NOTES ON CODE-BASED CRYPTOGRAPHY

GILLES ZÉMOR

1. Background on codes

1.1. **Error-correcting codes, Hamming distance.** Let $\mathbb{F}_q$ denote the finite field on q elements. An *error-correcting code*, in all generality, is a just a subset of $\mathbb{F}_q^n$. A *linear code* is a linear subspace of $\mathbb{F}_q^n$. We will only be dealing with linear codes, so most of the time we will simply refer to them as codes. The elements of a code are called codevectors or codewords.

Definition 1.1. The *Hamming distance* $d(\mathbf{x}, \mathbf{y})$ between any two vectors $\mathbf{x} = [x_1, \ldots, x_n], \mathbf{y} = [y_1, \ldots, y_n]$ of $\mathbb{F}_q^n$ is the number of coordinates where $\mathbf{x}$ and $\mathbf{y}$ differ. $d(\mathbf{x}, \mathbf{y}) = \#\{i, x_i \neq y_i\}$.

The function $d(\cdot, \cdot)$ is indeed a distance, it satisfies the triangular inequality

$$d(\mathbf{x}, \mathbf{y}) \leqslant d(\mathbf{x}, \mathbf{z}) + d(\mathbf{z}, \mathbf{y})$$

and also invariance by translation: $d(\mathbf{x}, \mathbf{y}) = d(\mathbf{x} + \mathbf{z}, \mathbf{y} + \mathbf{z})$. The *weight* of a vector $\mathbf{x}$, denoted by $|\mathbf{x}|$, is its distance to the zero vector, in other words it is the number of its non-zero coordinates. The *support* of $\mathbf{x}$ is the set of positions i of its non-zero coordinates x_i.

Definition 1.2. The *minimum distance* $d_{\min}(C)$ of a linear code C is the smallest distance between two distinct codewords. It is also equal to the smallest weight of a non-zero codeword.

The *parameters* of a code C are denoted by $[n, k, d]$. The number n is the dimension of the ambient space $\mathbb{F}_q^n$ and is called the length, k is the dimension of the code, $k = \dim_{\mathbb{F}_q} C$, and d is the code minimum distance. If it is unclear what is the finite field $\mathbb{F}_q$ over which C is defined, we may write $[n, k, d]_q$.

Decoding problem: it takes as input a vector $\mathbf{y} \in \mathbb{F}_q^n$ and asks for a codeword $\mathbf{c}$ that minimises the distance $d(\mathbf{c}, \mathbf{y})$ to $\mathbf{y}$. There may be several solutions to the decoding problem.

Date: September 2024.

Error correction. Suppose a codeword $\mathbf{c}$ is corrupted so that some of its coordinates are changed. It is converted to some vector $\mathbf{y} = \mathbf{c} + \mathbf{e}$. The vector $\mathbf{e}$ is called the *error vector*. If the weight of the error vector is not too large, solving the decoding problem with input $\mathbf{y}$ recovers the original codeword $\mathbf{c}$.

Theorem 1.3. *If $\mathbf{y} = \mathbf{c} + \mathbf{e}$ where $\mathbf{c}$ is a codeword of the code C, and if the error vector $\mathbf{e}$ has weight $< d_{\min}(C)/2$, then the codeword $\mathbf{c}$ is the unique solution to the decoding problem with input $\mathbf{y}$.*

Proof. Let $\mathbf{c}'$ be a solution to the decoding problem. By the triangular inequality we have

$$d(\mathbf{c}, \mathbf{c}') \leqslant d(\mathbf{c}, \mathbf{y}) + d(\mathbf{c}', \mathbf{y})$$

but $d(\mathbf{c}', \mathbf{y}) \leqslant d(\mathbf{c}, \mathbf{y}) < d_{\min}(C)/2$, so we have $d(\mathbf{c}, \mathbf{c}') < d_{\min}(C)$. Therefore we must have $\mathbf{c}' = \mathbf{c}$. □

Definition 1.4. The matrix $\mathbf{G}$ is said to be a *generator* (or generating) matrix of the code C, if its rows form a basis of C as a vector space. If a generator matrix is of the form $[\mathbf{I}_k \,|\, \mathbf{A}]$, where $\mathbf{I}_k$ is the $k \times k$ identity matrix, it is said to be in *systematic form.*

1.2. **Duality and the syndrome function.**

Dual code. The inner product of two vectors $\mathbf{x} = [x_1, \ldots, x_n]$ and $\mathbf{y} = [y_1, \ldots, y_n]$ is denoted by

$$\langle \mathbf{x}, \mathbf{y} \rangle = x_1 y_1 + \cdots + x_n y_n \in \mathbb{F}_q.$$

The *dual* (or orthogonal) code $C^\perp$ of a code C is defined as

$$C^\perp = \{\mathbf{x} \in \mathbb{F}_q^n,\ \forall \mathbf{c} \in C, \langle \mathbf{x}, \mathbf{c} \rangle = 0\}.$$

Note that if $\mathbf{G}_1, \ldots \mathbf{G}_k$ are the rows of a generator matrix $\mathbf{G}$ for C, then

$$C^\perp = \{\mathbf{x} \in \mathbb{F}_q^n,\ \langle \mathbf{x}, \mathbf{G}_i \rangle = 0,\ i = 1, \ldots, k\}.$$

We have:

Proposition 1.5. *For any code C in $\mathbb{F}_q^n$,*

$$\dim C + \dim C^\perp = n. \tag{1}$$

To find a generator matrix of the dual code $C^\perp$ given a generator matrix of a code C, the following proposition is useful.

Proposition 1.6. *If $[\mathbf{I}_k \,|\, \mathbf{A}]$ is a generator matrix for a code C, then $[-\mathbf{A}^\intercal \,|\, \mathbf{I}_{n-k}]$ is a generator matrix for $C^\perp$.*

Parity-check matrix, syndrome function. A generator matrix $\mathbf{H}$ of the dual code $C^\perp$ of a code C is called a *parity-check matrix* of the code C. Given a parity-check matrix $\mathbf{H}$ of the code C, the associated *syndrome* function is defined as:

$$\begin{aligned} \sigma \ : \mathbb{F}_q^n &\to \mathbb{F}_q^{n-k} \\ \mathbf{x} &\mapsto \sigma(\mathbf{x}) = \mathbf{H}\,\mathbf{x}^\intercal \\ \sigma(\mathbf{x}) &= \begin{bmatrix} \langle \mathbf{H}_1, \mathbf{x} \rangle \\ \cdots \\ \langle \mathbf{H}_{n-k}, \mathbf{x} \rangle \end{bmatrix} \end{aligned}$$

where the $\mathbf{H}_i$s are the rows of $\mathbf{H}$. However, one usually prefers to think of the syndrome function as given by the expression:

$$\sigma(\mathbf{x}) = x_1\mathbf{h}_1 + x_2\mathbf{h}_2 + \cdots + x_n\mathbf{h}_n \tag{2}$$

where $\mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_n$ denote the columns of the matrix $\mathbf{H}$. From the definition of $C^\perp$ and (1), we have $(C^\perp)^\perp = C$, and therefore:

Proposition 1.7. *Let C be a code and σ an associated syndrome function. For $\mathbf{x} \in \mathbb{F}_q^n$, we have $\mathbf{x} \in C$ iff $\sigma(\mathbf{x}) = 0$.*

Dual form of the decoding problem: it takes as input an element $\mathbf{s} \in \mathbb{F}_q^{n-k}$ of the syndrome space, and asks for a vector $\mathbf{e} \in \mathbb{F}_q^n$ of minimum weight such that $\sigma(\mathbf{e}) = \mathbf{s}$. This form of the decoding problem is also referred to as the *syndrome decoding problem.*

Note that the two forms of the decoding problem are really equivalent. If we can solve the syndrome version and we want to find the closest codeword to $\mathbf{y}$, we compute $\mathbf{s} = \sigma(\mathbf{y})$ and then solve the syndrome decoding problem to find $\mathbf{e}$ of minimum weight such that $\sigma(\mathbf{e}) = \mathbf{s}$. Then we set $\mathbf{c} = \mathbf{y} - \mathbf{e}$: since $\sigma(\mathbf{c}) = \sigma(\mathbf{y}) - \sigma(\mathbf{e}) = 0$ Proposition 1.7 implies that $\mathbf{c}$ is a codeword and it must be a solution to the decoding problem. Conversely, if we are given $\mathbf{s} \in \mathbb{F}_q^{n-k}$ and we want to find $\mathbf{e}$ of minimum weight such that $\sigma(\mathbf{e}) = \mathbf{s}$, then we can first find some arbitrary solution $\mathbf{y}$ to the system of linear equations $\sigma(\mathbf{y}) = \mathbf{s}$ (without any weight requirements), which is algorithmically simple linear algebra, and then we solve the decoding problem in its original form to find a codeword $\mathbf{c}$ closest to $\mathbf{y}$, after which $\mathbf{e} = \mathbf{y} - \mathbf{c}$ is the required solution to the syndrome decoding problem with input $\mathbf{s}$.

Example 1.8. The Hamming code of length 7. Let

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

and define C to be the binary code (code over $\mathbb{F}_2$) defined by the parity-check matrix $\mathbf{H}$. Proposition 1.7 together with the expression (2) of the syndrome imply that words of weight $1, 2$ cannot be codewords, since all columns of $\mathbf{H}$ are non-zero and distinct from each other. There are several (how many ?) triples of columns that sum to zero, so the code minimum distance is 3. The parameters of the code are $[7, 4, 3]$. Furthermore, since every non-zero element of $\mathbb{F}_2^3$ is a column of $\mathbf{H}$, we have that every possible input $\mathbf{y} \notin C$ to the decoding problem for C has a solution that is at distance 1 to $\mathbf{y}$.

2. Reed-Solomon codes

2.1. Narrow-sense Reed-Solomon codes.

Definition 2.1. Let $1 \leqslant k \leqslant n \leqslant q$, and let $\alpha_1, \dots \alpha_n$ be distinct elements of $\mathbb{F}_q$. We denote $\boldsymbol{\alpha} = [\alpha_1, \dots, \alpha_n]$. The narrow-sense Reed-Solomon code over $\mathbb{F}_q$ defined by $\boldsymbol{\alpha}$ and k is the set of n-tuples

$$[f(\alpha_1), f(\alpha_2), \dots, f(\alpha_n)]$$

where $f(X)$ ranges over all polynomials of $\mathbb{F}_q[X]$ of degree $< k$.

A Reed-Solomon code C is clearly a linear code. Since a non-zero polynomial of degree $< k \leqslant n$ cannot have n roots, the map

$$f(X) \mapsto [f(\alpha_1), f(\alpha_2), \dots, f(\alpha_n)]$$

is injective, and the dimension of C is that of the space of polynomials of degree $< k$, namely $\dim C = k$. By the same number of roots argument, a non-zero codeword of C has at most $k-1$ zero coordinates, so its weight is at least $n-k+1$. Therefore $d_{\min}(C) \geqslant n - k + 1$. The following theorem tells us that this inequality must actually be an equality.

Theorem 2.2. *(Singleton bound). The parameters $[n, k, d]$ of a code satisfy the inequality*

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

Proof. Consider the linear map

$$\begin{array}{rcl} C & \to & \mathbb{F}_q^{k-1} \\ [x_1, \dots, x_n] & \mapsto & [x_1, \dots, x_{k-1}]. \end{array}$$

It takes a space of dimension k (the code C) to a space of dimension $k-1$, so it has a non-zero kernel. A non-zero vector of the kernel is a codeword with $(k-1)$ zero coordinates, so it has weight $\leqslant n - k + 1$. □

A code satisfying the above Singleton bound is called an *MDS* code. They are optimal, meaning that any code with the same length and dimension cannot have a larger minimum distance. Altogether we have just proved:

Theorem 2.3. *The Reed-Solomon code over $\mathbb{F}_q$ defined by $\boldsymbol{\alpha}$ and k, denoted hereafter by $RS_k(\boldsymbol{\alpha})$, has parameters*

$$[n, k, d = n - k + 1]$$

and is therefore an MDS code.

Let us mention two useful facts on MDS codes.

Proposition 2.4. *Let $\mathbf{G}$ be a $k \times n$ generator matrix of an $[n, k, d]$ code C. The code C is an MDS code iff every $k \times k$ submatrix of $\mathbf{G}$ is non-singular.*

Proof. A vector $\mathbf{c}$ of C has weight $< n-k+1$ iff it has at least k coordinates equal to zero. This happens iff some non-trivial linear combination of the rows of $\mathbf{G}$ is zero on k coordinates. But this means exactly that some $k \times k$ submatrix of $\mathbf{G}$ has some linear combination of its rows equal to zero, i.e. is singular over $\mathbb{F}_q$. □

Theorem 2.5. *The dual of an MDS code is an MDS code.*

Proof. Let C be an MDS code of length n and dimension k with generator matrix $\mathbf{G}$. The dual code $C^\perp$ has therefore $\mathbf{G}$ as a parity-check matrix. Let σ be the associated syndrome function defined by the matrix $\mathbf{G}$. Since every $k \times k$ submatrix of $\mathbf{G}$ is non-singular, every non-trivial linear combination of at most k columns of $\mathbf{G}$ is non-zero. From Proposition 1.7 we therefore have that any codeword of $C^\perp$ must have weight at least $k + 1$, which means that $C^\perp$ is MDS since it has dimension $n - k$. □

Before generalising the definition of Reed-Solomon codes, let us mention:

Proposition 2.6. *The matrix*

$$\mathbf{G} = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ \alpha_1 & \alpha_2 & \cdots & \alpha_n \\ \alpha_1^2 & \alpha_2^2 & \cdots & \alpha_n^2 \\ \cdots & \cdots & \cdots & \cdots \\ \alpha_1^{k-1} & \alpha_2^{k-1} & \cdots & \alpha_n^{k-1} \end{bmatrix}$$

is a generator matrix for the Reed-Solomon code $RS_k(\boldsymbol{\alpha})$.

Proof. The rows of the matrix correspond to the evaluation of the polynomials $1, X, X^2, \ldots, X^{k-1}$. □

2.2. **Generalised Reed-Solomon codes.** We generalise slightly the definition of a Reed-Solomon code. Let $\boldsymbol{\alpha} = [\alpha_1, \ldots, \alpha_n]$ and $\boldsymbol{\beta} = [\beta_1, \ldots, \beta_n]$, where the α_i are *distinct* elements of $\mathbb{F}_q$ as before, and where the β_i are in $\mathbb{F}_q$, but *not necessarily distinct*, and are *non-zero*.

Definition 2.7. The (generalised) Reed-Solomon code over $\mathbb{F}_q$ defined by k, $1 \leqslant k \leqslant n \leqslant q$, and by $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$, denoted by $RS_k(\boldsymbol{\alpha}, \boldsymbol{\beta})$, is the set of n-tuples

$$[\beta_1 f(\alpha_1), \beta_2 f(\alpha_2), \ldots, \beta_n f(\alpha_n)]$$

where $f(X)$ ranges over all polynomials of $\mathbb{F}_q[X]$ of degree $< k$. From now on we refer to generalised Reed-Solomon codes simply as Reed-Solomon codes. Note that the narrow-sense Reed-Solomon code $RS_k(\boldsymbol{\alpha})$ is equal to $RS_k(\boldsymbol{\alpha}, \boldsymbol{\beta})$, with $\boldsymbol{\beta} = (1, 1, \ldots 1)$. All previous arguments carry over straightforwardly and we now have:

Theorem 2.8. *The RS codes $RS_k(\boldsymbol{\alpha}, \boldsymbol{\beta})$ have parameters $[n, k, d = n - k + 1]$.*

Proposition 2.9. *The matrix*

$$\mathbf{G} = \begin{bmatrix} \beta_1 & \beta_2 & \cdots & \beta_n \\ \beta_1\alpha_1 & \beta_2\alpha_2 & \cdots & \beta_n\alpha_n \\ \beta_1\alpha_1^2 & \beta_2\alpha_2^2 & \cdots & \beta_n\alpha_n^2 \\ \cdots & \cdots & \cdots & \cdots \\ \beta_1\alpha_1^{k-1} & \beta_2\alpha_2^{k-1} & \cdots & \beta_n\alpha_n^{k-1} \end{bmatrix}$$

is a generator matrix for the Reed-Solomon code $RS_k(\boldsymbol{\alpha}, \boldsymbol{\beta})$.

The following notion will be useful to find the dual of an RS code.

2.3. **Star products.** If $\mathbf{x} = [x_1, \ldots, x_n], \mathbf{y} = [y_1, \ldots, y_n] \in \mathbb{F}_q^n$, let us define the coordinate-wise product of $\mathbf{x}$ and $\mathbf{y}$:

$$\mathbf{x} * \mathbf{y} = [x_1 y_1, x_2 y_2, \ldots, x_n y_n].$$

To lighten notation, instead of $\mathbf{x} * \mathbf{y}$ we will simply write $\mathbf{xy}$, which, in our context, is unlikely to be confused with some other product structure. Notice that we have:

$$\langle \mathbf{x}, \mathbf{y} \rangle = \langle \mathbf{xy}, \mathbf{1} \rangle$$

where $\mathbf{1}$ denotes the all-one vector $\mathbf{1} = [1, 1, \ldots, 1]$. More generally we have

$$\langle \mathbf{xz}, \mathbf{y} \rangle = \langle \mathbf{x}, \mathbf{yz} \rangle. \tag{3}$$

Remark 2.10. Notice that the generator matrix $\mathbf{G}$ of the RS code $RS_k(\boldsymbol{\alpha}, \boldsymbol{\beta})$ given by Proposition 2.9 has rows that may be conveniently thought of as the geometric progression $\boldsymbol{\beta}, \boldsymbol{\beta}\boldsymbol{\alpha}, \boldsymbol{\beta}\boldsymbol{\alpha}^2, \ldots, \boldsymbol{\beta}\boldsymbol{\alpha}^{k-1}$.

Definition 2.11. If B and C are two codes of length n over $\mathbb{F}_q$, we define their star product (also called Hadamard product, or Schur product) $B * C$ as the *linear span* of all products $\mathbf{bc}$, for all $\mathbf{b} \in B$, $\mathbf{c} \in C$.

Again we will mostly write BC instead of $B * C$ and, similarly, use the shorthand C^2 to denote $C * C$. Context should dispel any confusion with Cartesian products.

Remark 2.12. If B and C are two codes of length n and dimensions k and ℓ respectively, and if $\mathbf{b}_1, \ldots, \mathbf{b}_k$ and $\mathbf{c}_1, \ldots, \mathbf{c}_\ell$ are bases of B and C respectively, then the code BC is generated by the set of products $\mathbf{b}_i\mathbf{c}_j$ of basis vectors, $i = 1, \ldots, k$ $j = 1, \ldots, \ell$. In particular we have

$$\dim B * C \leqslant \dim B \dim C$$

though the actual dimension of $B * C$ may sometimes be much smaller.

Finally, notice that we have:

Proposition 2.13. *If $B = RS_k(\boldsymbol{\alpha}, \boldsymbol{\beta})$ and $C = RS_\ell(\boldsymbol{\alpha}, \boldsymbol{\gamma})$, then*

$$B * C = BC = RS_{k+\ell-1}(\boldsymbol{\alpha}, \boldsymbol{\beta}\boldsymbol{\gamma})$$

(with the convention that if $k+\ell-1 > n$, then $RS_{k+\ell-1}(\quad)$ is defined as the whole space).

2.4. **The dual of a Reed-Solomon code.**

Theorem 2.14. *The dual of a Reed-Solomon code $RS_k(\boldsymbol{\alpha}, \boldsymbol{\beta})$ is a Reed-Solomon code $RS_{n-k}(\boldsymbol{\alpha}, \boldsymbol{\gamma})$ for some vector $\boldsymbol{\gamma} = [\gamma_1, \ldots, \gamma_n]$.*

Proof. Let $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ be fixed. Consider the Reed-Solomon code $C_{n-1} = RS_{n-1}(\boldsymbol{\alpha}, \boldsymbol{\beta})$ of dimension $n-1$. Let us set $\boldsymbol{\gamma}$ to be a non-zero codeword of the dual code $C_{n-1}^\perp$. Since C_{n-1} has dimension $n-1$, the vector $\boldsymbol{\gamma}$ is uniquely defined up to multiplication by a non-zero element of $\mathbb{F}_q$. Since C_{n-1} is MDS its dual is MDS (Theorem 2.5), which means that $\boldsymbol{\gamma}$ has weight n, i.e. all the γ_i are non-zero. It makes sense therefore to talk about Reed-Solomon codes $RS_k(\boldsymbol{\alpha}, \boldsymbol{\gamma})$ and our choice of $\boldsymbol{\gamma}$ means that we have $RS_{n-1}(\boldsymbol{\alpha}, \boldsymbol{\beta})^\perp = RS_1(\boldsymbol{\alpha}, \boldsymbol{\gamma})$. Equivalently, from Remark 2.10 we have

$$\langle \boldsymbol{\beta}\boldsymbol{\alpha}^i, \boldsymbol{\gamma} \rangle = 0 \quad \text{for all } i = 0 \ldots n-2.$$

Applying (3) to the above for $i = 1, \ldots n-2$, we obtain

$$\langle \boldsymbol{\beta}\boldsymbol{\alpha}^i, \boldsymbol{\gamma}\boldsymbol{\alpha} \rangle = 0 \quad \text{for all } i = 0 \ldots n-3.$$

We therefore have that both $\boldsymbol{\gamma}$ and $\boldsymbol{\gamma}\boldsymbol{\alpha}$ are orthogonal to $RS_{n-2}(\boldsymbol{\alpha}, \boldsymbol{\beta})$, meaning that $RS_{n-2}(\boldsymbol{\alpha}, \boldsymbol{\beta})^\perp = RS_2(\boldsymbol{\alpha}, \boldsymbol{\gamma})$. Continuing in this way we obtain that $RS_k(\boldsymbol{\alpha}, \boldsymbol{\beta})^\perp = RS_{n-k}(\boldsymbol{\alpha}, \boldsymbol{\gamma})$ for every k. □

Computing $\boldsymbol{\gamma}$ from $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ is straightforward linear algebra (Gaussian elimination). It may be tedious by hand though, but is simple in some cases, for example:

– Exercice 1. *Let $n = q-1$ and $\alpha_1, \ldots, \alpha_n$ be the non-zero elements of $\mathbb{F}_q$.*

(1) Show that $\alpha_1 + \cdots + \alpha_n = 0$.
(2) Show that $\alpha_1^i + \cdots + \alpha_n^i = -1$ if $i = 0 \bmod n$ and $\alpha_1^i + \cdots + \alpha_n^i = 0$ otherwise.

(3) Show that if $\boldsymbol{\beta} = \mathbf{1}$, then $\boldsymbol{\gamma} = \boldsymbol{\alpha}$. More generally, show that we have $\boldsymbol{\gamma} = \boldsymbol{\alpha}\boldsymbol{\beta}^{-1}$ (all coordinates of $\boldsymbol{\beta}$ are non-zero, so it is invertible for the star product).

2.5. **Decoding Reed-Solomon codes.** Let C be the $RS_k(\boldsymbol{\alpha}, \boldsymbol{\beta})$ Reed-Solomon code over $\mathbb{F}_q$, for some $\boldsymbol{\alpha}, \boldsymbol{\beta}$. Let $\mathbf{c}$ be a codeword and suppose that it is corrupted in t positions, so that we are given a vector $\mathbf{y} \in \mathbb{F}_q^n$, such that $\mathbf{y} = \mathbf{c} + \mathbf{e}$ with $|\mathbf{e}| = t$. We suppose furthermore that $t < d_{\min}/2$, equivalently $t \leqslant (n-k)/2$. The goal is to recover $\mathbf{c}$ with an algorithm of reasonable complexity.

The key to the decoding algorithm is to introduce an auxiliary Reed-Solomon code, which is $L = RS_{t+1}(\boldsymbol{\alpha}, \mathbf{1}) = RS_{t+1}(\boldsymbol{\alpha})$ and is called the *error-locator* code (or simply locator code). Note from Proposition 2.13 that the star product $\Pi = CL$ is the Reed-Solomon code $RS_{k+t}(\boldsymbol{\alpha}, \boldsymbol{\beta})$.

Note that there exists a non-zero codeword $\boldsymbol{\ell}$ of L that is zero on the t positions in error, i.e. the support of $\mathbf{e}$. This is simply because L has been chosen of dimension $t+1$. Alternatively, we can think of such a codeword as being defined by a polynomial $f(X)$ of degree t that is zero on the values α_i that define the support of $\mathbf{e}$. For such a vector $\boldsymbol{\ell}$ we therefore have, from $\mathbf{y} = \mathbf{c} + \mathbf{e}$ and $\mathbf{e}\boldsymbol{\ell} = 0$,

$$\mathbf{y}\boldsymbol{\ell} = \mathbf{c}\boldsymbol{\ell} + \mathbf{e}\boldsymbol{\ell} = \mathbf{c}\boldsymbol{\ell}.$$

Therefore, by definition of the star product, we have $\mathbf{y}\boldsymbol{\ell} \in \Pi$. So the first step of the decoding procedure is to *localise the errors.* That means find $\boldsymbol{\ell} \in L$ of weight $n - t$ such that $\mathbf{y}\boldsymbol{\ell} = 0$. We now know that such a $\boldsymbol{\ell}$ is a solution of $\mathbf{y}\boldsymbol{\ell} \in \Pi$. This is simply a linear system. Concretely, we compute $\mathbf{y}\boldsymbol{\ell}_0, \mathbf{y}\boldsymbol{\ell}_1, \ldots, \mathbf{y}\boldsymbol{\ell}_t$ where $\boldsymbol{\ell}_0, \boldsymbol{\ell}_1, \ldots, \boldsymbol{\ell}_t$ is a fixed basis of the code L. We write

$$\boldsymbol{\ell} = \lambda_0\boldsymbol{\ell}_0 + \lambda_1\boldsymbol{\ell}_1 + \cdots + \lambda_t\boldsymbol{\ell}_t \tag{4}$$

and define the syndrome function σ for the code Π (by means of a parity-check matrix of Π that can be precomputed). The condition $\mathbf{y}\boldsymbol{\ell} \in \Pi$ gets therefore rewritten as:

$$\sigma(\mathbf{y}\boldsymbol{\ell}) = \lambda_0\sigma(\mathbf{y}\boldsymbol{\ell}_0) + \lambda_1\sigma(\mathbf{y}\boldsymbol{\ell}_1) + \cdots + \lambda_t\sigma(\mathbf{y}\boldsymbol{\ell}_t) = 0.$$

So we solve this linear system in the variables $\lambda_0, \lambda_1, \ldots, \lambda_t$ which gives the required value of $\boldsymbol{\ell}$ defined by (4).

Now we know that among the solutions to the above linear system there must be one ℓ that is zero on the positions in error. But could there be other "parasite" solutions ? The answer is no, because if $\mathbf{y}\boldsymbol{\ell} \in \Pi$, then from $\mathbf{y} = \mathbf{c} + \mathbf{e}$ and $\mathbf{c}\boldsymbol{\ell} \in \Pi$ we have $\mathbf{e}\boldsymbol{\ell} \in \Pi$. But Π is a Reed-Solomon code of dimension $k + t$, and therefore of minimum distance $n - k - t + 1$ which is greater than t by our assumption $t \leqslant (n-k)/2$. So because $|\mathbf{e}\boldsymbol{\ell}| \leqslant |\mathbf{e}| = t$, it must be the case that $\mathbf{e}\boldsymbol{\ell} = 0$. Therefore, any non-zero solution $\boldsymbol{\ell}$ to $\mathbf{y}\boldsymbol{\ell} \in \Pi$ will be a vector with exactly t zero

coordinates (there can't be more because the minimum distance of the RS code L is $n-t$), which correspond exactly to the positions in error.

Summarising:

Decoding algorithm for $C = RS_k(\boldsymbol{\alpha}, \boldsymbol{\beta})$.

Input: $\mathbf{y} = \mathbf{c} + \mathbf{e}$, $|\mathbf{e}| = t$.

Set $\boldsymbol{\ell}_0 = \mathbf{1}, \boldsymbol{\ell}_1 = \boldsymbol{\alpha}, \ldots, \boldsymbol{\ell}_t = \boldsymbol{\alpha}^t$.

Define a syndrome function σ for the code $\Pi = RS_{k+t}(\boldsymbol{\alpha}, \boldsymbol{\beta})$.

Compute $\sigma(\mathbf{y}\boldsymbol{\ell}_0), \sigma(\mathbf{y}\boldsymbol{\ell}_1), \ldots, \sigma(\mathbf{y}\boldsymbol{\ell}_t)$.

Solve the linear system $\lambda_0\sigma(\mathbf{y}\boldsymbol{\ell}_0) + \lambda_1\sigma(\mathbf{y}\boldsymbol{\ell}_1) + \cdots + \lambda_t\sigma(\mathbf{y}\boldsymbol{\ell}_t) = 0$ in the indeterminates $\lambda_i \in \mathbb{F}_q$ and choose any non-zero solution.

Compute the codeword $\boldsymbol{\ell} = [\ell_1, \ell_2, \ldots, \ell_n] = \lambda_0\boldsymbol{\ell}_0 + \lambda_1\boldsymbol{\ell}_1 + \cdots + \lambda_t\boldsymbol{\ell}_t$ of L.

For every i such that $\ell_i \neq 0$, declare the position i to be error-free, i.e. set $c_i = y_i$.

Let $E = \{i, \ell_i = 0\}$. This is the set of positions in error. To recover the missing coordinates $c_i, i \in E$, solve the linear system in the indeterminates c_i

$$\sum_{i \in E} c_i \mathbf{h}_i + \sum_{i \notin E} y_i \mathbf{h}_i = 0$$

where $\mathbf{h}_1, \ldots, \mathbf{h}_n$ are the columns of a parity-check matrix $\mathbf{H}$ of C.

Output $\mathbf{c} = [c_1, \ldots, c_n]$.

Remark 2.15. For the above algorithm we have implicitly supposed that the number t of errors is known to the decoder, which will be the typical cryptography setting. However it may often be that t is only known to be an upper bound on the actual number of errors. In this case the algorithm runs without any changes, and the same arguments prove that it yields the correct result. What will happen is that there will be more available solutions $\lambda_0, \ldots, \lambda_t$ to the linear system, but any non-zero solution will still work: it will also be possible that $|E| < t$, and that some values of $i \in E$ will turn out to also be error-free coordinates for $\mathbf{c}$, i.e. $c_i = y_i$.

3. Goppa codes

3.1. **Alternant codes.** Suppose that we want codes over $\mathbb{F}_q$ of length n significantly larger than q. Then the Reed-Solomon construction does not work because it is restricted to $n \leqslant q$ (actually the Reed-Solomon construction generalises to $n = q + 1$, but this makes little difference). This is typically the case for small

values of q, notably for $q = 2$. We don't necessarily have to forget about the Reed-Solomon construction altogether though: what we can do is take a Reed-Solomon code C of length n over an extension field $\mathbb{F}_{q^m}$ of $\mathbb{F}_q$, and then consider the subcode of C made up of those codevectors whose coordinates all fall into $\mathbb{F}_q$. This is sometimes called a *subfield subcode* construction. Applied to Reed-Solomon codes, this gives us the class of so-called *alternant codes.*

Definition 3.1. An *alternant code* over $\mathbb{F}_q$ is the set of codewords of a Reed-Solomon code over $\mathbb{F}_{q^m}$, for some $m > 1$, whose coordinates all belong to $\mathbb{F}_q$.

An alternant code is usually specified by a parity-check matrix of the underlying Reed-Solomon code, so an $r \times n$ matrix $\mathbf{H}$ over $\mathbb{F}_{q^m}$ of the form

$$\begin{bmatrix} \boldsymbol{\gamma} \\ \boldsymbol{\gamma\alpha} \\ \cdots \\ \boldsymbol{\gamma\alpha}^{r-1} \end{bmatrix}$$

with $\boldsymbol{\alpha} = [\alpha_1, \ldots, \alpha_n]$ and $\boldsymbol{\gamma} = [\gamma_1, \ldots, \gamma_n]$, $\alpha_i, \gamma_i \in \mathbb{F}_{q^m} \setminus \{0\}$. The alternant code is then

$$C = \{\mathbf{c} \in \mathbb{F}_q^n, \mathbf{Hc}^\intercal = 0\}.$$

Decomposing the entries of $\mathbf{H}$ over an $\mathbb{F}_q$-basis of $\mathbb{F}_{q^m}$, we get that the r rows of $\mathbf{H}$ become rm rows over $\mathbb{F}_q$, in other words C becomes defined by rm linear equations over $\mathbb{F}_q$. We therefore have:

Proposition 3.2. *The alternant code defined by the $r \times n$ matrix $\mathbf{H}$ over $\mathbb{F}_{q^m}$ is an $\mathbb{F}_q$-linear code of dimension $k \geqslant n - rm$.*

In all generality we only have an inequality for the dimension k because we have no guarantee that the rm linear equations over $\mathbb{F}_q$ will be linearly independent.

Since the alternant code defined by $\mathbf{H}$ is contained in a Reed-Solomon code, its minimum distance is at least that of the RS code. In other words:

Proposition 3.3. *The alternant code defined by the $r \times n$ matrix $\mathbf{H}$ over $\mathbb{F}_{q^m}$ is an $\mathbb{F}_q$-linear code of minimum distance $d \geqslant r + 1$.*

Finally, alternant codes come with a natural decoding algorithm, they can be decoded simply by solving the decoding problem for the ambient Reed-Solomon code. If we are decoding $\mathbf{y} = \mathbf{c} + \mathbf{e}$ with $|\mathbf{e}| < (r+1)/2$, then we know that $\mathbf{c}$ is the unique solution to the decoding problem with input $\mathbf{y}$ and that the Reed-Solomon decoder will find it.

3.2. **Goppa codes, a first definition.** Goppa codes are a particular subclass of the family of alternant codes, though that is not immediately apparent from the construction below.

Let q be fixed and set $Q = q^m$. Let $\alpha_1, \ldots, \alpha_n$ be some fixed, distinct elements of $\mathbb{F}_Q$ and let $G(X) \in \mathbb{F}_Q[X]$ be some polynomial of degree r such that $G(\alpha_i) \neq 0$ for $i = 1, \ldots, n$. Now for any vector $\mathbf{a} = (a_1, \ldots, a_n)$ *over* $\mathbb{F}_q$, we define the rational function in the variable X,

$$R_{\mathbf{a}}(X) = \sum_{i=1}^{n} \frac{a_i}{X - \alpha_i}. \tag{5}$$

Note that the $(X - \alpha_i)$ are invertible modulo $G(X)$ since we have chosen $G(X)$ such that $G(\alpha_i) \neq 0$. We may therefore consider the quantity $R_{\mathbf{a}}(X)$ in the quotient ring $\mathbb{F}_Q[X]/G(X)$. We define:

Definition 3.4. For $\boldsymbol{\alpha} = [\alpha_1, \ldots, \alpha_n]$ and $G(X)$, the Goppa code $\Gamma(\boldsymbol{\alpha}, G)$ consists of all vectors $\mathbf{a} \in \mathbb{F}_q^n$ such that

$$R_{\mathbf{a}}(X) = 0$$

in the ring $\mathbb{F}_Q[X]/G(X)$.

It should be obvious that Γ defined above is a linear code over $\mathbb{F}_q$. We now compute a convenient parity-check matrix (over $\mathbb{F}_Q$) for Γ.

3.3. An $\mathbb{F}_Q$-parity-check matrix for Goppa codes.

Theorem 3.5. *Let $\boldsymbol{\alpha}$ and $G(X)$ define the Goppa code $\Gamma = \Gamma(\boldsymbol{\alpha}, G)$. Let*

$$\mathbf{H} = \begin{bmatrix} \boldsymbol{\gamma} \\ \boldsymbol{\gamma}\boldsymbol{\alpha} \\ \cdots \\ \boldsymbol{\gamma}\boldsymbol{\alpha}^{r-1} \end{bmatrix}$$

with $\boldsymbol{\gamma} = [G(\alpha_1)^{-1}, G(\alpha_2)^{-1}, \ldots, G(\alpha_n)^{-1}]$. Then $\Gamma = \{\mathbf{a} \in \mathbb{F}_q^n, \mathbf{H}\mathbf{a}^\intercal = 0\}$. In particular Γ is an alternant code.

Proof. First we express the inverse of $(X - \alpha_i)$ modulo $G(X)$ as a polynomial of degree $< r$. For this we need only write:

$$(X - \alpha_i)^{-1} = -\frac{G(X) - G(\alpha_i)}{X - \alpha_i} G(\alpha_i)^{-1},$$

since it is easily checked that multiplying the right-hand side by $(X - \alpha_i)$ gives 1. So by definition of Γ, $\mathbf{a} \in \Gamma$ iff

$$\sum_{i=1}^{n} a_i \frac{G(X) - G(\alpha_i)}{X - \alpha_i} G(\alpha_i)^{-1} = 0 \tag{6}$$

in $\mathbb{F}_Q[X]$, since the left-handside of (6) is a polynomial of degree $< r$. Writing $G(X) = g_r X^r + g_{r-1} X^{r-1} + \cdots + g_1 X + g_0$, where $g_i \in \mathbb{F}_Q$ and $g_r \neq 0$, we apply

the formula

$$\frac{X^j - \alpha_i^j}{X - \alpha_i} = X^{j-1} + \alpha_i X^{j-2} + \cdots + \alpha_i^{j-1}$$

with the purpose of making the left-hand side of (6) more explicit, to obtain:

$$\frac{G(X) - G(\alpha_i)}{X - \alpha_i} = g_r(X^{r-1} + \alpha_i X^{r-2} + \cdots + \alpha_i^{r-1}) + \cdots + g_2(X + \alpha_i) + g_1$$

Equating the coefficients of $X^{r-1}, \ldots, X, 1$ to zero in (6), we get that $\mathbf{a} \in \Gamma$ iff $\mathbf{M}\mathbf{a}^\intercal = 0$ with

$$\mathbf{M} = \begin{bmatrix} g_r G(\alpha_1)^{-1} & \cdots & g_r G(\alpha_n)^{-1} \\ (g_{r-1} + \alpha_1 g_r) G(\alpha_1)^{-1} & \cdots & (g_{r-1} + \alpha_n g_r) G(\alpha_n)^{-1} \\ \cdots & \cdots & \cdots \\ (g_1 + \alpha_1 g_2 + \cdots + \alpha_1^{r-1} g_r) G(\alpha_1)^{-1} & \cdots & (g_1 + \alpha_n g_2 + \cdots + \alpha_n^{r-1} g_r) G(\alpha_n)^{-1} \end{bmatrix}$$

Now $\mathbf{M}$ gets rewritten as:

$$\mathbf{M} = \begin{bmatrix} g_r & 0 & 0 & \cdots & 0 \\ g_{r-1} & g_r & 0 & \cdots & 0 \\ g_{r-2} & g_{r-1} & g_r & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ g_1 & g_2 & g_3 & \cdots & g_r \end{bmatrix} \begin{bmatrix} 1 & 1 & \cdots & 1 \\ \alpha_1 & \alpha_2 & \cdots & \alpha_n \\ \alpha_1^2 & \alpha_2^2 & \cdots & \alpha_n^2 \\ \cdots & \cdots & \cdots & \cdots \\ \alpha_1^{r-1} & \alpha_2^{r-1} & \cdots & \alpha_n^{r-1} \end{bmatrix} \mathbf{D}$$

with $\mathbf{D}$ being the diagonal matrix

$$\mathbf{D} = \begin{bmatrix} G(\alpha_1)^{-1} & & & 0 \\ & G(\alpha_2)^{-1} & & \\ & & \ddots & \\ 0 & & & G(\alpha_n)^{-1} \end{bmatrix}.$$

Since the leftmost matrix in the decomposition of $\mathbf{M}$ is square and invertible, we have that $\mathbf{M}\mathbf{a}^\intercal = 0$ iff $\mathbf{H}\mathbf{a}^\intercal = 0$ where

$$\mathbf{H} = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ \alpha_1 & \alpha_2 & \cdots & \alpha_n \\ \alpha_1^2 & \alpha_2^2 & \cdots & \alpha_n^2 \\ \cdots & \cdots & \cdots & \cdots \\ \alpha_1^{r-1} & \alpha_2^{r-1} & \cdots & \alpha_n^{r-1} \end{bmatrix} \mathbf{D}$$

which is exactly the parity-check matrix claimed by the Theorem. □

So why the convoluted definition of Section 3.2 if they can be defined by the characterisation of Theorem 3.5 ? Because the original definition allows us to sometimes obtain information that would be difficult to obtain from the parity-check matrix of Theorem 3.5. We now illustrate.

3.4. **Binary Goppa codes.** Let $q = 2$ and $Q = 2^m$. Define a Goppa code $\Gamma = \Gamma(\boldsymbol{\alpha}, G)$ and suppose $G(X)$ is chosen to be squarefree, meaning that for every irreducible polynomial $P(X)$ of $\mathbb{F}_Q[X]$, we have $P(X)^2 \not| G(X)$. Let $\mathbf{a} = [a_1, \ldots, a_n]$ be a (binary) codeword in Γ and let $I_{\mathbf{a}} = \{i, a_i = 1\}$ be the support of $\mathbf{a}$. Define the polynomial:

$$f_{\mathbf{a}}(X) = \prod_{i \in I_{\mathbf{a}}} (X - \alpha_i)$$

We have:

$$f'_{\mathbf{a}}(X) = \sum_{i \in I_{\mathbf{a}}} \prod_{\substack{j \in I_{\mathbf{a}} \\ j \neq i}} (X - \alpha_j)$$

and (5) gets rewritten as:

$$R_{\mathbf{a}}(X) = \sum_{i \in I_{\mathbf{a}}} \frac{1}{X - \alpha_i} = \frac{f'_{\mathbf{a}}(X)}{f_{\mathbf{a}}(X)}$$

Since $G(\alpha_i) \neq 0$ for every i, we have that $f_{\mathbf{a}}(X)$ is invertible modulo $G(X)$, so

$$R_{\mathbf{a}}(X) = 0 \bmod G(X) \quad \text{iff} \quad G(X) | f'_{\mathbf{a}}(X).$$

Now the field $\mathbb{F}_Q$ is of charateristic 2, so $f'_{\mathbf{a}}(X)$ is a sum of monomials of even degree, so it is a sum of squares (recall that every element of $\mathbb{F}_Q$ is a square), hence it is a square in $\mathbb{F}_Q[X]$. Therefore $G(X) | f'_{\mathbf{a}}(X)$ iff $G(X)^2 | f'_{\mathbf{a}}(X)$. Summarising:

Proposition 3.6. *If the Goppa polynomial $G(X)$ is squarefree, then, over the binary field we have:*

$$\Gamma = \Gamma(\boldsymbol{\alpha}, G(X)) = \Gamma(\boldsymbol{\alpha}, G(X)^2).$$

So, setting $r = \deg G(X)$, if we think of Γ as defined by $G(X)$, then Proposition 3.6 tells us that the minimum distance of Γ is at least $2r + 1$ as opposed to $r + 1$ guaranteed by Proposition 3.3. Alternatively, if we think of Γ as defined by $G(X)^2$, then Proposition 3.6 tells us that the dimension of Γ is at least $n - rm$ as opposed to $n - 2rm$ guaranteed by Proposition 3.2.

To decode Γ we should use its structure given by $\Gamma(\boldsymbol{\alpha}, G(X)^2)$, since Reed-Solomon decoding is then guaranteed to decode correctly up to r errors.

4. The original McEliece cryptosystem

4.1. **The McEliece paradigm.** McEliece proposed a very general method for devising a *public-key* encryption scheme. Let $\mathcal{M} = \mathbb{F}_2^k$ be the space of plain messages that we want to encrypt, in other words we assume our plaintexts to be k-bit strings. Choose an error-correcting code C of length n and dimension k and

publish an arbitrary generator matrix $\mathbf{G}$ for this code. The matrix $\mathbf{G}$ is the public key and is used for encryption. To encrypt a message $\mathbf{m} \in \mathbb{F}_2^k$, we compute

$$\mathbf{y} = \mathbf{mG} + \mathbf{e}$$

where $\mathbf{e}$ is a *random* binary vector of length n and some fixed weight t. Notice that $\mathbf{mG}$ is a codevector for the code C, so the ciphertext $\mathbf{y}$ is a noisy version of a codeword.

The secret key gives access to a hidden decoding algorithm that is able to efficiently remove error vectors of weight t. So to decipher the ciphertext $\mathbf{y}$, apply the decoding algorithm to remove the error $\mathbf{e}$ and obtain $\mathbf{mG}$. Writing $\mathbf{G} = [\mathbf{A} \mid \mathbf{B}]$ and assuming, without loss of generality, $\mathbf{A}$ to be invertible, we have $\mathbf{mG} = [\mathbf{mA} \mid \mathbf{mB}]$ and we may multiply on the right $\mathbf{mA}$ by $\mathbf{A}^{-1}$ to recover $\mathbf{m}$.

The working assumption is that without knowledge of the hidden data that allows one to decode noisy codewords of C, the adversary that tries to decrypt is faced with a generic instance of the decoding problem, which is generally assumed to be intractable (more on that later).

The assumption for a McEliece cryptosystem is therefore generally formulated as *indistinguishability* from a random code. To make this slightly more formal, what we require is a family $\mathcal{F}$ of codes C, of length n and dimension k say, that come with a low-complexity decoding algorithm from errors of weight t. We assume:

Indistinguishability assumption. Consider the following two ways of constructing a matrix G :

(1) Choose a random code C from the family $\mathcal{F}$, and choose a random $k \times n$ generator matrix G for C,
(2) Choose a uniformly random $k \times n$ matrix G.

There should not exist an algorithm with complexity less than some security parameter (e.g. that uses less than 2^{80} arithmetic operations), that given as input a matrix G that has been randomly constructed either according to method 1 or method 2, outputs (1) or (2) with a success probability significantly better than $1/2$ (a random guess).

We now remark that the indistinguishibility assumption implies that deciphering random encrypted messages is not feasible without the secret or some extra knowledge. Indeed, if there were an algorithm that decrypts, then, by definition of the McEliece scheme, such an algorithm decodes from t errors random codewords of a member C of $\mathcal{F}$; the indistinguishibility assumption implies therefore that this algorithm also decodes from t errors random codewords of a *random code* (because if the algorithm behaves differently on a random code, then it is a distinguisher !)

If we are convinced that decoding random codes of length n and dimension k from t errors is infeasible, then the indistinguishibility assumption implies security. We

could envisage having security without indistinguishibility, but cryptographers like to err on the side of caution.

4.2. **The original McEliece cryptosystem.** How can we implement the above idea pratically ? McEliece's proposal was to use binary Goppa codes $\Gamma(\boldsymbol{\alpha}, G(X))$. So the parameters of the Goppa code are chosen, namely m, the degree of the extension field $\mathbb{F}_Q$ of $\mathbb{F}_2$, the length n, and the degree r of the Goppa polynomial. Then a random vector $\boldsymbol{\alpha} = [\alpha_1, \ldots, \alpha_n]$, with distinct $\alpha_i \in \mathbb{F}_Q$, is chosen. Finally one chooses a random squarefree polynomial $G(X) \in \mathbb{F}_Q[X]$ of degree r that does not evaluate to 0 on any of the α_i. the public-key is an arbitrary (random) generator matrix G for the code C. Everything else about the code is kept secret.

We recall that the matrix $\mathbf{H}$ in Theorem 3.5 defines a Reed-Solomon code and that its dual is a Reed-Solomon code $RS_k(\boldsymbol{\alpha}, \boldsymbol{\beta})$ with $k = n - r$. We can think of $RS_k(\boldsymbol{\alpha}, \boldsymbol{\beta})$ is the *parent* Reed-Solomon code of the Goppa code $\Gamma(\boldsymbol{\alpha}, G(X)$, i.e. $\Gamma(\boldsymbol{\alpha}, G(X))$ is the set of binary vectors that belong to $RS_k(\boldsymbol{\alpha}, \boldsymbol{\beta})$. The secret key consists of therefore of $\boldsymbol{\alpha}, \boldsymbol{\beta}$ which enables us to decode both the parent Reed-Solomon code and the Goppa code $\Gamma(\boldsymbol{\alpha}, G(X))$ by applying the decoding algorithm of Section 2.5. Summarising:

The original McEliece cryptosystem Choose parameters m, n, r. Choose $\boldsymbol{\alpha} = [\alpha_1, \alpha_n]$, where the α_i are distinct elements of $\mathbb{F}_Q$, $Q = 2^m$. Choose a random squarefree polynomial $G(X) \in \mathbb{F}_Q[X]$ of degree r. Define the binary Goppa code $C = \Gamma(\boldsymbol{\alpha}, G(X))$.

Public key: a random generator matrix G of C

Secret key: $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ that define the Reed-Solomon code $RS_{n-r}(\boldsymbol{\alpha}, \boldsymbol{\beta})$ parent to C.

Encryption: for a message $\mathbf{m} \in \mathbb{F}_2^k$, construct the ciphertext as

$$\mathbf{y} = \mathbf{mG} + \mathbf{e}$$

where $\mathbf{e} \in \mathbb{F}_2^n$ is a random binary vector of weight r.

Decryption: Use $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ to apply the decoding algorithm of Section 2.5 and recover $\mathbf{c} = \mathbf{mG}$ from $\mathbf{y}$. Recover $\mathbf{m}$ from $\mathbf{mG}$ by linear algebra (solve a linear system).

Remark 4.1. One doesn't really need the knowledge of $\boldsymbol{\beta}$ to decode, the vector $\boldsymbol{\alpha}$ is sufficient. Indeed, in the decoding algorithm of Section 2.5 one can define the code Π as the star-product of L with the code over $\mathbb{F}_Q$ generated by the matrix G, which will be a subcode of (and probably simply equal to) the product of L with the parent Reed-Solomon code $RS_{n-r}(\boldsymbol{\alpha}, \beta)$.

McEliece originally proposed to use the parameters $m = 10$, $n = 1024$ and $r = 50$. So the Goppa code C has parameters $[1024, k \geqslant 524, d \geqslant 101]$ and the decoding algorithm allows the decoding of any pattern of 50 errors. In practice the dimension k of the code actually equals 524.