Alternant codes

Alternant codes are defined as *subfield subcodes* of GRS codes.

Definition 1 Let $\mathcal{C}$ be an $[n, k]$ linear code over $\mathbb{F}_{q^m}$. The *Subfield Subcode* $\mathcal{C}|_{\mathbb{F}_q}$ of $\mathcal{C}$ over $\mathbb{F}_q$ is the vector space $\mathcal{C} \cap \mathbb{F}_q^n$.

The easiest way to construct a subfield subcode is by using a trace construction. It is easy to verify that the dual of $\mathcal{C}|_{\mathbb{F}_q}$ is the trace of the dual of $\mathcal{C}$. Since a generator matrix for the dual code is in fact a parity-check matrix for $\mathcal{C}$, in practice this means that it's possible to build a parity-check matrix for the subfield subcode directly from $\mathcal{C}$.

Theorem 1 *Let $\mathcal{C}$ be an $[n, k, d]$ linear code over $\mathbb{F}_{q^m}$. Fix an ordered basis $E = \{\boldsymbol{b}_1, \ldots, \boldsymbol{b}_m\}$ for $\mathbb{F}_{q^m}$ over $\mathbb{F}_q$ and the corresponding projection function $\phi_E : \mathbb{F}_{q^m} \to \mathbb{F}_q^m$ defined by $\phi_E(\alpha) = (a_1, \ldots, a_m)^T$ for $\alpha = a_1\boldsymbol{b}_1 + \cdots + a_m\boldsymbol{b}_m$. Let H be a parity-check matrix for $\mathcal{C}$. Then the subfield subcode $\mathcal{C}|_{\mathbb{F}_q}$ is an $[n, k', d']$ linear code over $\mathbb{F}_q$, where $k' \geq n - m(n - k)$, $d' \geq d$ and $\hat{H} = \phi_E(H)$ is a parity-check matrix for it.*

Definition 2 Let $\mathsf{GRS}_r(\boldsymbol{x}, \boldsymbol{y})$ be a GRS code of order r over $\mathbb{F}_{q^m}$ for a certain prime power q and extension degree $m > 1$. The *Alternant Code* $\mathsf{A}_r(\boldsymbol{x}, \boldsymbol{y})$ is the subfield subcode $\mathsf{GRS}_r(\boldsymbol{x}, \boldsymbol{y})|_{\mathbb{F}_q}$.

Alternant codes admit a modified version of the Berlekamp-Massey algorithm, which is presented below. First it is necessary to introduce a few important notions.

Definition 3 Let $\mathsf{A}_r(\boldsymbol{x}, \boldsymbol{y})$ be an alternant code over $\mathbb{F}_q$ as defined above and let $\boldsymbol{x}$ be the transmitted codeword. Suppose the received vector is $\boldsymbol{z} = \boldsymbol{x} + \boldsymbol{e}$, with $\mathsf{wt}(\boldsymbol{e}) = w$ within the correction range and error values $v_1, \ldots, v_w$ in positions $p_1, \ldots, p_w$. Then the following objects are defined:

- *Error Locators* the elements $x_{p_1}, \ldots, x_{p_w}$

- *Error Locator Polynomial* the polynomial $\Lambda(z) = \prod_{i=1}^{w}(1 - x_{p_i}z)$

- *Error Evaluator Polynomial* the poly $\Omega(z) = \sum_{j=1}^{w} v_j y_{p_j} \prod_{\substack{1 \leq i \leq w \\ i \neq j}} (1 - x_{p_i}z)$.

It is evident that the error positions are uniquely determined by the reciprocals of the roots of Λ. Once these are found, the error values are given by

$$v_j = \frac{\Omega(x_{p_j}^{-1})}{y_{p_j} \prod_{\substack{1 \leq i \leq w \\ i \neq j}} (1 - x_{p_i}x_{p_j}^{-1})}. \tag{1}$$

Table 1: Alternant decoding

Input	An $r \times n$ parity-check matrix $H(\boldsymbol{x}, \boldsymbol{y})$ and the received word $\boldsymbol{z} = \boldsymbol{x} + \boldsymbol{e} \in \mathbb{F}_q^n$.
Output	The codeword $\boldsymbol{x}$.
1.	Calculate the syndrome $\boldsymbol{s} = H\boldsymbol{z}^T$ and write down the corresponding polynomial $S(z) = \sum_{i=0}^{r-1} s_i z^i$.
2.	Use the Euclidean algorithm for polynomials to solve the *key equation* $$\Omega(z) \equiv \Lambda(z)S(z) \pmod{z^r} \quad (2)$$ and retrieve Λ and Ω.
3.	Use a root-finding algorithm[1] to find the roots of Λ. Find the corresponding error positions $p_1, \ldots, p_w$ and then the values $v_1, \ldots, v_w$; build the error vector $\boldsymbol{e}$ having the value v_i in position p_i for $i = 1, \ldots, w$ and 0 everywhere else. Return $\boldsymbol{x} = \boldsymbol{z} - \boldsymbol{e}$.

Among alternant codes are some very important families of algebraic codes, such as:

- *Chien-Choy generalized BCH codes*
- *Goppa codes*
- *Generalized Srivastava codes*

We will analyze in detail Goppa codes, which are of cryptographic importance.

Goppa Codes

Goppa codes were first introduced in 1970s by Victor Goppa and represent a simple case of *algebraic-geometric codes*. Those are evaluation codes, like RS codes, but where the objects involved, rather than polynomials, are functions evaluated on rational points of a certain algebraic curve. The original formulation is the following.

Let χ be an algebraic curve over $\mathbb{F}_q$, $P_1, \ldots, P_n$ distinct rational points on χ and $\mathcal{D}$ the divisor $[P_1] + \cdots + [P_n]$. Let $\mathcal{G}$ be another divisor such that $\text{supp}(\mathcal{G}) \cap \text{supp}(\mathcal{D}) = \emptyset$ and denote by $L(\mathcal{G})$ the unique[2] finite-dimensional vector space[3], with respect to the divisor $\mathcal{G}$, such that $L(\mathcal{G})$ is a subspace of $\mathbb{F}$, the function field of χ. The *Goppa Code* $\Gamma(\mathcal{D}, \mathcal{G})$ is defined by

$$\Gamma(\mathcal{D}, \mathcal{G}) = \{(f(P_1), \ldots, f(P_n)) : f \in L(\mathcal{G})\}. \quad (3)$$

Sometimes, Goppa codes expressed in this way are referred to as *geometric Goppa codes*.
An equivalent, more common formulation is given by means of a generator polynomial, much like BCH codes, and makes use of the subfield subcode construction.

[1] Commonly a *Chien search*.
[3] By the Riemann-Roch theorem.
[3] That is, $L(\mathcal{G}) = \{f \in \mathbb{F} \ : \ div(f) + \mathcal{D} \geq 0\} \cup \{0\}$.

Definition 4 Fix a finite field $\mathbb{F}_q$ and an extension degree $m > 1$. Choose a polynomial $g(x)$ in $\mathbb{F}_{q^m}[x]$ of degree $\ell < n/m$ and a sequence of distinct elements $\alpha_1, \ldots, \alpha_n \in \mathbb{F}_{q^m}$ (called *support*) such that $g(\alpha_i) \neq 0$ for all i. The polynomial $g(x)$ is called the *Goppa Polynomial*. Define the $[n, n-\ell]$ linear code $\mathcal{C}$ over $\mathbb{F}_{q^m}$ as the set of words $\boldsymbol{a} = (a_1, \ldots, a_n) \in \mathbb{F}_{q^m}^n$ such that

$$\sum_{i=1}^{n} \frac{a_i}{x - \alpha_i} \equiv 0 \pmod{g(x)}. \tag{4}$$

The *Goppa Code* $\Gamma = \Gamma(\alpha_1, \ldots, \alpha_n, g)$ over $\mathbb{F}_q$ is the corresponding subfield subcode $\mathcal{C}|_{\mathbb{F}_q}$.

It is easy to see that a Goppa code defined in this way admits a parity-check matrix of the form

$$H(\boldsymbol{\alpha}, g) = \begin{pmatrix} \frac{1}{g(\alpha_1)} & \cdots & \frac{1}{g(\alpha_n)} \\ \vdots & \vdots & \vdots \\ \frac{\alpha_1^{\ell-1}}{g(\alpha_1)} & \cdots & \frac{\alpha_n^{\ell-1}}{g(\alpha_n)} \end{pmatrix} \tag{5}$$

from which is possible to see that the Goppa code Γ is de facto an alternant code, precisely $\mathsf{A}_\ell(\boldsymbol{x}, \boldsymbol{y})$ with $x_i = \alpha_i$, $y_i = 1/g(\alpha_i)$ for $i = 1, \ldots, n$.
It is then also evident that a Goppa code has dimension $k \geq n - m\ell$. The minimum distance is $\ell + 1$, or $2\ell + 1$ in the special binary case ($q = 2$).

Goppa codes enjoy a particularly efficient decoding algorithm, an adaptation of the Berlekamp-Massey algorithm given by Patterson.

Coding Theory Problems

The following problem, commonly called *general decoding problem (GDP)*, is the main hard problem related to coding theory.

Table 2: General Decoding Problem

Given	An $[n, k]$ linear code $\mathcal{C}$ over $\mathbb{F}_q$ and a vector $\boldsymbol{y} \in \mathbb{F}_q^n$.
Goal	Find $\boldsymbol{x} \in \mathcal{C}$ such that $\mathsf{d}(\boldsymbol{x}, \boldsymbol{y})$ is minimal.

Note that this corresponds to correcting a certain number of errors occurred on the codeword $\boldsymbol{x}$, represented by an error vector $\boldsymbol{e}$, that is $\boldsymbol{y} = \boldsymbol{x} + \boldsymbol{e}$. By the theorem presented in the previous lesson, a unique solution exists if the weight of $\boldsymbol{e}$ is less than or equal to $w = \lfloor \frac{d-1}{2} \rfloor$, where d is the minimum distance of $\mathcal{C}$.

This problem is well known and was proved to be NP-complete by Berlekamp, McEliece and van Tilborg. Moreover, GDP is believed to be hard on average, and not just on the worst-case instances.

An alternative and very popular formulation is given in terms of the parity-check matrix, and is known as the *Syndrome Decoding Problem (SDP)*.

Table 3: Syndrome Decoding Problem

Given	An $[n-k, n]$ parity-check matrix for an $[n, k]$ linear code $\mathcal{C}$ over $\mathbb{F}_q$, a vector $\boldsymbol{s} \in \mathbb{F}_q^{n-k}$ and an integer $w \in \mathbb{N}^+$.
Goal	Find $\boldsymbol{e} \in \mathbb{F}_q^n$ of weight $\leq w$ such that $\boldsymbol{s} = H\boldsymbol{e}^T$.

Sometimes SDP is referred to as *computational* syndrome decoding problem. However, note that, for all of the above problems, there is no difference between the computational and the decisional versions of SDP, so the distinction is not needed.

The following is a very important bound for linear codes:

Definition 5 Let $\mathcal{C}$ be an $[n, k]$ linear code over $\mathbb{F}_q$. The *Gilbert-Varshamov (GV) Distance* is the largest integer d_0 such that

$$\sum_{i=0}^{d_0-1} \binom{n}{i} (q-1)^i \leq q^{n-k}. \tag{6}$$

It is then clear that, if $w \leq d_0$, there is a unique solution to SDP. Otherwise, multiple solutions exist. It follows that decoding problems are meaningful only if the weight w is *small*. If the given syndrome is random, then the weight is likely to be close to the GV bound, therefore providing a guarantee for the hardness of the problem. However in practice, for cryptographic schemes the weight is much smaller since it has to be within the correction range of the code.

The McEliece Cryptosystem

As the name suggests the scheme is due to Robert J. McEliece and dates back to 1978.

Table 4: The McEliece cryptosystem

Setup	Fix public system parameters $q, m, n, k, w \in \mathbb{N}$ such that $k \geq n - wm$.
K	$\mathsf{K}_{\mathsf{publ}}$ the set of $k \times n$ matrices over $\mathbb{F}_q$. $\mathsf{K}_{\mathsf{priv}}$ the set of triples formed by a $k \times k$ invertible matrix, an $n \times n$ permutation matrix and a code description[4].
P	The vector space $\mathbb{F}_q^k$.
C	The vector space $\mathbb{F}_q^n$.
KeyGen	Generate at random a polynomial $g \in \mathbb{F}_{q^m}[x]$ and elements $\alpha_1, \ldots, \alpha_n \in \mathbb{F}_{q^m}$, then build the Goppa code $\Gamma = \Gamma(\alpha_1, \ldots, \alpha_n, g)$ over $\mathbb{F}_q$ and its generator matrix G. Select at random a $k \times k$ invertible matrix S and an $n \times n$ permutation matrix P. Publish the public key $\hat{G} = SGP \in \mathsf{K}_{\mathsf{publ}}$ and store the private key $(S, P, \Gamma) \in \mathsf{K}_{\mathsf{priv}}$.
Enc	On input a public key $\hat{G} \in \mathsf{K}_{\mathsf{publ}}$ and a plaintext $\boldsymbol{m} \in \mathsf{P}$, sample a random error vector $\boldsymbol{e}$ of weight w in $\mathbb{F}_q^n$ and return the ciphertext $\psi = \boldsymbol{m}\hat{G} + \boldsymbol{e} \in \mathsf{C}$.
Dec	On input the private key $(S, P, \Gamma) \in \mathsf{K}_{\mathsf{priv}}$ and a ciphertext $\psi \in \mathsf{C}$, first compute ψP^{-1} then apply the decoding algorithm D_Γ to it. If the decoding succeeds, multiply the output $\hat{\boldsymbol{m}}$ by S^{-1}, and return the resulting plaintext $\phi = \hat{\boldsymbol{m}}S^{-1}$. Otherwise, output $\perp$.

[4]For Goppa codes, given by the support $\alpha_1, \ldots, \alpha_n$ and the Goppa polynomial g.

The original formulation makes use of binary Goppa codes. According to the author, these are chosen mainly for two reasons: they form a large family, providing a vast number of potential public keys, and there exists an efficient, i.e. polynomial-time, algorithm for decoding these codes (e.g. Patterson's algorithm). The scheme can be easily generalized to codes over $\mathbb{F}_q$.

It is easy to see that the decryption process works when the ciphertext is correctly formed. In fact, if $\psi = \boldsymbol{m}\hat{G} + \boldsymbol{e}$ with $\mathsf{wt}(\boldsymbol{e}) = w$, then $\psi P^{-1} = \boldsymbol{m}SG + \boldsymbol{e}P^{-1}$, and since P is a permutation matrix, the vector $\boldsymbol{e}P^{-1}$ has still weight w. It's possible to consider this as the encoding of $\boldsymbol{m}S$ for the code defined by G. The decoding algorithm will succeed returning $\hat{\boldsymbol{m}} = \boldsymbol{m}S$, from which it's easy to recover $\boldsymbol{m}$.

There are two computational assumptions underlying the security of the scheme.

Assumption 1 (Indistinguishability) *The matrix $\hat{G}$ output by* KeyGen *is computationally indistinguishable from a uniformly chosen matrix of the same size.*

Assumption 2 (Decoding hardness) *Decoding a random linear code with parameters n, k, w is hard.*

Note that Assumption **??** is in fact equivalent to assuming the hardness of GDP.