbook of coding theory, Vol. I*, 871–962, North-Holland, Amsterdam, (1998).

[16] J.H. van Lint and G. van der Geer, Introduction to Coding Theory and Algebraic Geometry (DMV Seminar, vol. 12), Birkhäuser, Basel, (1998)

[17] R. Hartshorne, Algebraic geometry, (Graduate Texts in Mathematics, vol. 52), Springer-Verlag, New York, (1977).

[18] J. Tate, Algorithm for determining the type of a singular fiber in an elliptic pencil, in *Modular functions of one variable, IV (Proc. Internat. Summer School, Univ. Antwerp, Antwerp,1972)*, (Lecture Notes in Math., vol. 476), 33–52, Springer-Verlag, Berlin, (1975).