Chapter 5

Generalized Reed-Solomon Codes

In 1960, I.S. Reed and G. Solomon introduced a family of error-correcting codes that are doubly blessed. The codes and their generalizations are useful in practice, and the mathematics that lies behind them is interesting. In the first section we give the basic properties and structure of the generalized Reed-Solomon codes, and in the second section we describe in detail one method of algebraic decoding that is quite efficient.

5.1 Basics

Let F be a field and choose nonzero elements $v_1, \dots, v_n \in F$ and distinct elements $\alpha_1, \dots, \alpha_n \in F$. Set $\mathbf{v} = (v_1, \dots, v_n)$ and $\boldsymbol{\alpha} = (\alpha_1, \dots, \alpha_n)$. For $0 \le k \le n$ we define the *generalized Reed-Solomon codes*

generalized Reed-Solomon codes

$$\mathrm{GRS}_{n,k}(\boldsymbol{\alpha}, \mathbf{v}) = \{ \ (v_1 f(\alpha_1), v_2 f(\alpha_2), \dots, v_n f(\alpha_n)) \mid f(x) \in F[x]_k \ \} \ .$$

Here we write $F[x]_k$ for the set of polynomial in $F[x]$ of degree less than k, a vector space of dimension k over F. For fixed n, $\boldsymbol{\alpha}$, and $\mathbf{v}$, the various *GRS* codes enjoy the nice embedding property $\mathrm{GRS}_{n,k-1}(\boldsymbol{\alpha}, \mathbf{v}) \le \mathrm{GRS}_{n,k}(\boldsymbol{\alpha}, \mathbf{v})$.

If $f(x)$ is a polynomial, then we shall usually write $\mathbf{f}$ for its associated codeword. This codeword also depends upon $\boldsymbol{\alpha}$ and $\mathbf{v}$; so at times we prefer to write unambiguously

$$\mathbf{ev}_{\boldsymbol{\alpha},\mathbf{v}}(f(x)) = (v_1 f(\alpha_1), v_2 f(\alpha_2), \dots, v_n f(\alpha_n)) \, ,$$

indicating that the codeword $\mathbf{f} = \mathbf{ev}_{\boldsymbol{\alpha},\mathbf{v}}(f(x))$ arises from evaluating the polynomial $f(x)$ at $\boldsymbol{\alpha}$ and scaling by $\mathbf{v}$.

(5.1.1) Theorem. $\mathrm{GRS}_{n,k}(\boldsymbol{\alpha}, \mathbf{v})$ *is an* $[n, k]$ *linear code over* F *with length* $n \le |F|$*. We have* $\mathrm{d}_{\min} = n - k + 1$ *provided* $k \ne 0$*. In particular, GRS codes are MDS codes.*

Proof. As by definition the entries in α are distinct, we must have $n \leq |F|$. If $a \in F$ and $f(x), g(x) \in F[x]_k$, then $af(x) + g(x)$ is also in $F[x]_k$; and

$$\mathbf{ev}_{\boldsymbol{\alpha},\mathbf{v}}(af(x) + g(x)) = a\,\mathbf{ev}_{\boldsymbol{\alpha},\mathbf{v}}(f(x)) + \mathbf{ev}_{\boldsymbol{\alpha},\mathbf{v}}(g(x)) = a\mathbf{f} + \mathbf{g}\,.$$

Therefore $\mathrm{GRS}_{n,k}(\boldsymbol{\alpha}, \mathbf{v})$ is linear of length n over F.

Let $f(x), g(x) \in F[x]_k$ be distinct polynomials. Set $h(x) = f(x) - g(x) \neq 0$, also of degree less than k. Then $\mathbf{h} = \mathbf{f} - \mathbf{g}$ and $\mathrm{w_H}(\mathbf{h}) = \mathrm{d_H}(\mathbf{f}, \mathbf{g})$. But the weight of $\mathbf{h}$ is n minus the number of 0's in $\mathbf{h}$. As all the v_i are nonzero, this equals n minus the number of roots that $h(x)$ has among $\{\alpha_1, \ldots, \alpha_n\}$. As $h(x)$ has at most $k - 1$ roots by Proposition A.2.10, the weight of $\mathbf{h}$ is at least $n - (k - 1) = n - k + 1$. Therefore $\mathrm{d_{min}} \geq n - k + 1$, and we get equality from the Singleton bound 3.1.14. (Alternatively, $h(x) = \prod_{i=1}^{k-1}(x - \alpha_i)$ produces a codeword $\mathbf{h}$ of weight $n - k + 1$.)

The argument of the previous paragraph also shows that distinct polynomials $f(x), g(x)$ of $F[x]_k$ give distinct codewords. Therefore the code contains $|F|^k$ codewords and has dimension k. □

The vector $\mathbf{v}$ plays little role here, and its uses will be more apparent later. At present, it serves to make sure that any code that is monomially equivalent to a *GRS* code is itself a *GRS* code.

Let us now find a generator matrix for $\mathrm{GRS}_{n,k}(\boldsymbol{\alpha}, \mathbf{v})$. The argument of Theorem 5.1.1 makes it clear that any basis $f_1(x), \ldots, f_k(x)$ of $F[x]_k$ gives rise to a basis $\mathbf{f}_1, \ldots, \mathbf{f}_k$ of the code. A particularly nice polynomial basis is the set of monomials $1, x, \ldots, x^i, \ldots, x^{k-1}$. The corresponding generator matrix, whose i^{th} row (numbering rows from 0 to $k - 1$) is

$$\mathbf{ev}_{\boldsymbol{\alpha},\mathbf{v}}(x^i) = (v_1\alpha_1^i, \ldots, v_j\alpha_j^i, \ldots, v_n\alpha_n^i)\,,$$

canonical generator matrix

is the *canonical generator matrix* for $\mathrm{GRS}_{n,k}(\boldsymbol{\alpha}, \mathbf{v})$:

$$\begin{bmatrix} v_1 & v_2 & \ldots & v_j & \ldots & v_n \\ v_1\alpha_1 & v_2\alpha_2 & \ldots & v_j\alpha_j & \ldots & v_n\alpha_n \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ v_1\alpha_1^i & v_2\alpha_2^i & \ldots & v_j\alpha_j^i & \ldots & v_n\alpha_n^i \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ v_1\alpha_1^{k-1} & v_2\alpha_2^{k-1} & \ldots & v_j\alpha_j^{k-1} & \ldots & v_n\alpha_n^{k-1} \end{bmatrix}$$

(5.1.2) Problem. *Consider the code $C = \mathrm{GRS}_{n,k}(\boldsymbol{\alpha}, \mathbf{v})$, and assume that all the entries of the vector $\boldsymbol{\alpha}$ are nonzero. If*

$$\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \ \ldots\ , \alpha_n)\ ,$$

define

$$\boldsymbol{\beta} = (\alpha_1^{-1}, \alpha_2^{-1}, \ \ldots\ , \alpha_n^{-1})\ .$$

Find a vector $\mathbf{w}$ *such that* $C = \mathrm{GRS}_{n,k}(\beta, \mathbf{w})$.

(5.1.3) Problem. (a) *Consider the code* $\mathrm{GRS}_{n,k}(\boldsymbol{\alpha},\mathbf{v})$ *over* F. *Let* a,c *be nonzero elements of* F, *and let* $\mathbf{b}$ *be the vector of* F^n *all of whose entries are equal to* $b\in F$. *Prove that*

$$\mathrm{GRS}_{n,k}(\boldsymbol{\alpha},\mathbf{v})=\mathrm{GRS}_{n,k}(a\boldsymbol{\alpha}+\mathbf{b},c\mathbf{v})\,.$$

(b) *If* $n<|F|$, *prove that there is an* $\boldsymbol{\alpha}'$ *with no entries equal to* 0 *and*

$$\mathrm{GRS}_{n,k}(\boldsymbol{\alpha},\mathbf{v})=\mathrm{GRS}_{n,k}(\boldsymbol{\alpha}',\mathbf{v})\,.$$

(5.1.4) Problem. *Consider the code* E, *which will be linear of length* $n+1$ *and dimension* k, *whose generator matrix results from adding a new column to the canonical generator matrix for* $\mathrm{GRS}_{n,k}(\boldsymbol{\alpha},\mathbf{v})$:

$$\begin{bmatrix} v_1 & v_2 & \dots & v_j & \dots & v_n & 0\\ v_1\alpha_1 & v_2\alpha_2 & \dots & v_j\alpha_j & \dots & v_n\alpha_n & 0\\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \vdots\\ v_1\alpha_1^i & v_2\alpha_2^i & \dots & v_j\alpha_j^i & \dots & v_n\alpha_n^i & 0\\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \vdots\\ v_1\alpha_1^{k-2} & v_2\alpha_2^{k-2} & \dots & v_j\alpha_j^{k-2} & \dots & v_n\alpha_n^{k-2} & 0\\ v_1\alpha_1^{k-1} & v_2\alpha_2^{k-1} & \dots & v_j\alpha_j^{k-1} & \dots & v_n\alpha_n^{k-1} & 1 \end{bmatrix}$$

Prove that $\mathrm{d}_{\min}(E)=n-k+2$.

Remark. *As* $n-k+2=(n+1)-k+1$, *this shows that the code* E *satisfies the Singleton Bound with equality and so is maximum distance separable (MDS), just as all GRS codes are.*

It is extremely profitable to think of Theorem 5.1.1 again in terms of polynomial interpolation:

> Any polynomial of degree less than k is uniquely determined by its values at k (or more) distinct points.

Here, any codeword with as many as k entries equal to 0 corresponds to a polynomial of degree less than k whose values match the 0 polynomial in k points and so must itself be the 0 polynomial.

Given any n-tuple $\mathbf{f}$, we can easily reconstruct the unique polynomial $f(x)$ of degree less than n with $\mathbf{f}=\mathbf{ev}_{\boldsymbol{\alpha},\mathbf{v}}(f(x))$. We first introduce some notation. Set

$$L(x)=\prod_{i=1}^{n}(x-\alpha_i)$$

and

$$L_i(x)=L(x)/(x-\alpha_i)=\prod_{j\neq i}(x-\alpha_j)\,.$$

The polynomials $L(x)$ and $L_i(x)$ are monic of degrees n and $n-1$, respectively. The vector $\mathbf{f}$ has i^{th} coordinate $v_if(\alpha_i)$, so we have enough information to calculate, using the Lagrange interpolation formula A.2.11,

$$f(x)=\sum_{i=1}^{n}\frac{L_i(x)}{L_i(\alpha_i)}f(\alpha_i)\ .$$

The coefficients $L_i(\alpha_i)$ are always nonzero and are often easy to compute.

(5.1.5) PROBLEM. *(a) Prove that $L_i(\alpha_i) = L'(\alpha_i)$, where $L'(x)$ is the formal derivative of $L(x)$ as defined in Problem A.2.26.*

(b) If $n = |F|$ and $\{\alpha_1, \ldots, \alpha_n\} = F$, then $L_i(\alpha_i) = -1$, for all i.

(c) If $\{\alpha_1, \ldots, \alpha_n\}$ is composed of n roots of $x^n - 1$ in F, then $L_i(\alpha_i) = n\alpha_i^{-1}$ $(\neq 0)$. In particular, if $n = |F| - 1$ and $\{\alpha_1, \ldots, \alpha_n\} = F - \{0\}$, then $L_i(\alpha_i) = -\alpha_i^{-1}$ (hence $\alpha_i^{-1} L_i(\alpha_i)^{-1} = -1$).

The polynomial $f(x)$ has degree less than k, while the interpolation polynomial of the righthand side above has apparent degree $n - 1$. The resolution of this confusion allows us to find the dual of a *GRS* code easily.

(5.1.6) THEOREM. *We have*

$$\mathrm{GRS}_{n,k}(\boldsymbol{\alpha}, \mathbf{v})^\perp = \mathrm{GRS}_{n,n-k}(\boldsymbol{\alpha}, \mathbf{u}),$$

where $\mathbf{u} = (u_1, \ldots, u_n)$ with $u_i^{-1} = v_i \prod_{j \neq i}(\alpha_i - \alpha_j)$.

PROOF. By definition $u_i = v_i^{-1} L_i(\alpha_i)^{-1}$.

We prove that every $\mathbf{f}$ in $\mathrm{GRS}_{n,k}(\boldsymbol{\alpha}, \mathbf{v})$ has dot product 0 with every $\mathbf{g}$ in $\mathrm{GRS}_{n,n-k}(\boldsymbol{\alpha}, \mathbf{u})$, from which the result is immediate. Let $\mathbf{f} = \mathbf{ev}_{\boldsymbol{\alpha},\mathbf{v}}(f(x))$ and $\mathbf{g} = \mathbf{ev}_{\boldsymbol{\alpha},\mathbf{u}}(g(x))$. The polynomial $f(x)$ has degree less than k while $g(x)$ has degree less than $n - k$. Therefore their product $f(x)g(x)$ has degree at most $n - 2$. By Lagrange interpolation A.2.11 we have

$$f(x)g(x) = \sum_{i=1}^{n} \frac{L_i(x)}{L_i(\alpha_i)} f(\alpha_i)g(\alpha_i) \; .$$

Equating the coefficient of x^{n-1} from the two sides gives:

$$\begin{aligned} 0 &= \sum_{i=1}^{n} \frac{1}{L_i(\alpha_i)} f(\alpha_i)g(\alpha_i) \\ &= \sum_{i=1}^{n} (v_i f(\alpha_i)) \left(\frac{v_i^{-1}}{L_i(\alpha_i)} g(\alpha_i) \right) \\ &= \sum_{i=1}^{n} (v_i f(\alpha_i))(u_i g(\alpha_i)) \\ &= \mathbf{f} \cdot \mathbf{g}\,, \end{aligned}$$

as required. □

The ability in the class of *GRS* codes to choose different vectors $\mathbf{v}$ to accompany a fixed $\boldsymbol{\alpha}$ has been helpful here.

Of course, to specify $\mathbf{f}$ as a codeword in $C = \mathrm{GRS}_{n,k}(\boldsymbol{\alpha}, \mathbf{v})$ we do not need to check it against every $\mathbf{g}$ of $C^\perp = \mathrm{GRS}_{n,n-k}(\boldsymbol{\alpha}, \mathbf{u})$. It is enough to consider a basis of $C^\perp$, a nice one being the rows of the canonical generator matrix for

$C^\perp$, a check matrix for C. Our introduction of *GRS* codes such as C essentially defines them via their canonical generator matrices. As we have seen before, describing a linear code instead in terms of a check matrix can be fruitful. In particular this opens the possibility of syndrome decoding.

Set $r = n - k$, and let $\mathbf{c} = (c_1, \dots, c_n) \in F^n$. Then

$$\begin{aligned} \mathbf{c} \in C \quad &\iff \quad 0 = \mathbf{c} \cdot \mathbf{ev}_{\boldsymbol{\alpha},\mathbf{u}}(x^j), \text{ for } 0 \le j \le r-1 \\ &\iff \quad 0 = \sum_{i=1}^{n} c_i u_i \alpha_i^j, \text{ for } 0 \le j \le r-1\,. \end{aligned}$$

We rewrite these r equations as a single equation in the polynomial ring $F[z]$ in a new indeterminate z. The vector $\mathbf{c}$ is in C if and only if in $F[z]$ we have

$$\begin{aligned} 0 \quad &= \quad \sum_{j=0}^{r-1} \Big(\sum_{i=1}^{n} c_i u_i \alpha_i^j \Big) z^j \\ &= \quad \sum_{i=1}^{n} c_i u_i \Big(\sum_{j=0}^{r-1} (\alpha_i z)^j \Big) \end{aligned}$$

The polynomials $1 - \alpha z$ and z^r are relatively prime, so it is possible to invert $1 - \alpha z$ in the ring $F[z] \pmod{z^r}$. Indeed

$$\frac{1}{1 - \alpha z} = \sum_{j=0}^{r-1} (\alpha z)^j \pmod{z^r}\,,$$

a truncation of the usual geometric series expansion (which could equally well be taken as a definition for the inverse of $1 - \alpha z$ module z^r). We are left with:

(5.1.7) Theorem. (Goppa formulation for GRS codes.) *The generalized Reed-Solomon code* $\mathrm{GRS}_{n,k}(\boldsymbol{\alpha}, \mathbf{v})$ *over F is equal to the set of all n-tuples* $\mathbf{c} = (c_1, c_2, \dots, c_n) \in F^n$, *such that*

$$\sum_{i=1}^{n} \frac{c_i u_i}{1 - \alpha_i z} = 0 \pmod{z^r}\ ,$$

where $r = n - k$ and $u_i^{-1} = v_i \prod_{j \ne i} (\alpha_i - \alpha_j)$. □

This interpretation of GRS codes has two main values. First, it is open to a great deal of generalization, as we shall later see. Second, it suggests a practical method for the decoding of GRS codes, the topic of the next section.

5.2 Decoding GRS codes

As GRS codes are *MDS*, they can be decoded using threshold decoding as in Problem 3.2.4. We now present an efficient and more specific algorithm for decoding the dual of $\mathrm{GRS}_{n,r}(\boldsymbol{\alpha}, \mathbf{u})$, starting from the Goppa formulation.

syndrome polynomia

Suppose $\mathbf{c} = (c_1, c_2, \ldots, c_n)$ is transmitted, and $\mathbf{p} = (p_1, p_2, \ldots, p_n)$ is received, so that the error vector $\mathbf{e} = (e_1, e_2, \ldots, e_n)$ has been introduced; $\mathbf{p} = \mathbf{c} + \mathbf{e}$. We calculate the *syndrome polynomial* of $\mathbf{p}$:

$$S_{\mathbf{p}}(z) = \sum_{i=1}^{n} \frac{p_i u_i}{1 - \alpha_i z} \pmod{z^r} .$$

Then it is easily seen that

$$S_{\mathbf{p}}(z) = S_{\mathbf{c}}(z) + S_{\mathbf{e}}(z) \pmod{z^r},$$

whence, by the Goppa formulation of Theorem 5.1.7,

$$S_{\mathbf{p}}(z) = S_{\mathbf{e}}(z) \pmod{z^r}.$$

Let B be the set of error locations:

$$B = \{ i \mid e_i \neq 0 \} .$$

Then we have the syndrome polynomial

$$S_{\mathbf{p}}(z) = S_{\mathbf{e}}(z) = \sum_{b \in B} \frac{e_b u_b}{1 - \alpha_b z} \pmod{z^r}.$$

We now drop the subscripts and write $S(z)$ for the syndrome polynomial.

Key Equation

Clear denominators to find the *Key Equation*:

$$\sigma(z) S(z) = \omega(z) \pmod{z^r},$$

where

$$\sigma(z) = \sigma_{\mathbf{e}}(z) = \prod_{b \in B} (1 - \alpha_b z)$$

and

$$\omega(z) = \omega_{\mathbf{e}}(z) = \sum_{b \in B} e_b u_b \Big(\prod_{a \in B, a \neq b} (1 - \alpha_a z) \Big) .$$

error locator
error evaluator

(Empty products are taken as 1.) The polynomial $\sigma(z)$ is called the *error locator* polynomial, and the polynomial $\omega(z)$ is the *error evaluator* polynomial.

The names are justifiable. Given the polynomials $\sigma(z) = \sigma_{\mathbf{e}}(z)$ and $\omega(z) = \omega_{\mathbf{e}}(z)$, we can reconstruct the error vector $\mathbf{e}$. Assume for the moment that none of the α_i are equal to 0 (although similar results are true when some α_i is 0) . Then:

$$B = \{ \, b \mid \sigma(\alpha_b^{-1}) = 0 \, \}$$

and, for each $b \in B$,

$$e_b = \frac{-\alpha_b \omega(\alpha_b^{-1})}{u_b \sigma'(\alpha_b^{-1})} ,$$

where $\sigma'(z)$ is the formal derivative of $\sigma(z)$ (see Problem A.2.26). In fact the polynomials $\sigma(z)$ and $\omega(z)$ determine the error vector even when some α_i is 0.

(5.2.1) PROBLEM. *Let $\sigma(z)$ and $\omega(z)$ be the error locator and evaluator polynomials for the error vector $\mathbf{e} \neq \mathbf{0}$. Set $B = \{\, b \mid e_b \neq 0 \,\}$ and $B_0 = \{\, b \in B \,|\, \alpha_b \neq 0\}$. Recall that there is at most one index b with $\alpha_b = 0$.*

Prove the following:

(a) $\mathrm{w_H}(\mathbf{e}) = |B|$ *is equal to* $|B_0| + 1$ *or* $|B_0|$ *depending upon whether or not there is an index b with $\alpha_b = 0$ and $e_b \neq 0$.*

(b) $\deg \sigma(z) = |B_0|$ *and* $\deg \omega(z) \leq |B| - 1$.

(c) $B_0 = \{\, b \,|\, \alpha_b \neq 0,\ \sigma(\alpha_b^{-1}) = 0 \,\}$.

(d) *The index b with $\alpha_b = 0$ belongs to B if and only if* $\deg \sigma(z) = \deg \omega(z)$.

(e) *For $b \in B_0$, e_b is given by the formula above. If $b \in B \setminus B_0$ then*

$$e_b = w u_b^{-1} \Big(\prod_{a \in B_0} (-\alpha_a) \Big)^{-1} ,$$

where w is the coefficient of z^f in $\omega(z)$ for $f = \deg \omega(z)$.

If the error vector $\mathbf{e} \neq \mathbf{0}$ has weight at most $r/2$ $(= (\mathrm{d_{min}} - 1)/2)$, then relative to the syndrome polynomial $S_{\mathbf{e}}(z) = S(z)$ the pair of polynomials $\sigma_{\mathbf{e}}(z) = \sigma(z)$ and $\omega_{\mathbf{e}}(z) = \omega(z)$ has the three properties by which it is characterized in the next theorem. Indeed (1) is just the Key Equation. Property (2) is a consequence of the assumption on error weight and the definitions of the polynomials $\sigma(z)$ and $\omega(z)$ (see Problem 5.2.1 above). For (3) we have $\sigma(0) = 1$ trivially. As $\sigma(z)$ has $\deg(\sigma(z))$ distinct roots, either $\gcd(\sigma(z), \omega(z)) = 1$ or the two polynomials have a common root. But for each root α_b^{-1} of $\sigma(z)$ we have $0 \neq \omega(\alpha_b^{-1})$, a factor of $e_b \neq 0$.

Our decoding method solves the Key Equation and so finds the error vector $\mathbf{e}$ as above. The following theorem provides us with a characterization of the solutions we seek.

(5.2.2) THEOREM. *Given r and $S(z) \in F[z]$ there is at most one pair of polynomials $\sigma(z), \omega(z)$ in $F[z]$ satisfying:*

(1) $\sigma(z)S(z) = \omega(z) \pmod{z^r}$;

(2) $\deg(\sigma(z)) \leq r/2$ *and* $\deg(\omega(z)) < r/2$;

(3) $\gcd(\sigma(z), \omega(z)) = 1$ *and* $\sigma(0) = 1$.

In fact we prove something slightly stronger.

(5.2.3) PROPOSITION. *Assume that $\sigma(z), \omega(z)$ satisfy* (1)-(3) *of Theorem 5.2.2 and that $\sigma_1(z), \omega_1(z)$ satisfy* (1) *and* (2). *Then there is a polynomial $\mu(z)$ with $\sigma_1(z) = \mu(z)\sigma(z)$ and $\omega_1(z) = \mu(z)\omega(z)$.*

PROOF. From (1)

$$\sigma(z)\omega_1(z) = \sigma(z)\sigma_1(z)S(z) = \sigma_1(z)\omega(z) \pmod{z^r} ;$$

so

$$\sigma(z)\omega_1(z) - \sigma_1(z)\omega(z) = 0 \pmod{z^r} .$$

But by (2) the lefthand side of this equation has degree less than r. Therefore

$$\sigma(z)\omega_1(z) = \sigma_1(z)\omega(z)\,.$$

From (3) we have $\gcd(\sigma(z), \omega(z)) = 1$, so by Lemma A.2.20 $\sigma(z)$ divides $\sigma_1(z)$. Set $\sigma_1(z) = \mu(z)\sigma(z)$. Then

$$\sigma(z)\omega_1(z) = \sigma_1(z)\omega(z) = \sigma(z)\mu(z)\omega(z)\,.$$

The polynomial $\sigma(z)$ is nonzero since $\sigma(0) = 1$; so by cancellation $\omega_1(z) = \mu(z)\omega(z)$, as desired. □

Proof of Theorem 5.2.2.

Any second such pair has

$$\sigma_1(z) = \mu(z)\sigma(z) \text{ and } \omega_1(z) = \mu(z)\omega(z)$$

by the proposition. So $\mu(z)$ divides $\gcd(\sigma_1(z), \omega_1(z))$ which is 1 by (3). Therefore $\mu(z) = \mu$ is a constant. Indeed

$$1 = \sigma_1(0) = \mu(0)\sigma(0) = \mu \cdot 1 = \mu\,.$$

Thus $\sigma_1(z) = \mu(z)\sigma(z) = \sigma(z)$ and $\omega_1(z) = \mu(z)\omega(z) = \omega(z)$. □

Using the characterization of Theorem 5.2.2 we now verify a method of solving the Key Equation with the Euclidean algorithm, as presented in Section A.3.1 of the appendix.

(5.2.4) Theorem. (Decoding *GRS* using the Euclidean Algorithm.) *Consider the code* $\mathrm{GRS}_{n,k}(\boldsymbol{\alpha}, \mathbf{v})$ *over* F*, and set* $r = n - k$*. Given a syndrome polynomial* $S(z)$ *(of degree less than* r*), the following algorithm halts, producing polynomials* $\tilde\sigma(z)$ *and* $\tilde\omega(z)$*:*

```
Set $a(z) = z^r$ and $b(z) = S(z)$.
Step through the Euclidean Algorithm A.3.1
        until at Step j, $\deg(r_j(z)) < r/2$.
Set $\tilde\sigma(z) = t_j(z)$
and $\tilde\omega(z) = r_j(z)$.
```

If there is an error word $\mathbf{e}$ *of weight at most* $r/2 = (\mathrm{d}_{\min} - 1)/2$ *with* $S_{\mathbf{e}}(z) = S(z)$*, then* $\widehat\sigma(z) = \tilde\sigma(0)^{-1}\tilde\sigma(z)$ *and* $\widehat\omega(z) = \tilde\sigma(0)^{-1}\tilde\omega(z)$ *are the error locator and evaluator polynomials for* $\mathbf{e}$*.*

Proof. It is the goal of the Euclidean algorithm to decrease the degree of $r_j(z)$ at each step, so the algorithm is guaranteed to halt.

Now assume that $S(z) = S_{\mathbf{e}}(z)$ with $\mathrm{w_H}(\mathbf{e}) \leq r/2$. Therefore the error locator and evaluator pair $\sigma(z) = \sigma_{\mathbf{e}}(z)$ and $\omega(z) = \omega_{\mathbf{e}}(z)$ satisfies (1), (2), and

(3) of Theorem 5.2.2. We first check that, for the j defined, the pair $t_j(z)$ and $r_j(z)$ satisfies (1) and (2).

Requirement (1) is just the Key Equation and is satisfied at each step of the Euclidean algorithm since always

$$E_j: \quad r_j(z) = s_j(z)z^r + t_j(z)S(z).$$

For (2), our choice of j gives $\deg(r_j(z)) < r/2$ and also $\deg(r_{j-1}(z)) \geq r/2$. Therefore, from Problem A.3.5,

$$\begin{aligned} \deg(t_j(z)) + r/2 &\leq \deg(t_j(z)) + \deg(r_{j-1}(z)) \\ &= \deg(a(z)) = \deg(z^r) = r\,. \end{aligned}$$

Hence $\deg(t_j(z)) \leq r/2$, giving (2).

By Proposition 5.2.3 there is a polynomial $\mu(z)$ with

$$t_j(z) = \mu(z)\sigma(z) \text{ and } r_j(z) = \mu(z)\omega(z)\,.$$

Here $\mu(z)$ is not the zero polynomial by Lemma A.3.3(1).

If we substitute for $t_j(z)$ and $r_j(z)$ in equation E_j we have

$$s_j(z)z^r + (\mu(z)\sigma(z))S(z) = \mu(z)\omega(z)\,,$$

which becomes

$$\mu(z)\Big(\omega(z) - \sigma(z)S(z)\Big) = s_j(z)z^r\,.$$

By the Key Equation, the parenthetical expression on the left is $p(z)z^r$, for some $p(z)$; so we are left with $\mu(z)p(z)z^r = s_j(z)z^r$ or

$$\mu(z)p(z) = s_j(z)\,.$$

Thus $\mu(z)$ divides $\gcd(t_j(z), s_j(z))$, which is 1 by Corollary A.3.4.

We conclude that $\mu(z) = \mu$ is a nonzero constant function. Furthermore

$$t_j(0) = \mu(0)\sigma(0) = \mu\,;$$

so

$$\sigma(z) = t_j(0)^{-1}t_j(z) \text{ and } \omega(z) = t_j(0)^{-1}r_j(z)\,,$$

as desired. □

When this algorithm is used, decoding default occurs when $\widehat{\sigma}(z)$ does not split into linear factors whose roots are inverses of entries in α with multiplicity 1. (Here we assume that none of the α_i are 0.) That is, the number of roots of $\widehat{\sigma}(z)$ among the α_i^{-1} must be equal to the degree of $\widehat{\sigma}(z)$. If this is not the case, then we have detected errors that we are not able to correct. Another instance of decoder default occurs when $t_j(0) = 0$, so the final division to determine $\widehat{\sigma}(z)$ can not be made.

If $t_j(0) \neq 0$ and $\widehat{\sigma}(z)$ does split as described, then we can go on to evaluate errors at each of the located positions and find a vector of weight at most $r/2$ with our original syndrome. In this case we have either decoded correctly, or we had more than $r/2$ errors and have made a decoding error. (We need not worry about division by 0 in evaluating errors, since this can only happen if $\widehat{\sigma}(z)$ has roots of multiplicity greater than one; see Problem A.2.27.)

Assume now that r is even or that $\boldsymbol{\alpha}$ has weight n. Then this algorithm only produces error vectors of weight $r/2$ or less. In particular if more than $r/2$ errors occur then we will have a decoding default or a decoder error. Suppose that we have found polynomials $\widehat{\sigma}(z)$ and $\widehat{\omega}(z)$ that allow us to calculate a candidate error vector $\mathbf{e}$ of weight at most $r/2$. It follows from Lagrange interpolation A.2.11 that $\widehat{\sigma}(z) = \sigma_{\mathbf{e}}(z)$ and $\widehat{\omega}(z) = \omega_{\mathbf{e}}(z)$. Also since $\sigma(z)$ is invertible modulo z^r, we can solve the Key Equation to find that $S(z) = S_{\mathbf{e}}(z)$. Therefore the received vector is within a sphere of radius $r/2$ around a codeword and is decoded to that codeword. That is, under these conditions Euclidean algorithm decoding as given in Theorem 5.2.4 is an explicit implementation of the decoding algorithm $\mathbf{SS}_{r/2}$.

Example. Consider the code $C = \mathrm{GRS}_{6,2}(\boldsymbol{\alpha}, \mathbf{v})$ over the field $\mathbb{F}_7$ of integers modulo 7, where

$$\boldsymbol{\alpha} = (2, 4, 6, 1, 3, 5)$$

and

$$\mathbf{v} = (1, 1, 1, 1, 1, 1)\,.$$

First calculate a vector $\mathbf{u}$ for which $C^{\perp} = \mathrm{GRS}_{6,4}(\boldsymbol{\alpha}, \mathbf{u})$. Starting with

$$L(x) = (x-2)(x-4)(x-6)(x-1)(x-3)(x-5)$$

we find:

$L_1(2) =$		(-2)	(-4)	(1)	(-1)	(-3)	$=$	24	$= 3$
$L_2(4) =$	(2)		(-2)	(3)	(1)	(-1)	$=$	12	$= 5$
$L_3(6) =$	(4)	(2)		(5)	(3)	(1)	$=$	120	$= 1$
$L_4(1) =$	(-1)	(-3)	(-5)		(-2)	(-4)	$=$	-120	$= 6$
$L_5(3) =$	(1)	(-1)	(-3)	(2)		(-2)	$=$	-12	$= 2$
$L_6(5) =$	(3)	(1)	(-1)	(4)	(2)		$=$	-24	$= 4$

(Notice that these values could have been found easily using Problem 5.1.5(c).) Now $u_i = (v_i L_i(\alpha_i))^{-1} = L_i(\alpha_i)^{-1}$ since $v_i = 1$; so

$$\mathbf{u} = (5, 3, 1, 6, 4, 2)\,.$$

Next calculate the syndrome polynomial of an arbitrary received vector

$$\mathbf{p} = (p_1, p_2, p_3, p_4, p_5, p_6)\,.$$

In our example $r = 6 - 2 = 4$.

$$S_{\mathbf{p}}(z) = \frac{5 \cdot p_1}{1-2z} + \frac{3 \cdot p_2}{1-4z} + \frac{1 \cdot p_3}{1-6z} + \frac{6 \cdot p_4}{1-1z} + \frac{4 \cdot p_5}{1-3z} + \frac{2 \cdot p_6}{1-5z} \pmod{z^4}$$

$$
\begin{array}{rrrrrrl}
= & 5p_1(& 1 & +2z & +4z^2 & +z^3) & \\
 & +3p_2(& 1 & +4z & +2z^2 & +z^3) & \\
 & +p_3(& 1 & +6z & +z^2 & +6z^3) & \\
 & +6p_4(& 1 & +z & +z^2 & +z^3) & \pmod{z^4} \\
 & +4p_5(& 1 & +3z & +2z^2 & +6z^3) & \\
 & +2p_6(& 1 & +5z & +4z^2 & +6z^3) &
\end{array}
$$

$$
\begin{array}{rrrrrrl}
= & p_1(& 5 & +3z & +6z^2 & +5z^3) & \\
 & +p_2(& 3 & +5z & +6z^2 & +3z^3) & \\
 & +p_3(& 1 & +6z & +z^2 & +6z^3) & \\
 & +p_4(& 6 & +6z & +6z^2 & +6z^3) & \pmod{z^4}\ . \\
 & +p_5(& 4 & +5z & +z^2 & +3z^3) & \\
 & +p_6(& 2 & +3z & +z^2 & +5z^3) &
\end{array}
$$

Notice that this calculation amounts to finding the canonical check matrix for the code.

We now use the algorithm of Theorem 5.2.4 to decode the received vector

$$\mathbf{p} = (1, 3, 6, 5, 4, 2)\,.$$

We have the syndrome polynomial

$$S(z) = \frac{5 \cdot 1}{1 - 2z} + \frac{3 \cdot 3}{1 - 4z} + \frac{1 \cdot 6}{1 - 6z} + \frac{6 \cdot 5}{1 - 1z} + \frac{4 \cdot 4}{1 - 3z} + \frac{2 \cdot 2}{1 - 5z} \pmod{z^4}$$

$$
\begin{array}{rrrrrrl}
= & 1(& 5 & +3z & +6z^2 & +5z^3) & \\
 & +3(& 3 & +5z & +6z^2 & +3z^3) & \\
 & +6(& 1 & +6z & +z^2 & +6z^3) & \\
 & +5(& 6 & +6z & +6z^2 & +6z^3) & \pmod{z^4}\ . \\
 & +4(& 4 & +5z & +z^2 & +3z^3) & \\
 & +2(& 2 & +3z & +z^2 & +5z^3) &
\end{array}
$$

$$= 5z + 3z^2 + 4z^3 \pmod{z^4}\ .$$

The algorithm now requires that, starting with initial conditions

$$a(z) = z^4 \text{ and } b(z) = 4z^3 + 3z^2 + 5z\,,$$

we step through the Euclidean Algorithm until at Step j we first have $\deg(r_j(z)) < r/2 = 2$.

This is precisely the Euclidean Algorithm example done in the appendix. At Step 2. we have the first occurrence of a remainder term with degree less than 2; we have $r_2(z) = 6z$. We also have $t_2(z) = 3z^2 + 6z + 4$, so $t_2(0)^{-1} = 4^{-1} = 2$. Therefore we have error locator and evaluator polynomials:

$$\sigma(z) = t_2(0)^{-1} t_2(z) = 2(3z^2 + 6z + 4) = 6z^2 + 5z + 1$$

$$\omega(z) = t_2(0)^{-1} r_2(z) = 2(6z) = 5z\ .$$

The error locations are those in $B = \{\, b \mid \sigma(\alpha_b^{-1}) = 0 \,\}$; so to find the error locations, we must extract the roots of $\sigma(z)$. As $\mathbb{F}_7$ does not have characteristic 2, we can use the usual quadratic formula and find that the roots are

$$2\,,\,3 = \frac{-5 \pm \sqrt{25 - 4 \cdot 6}}{2 \cdot 6}\ .$$

Now $2^{-1} = 4 = \alpha_2$ and $3^{-1} = 5 = \alpha_6$, so $B = \{2, 6\}$.

An error value e_b is given by

$$e_b = \frac{-\alpha_b \omega(\alpha_b^{-1})}{u_b \sigma'(\alpha_b^{-1})} \ ,$$

where $\sigma'(z) = 5z + 5$. Thus $e_2 = \frac{-4\cdot 10}{3\cdot 15} = 3$ and $e_6 = \frac{-5\cdot 15}{2\cdot 20} = 6$.

We have thus found

$$\mathbf{e} = (0, 3, 0, 0, 0, 6) \,,$$

so we decode the received word $\mathbf{p}$ to

$$\mathbf{c} = \mathbf{p} - \mathbf{e} = (1, 3, 6, 5, 4, 2) - (0, 3, 0, 0, 0, 6) = (1, 0, 6, 5, 4, 3) \ .$$

In the example we have been able to use the quadratic formula to calculate the roots of $\sigma(z)$ and so find the error locations. This will not always be possible. There may be more than 2 errors. In any case, the quadratic formula involves division by 2 and so is not valid when the characteristic of the field F is 2, one of the most interesting cases. A method that is often used is the substitution, one-by-one, of all field elements into $\sigma(z)$, a *Chien search.* Although this lacks subtlety, it is manageable when the field is not too big. There do not seem to be general alternatives that are good and simple. Chien search

(5.2.5) PROBLEM. *Consider the* $\mathrm{GRS}_{8,4}(\boldsymbol{\alpha}, \mathbf{v})$ *code* C *over* $\mathbb{F}_{13}$ *with*

$$\mathbf{v} = (1, 1, 1, 1, 1, 1, 1, 1)$$

$$\boldsymbol{\alpha} = (1, 4, 3, 12, 9, 10, 5, 8) \ .$$

(*a*) *Give* $n, k, \boldsymbol{\beta}, \mathbf{u}$ *with* $C^{\perp} = \mathrm{GRS}_{n,k}(\boldsymbol{\beta}, \mathbf{u})$.

(*b*) *When transmitting with* C, *assume that the vector*

$$\mathbf{p} = (0, 0, 0, 0, 0, 0, 3, 5) \ .$$

is received. Use the Euclidean algorithm to find an error vector $\mathbf{e}$ *and a decoded codeword* $\mathbf{c}$. *(The answers should be obvious. Use the question to check your understanding of the process.)*

(*c*) *When transmitting with* C, *assume that the vector*

$$\mathbf{p} = (3, 6, 0, 4, 0, 5, 0, 12)$$

is received. Use the Euclidean algorithm to find an error vector $\mathbf{e}$ *and a decoded codeword* $\mathbf{c}$.

(5.2.6) PROBLEM. *Consider the* $\mathrm{GRS}_{10,4}(\boldsymbol{\alpha}, \mathbf{v})$ *code* C *over* $\mathbb{F}_{13}$ *with*

$$\mathbf{v} = (1, 1, 1, 1, 1, 1, 1, 1, 1, 1)$$

$$\boldsymbol{\alpha} = (1, 2, 3, 4, 6, 7, 9, 10, 11, 12) \ .$$

(a) *Give* $n, k, \boldsymbol{\beta}, \mathbf{u}$ *with* $C^\perp = \mathrm{GRS}_{n,k}(\boldsymbol{\beta}, \mathbf{u})$.
*(*Hint: $\mathbf{u} = (*, *, 9, 10, 12, 1, 3, 4, *, *)$.*)*

(b) *When transmitting with* C*, assume that the vector*

$$\mathbf{p} = (4, 5, 6, 0, 0, 0, 0, 0, 0, 0) \ .$$

is received. Use the Euclidean algorithm to find an error vector $\mathbf{e}$ *and a decoded codeword* $\mathbf{c}$*. (The answers should be obvious. Use the question to check your understanding of the process.)*

(c) *When transmitting with* C*, assume that the vector*

$$\mathbf{p} = (3, 1, 0, 0, 0, 0, 0, 5, 7, 12) \ .$$

is received. Use the Euclidean algorithm to find an error vector $\mathbf{e}$ *and a decoded codeword* $\mathbf{c}$.

(5.2.7) Problem. *Let the field* $\mathbb{F}_8$ *be given as polynomials of degree at most* 2 *in* α*, a root of the primitive polynomial* $x^3+x+1 \in \mathbb{F}_2[x]$*. Consider the code* $C = \mathrm{GRS}_{7,3}(\boldsymbol{\alpha}, \mathbf{v})$ *over* $\mathbb{F}_8$ *with*

$$\boldsymbol{\alpha} = \mathbf{v} = (1, \alpha, \alpha^2, \alpha^3, \alpha^4, \alpha^5, \alpha^6) \, .$$

By Problem 5.1.5(c) we have $C^\perp = \mathrm{GRS}_{7,4}(\boldsymbol{\alpha}, \mathbf{u})$ *for* $\mathbf{u} = (1, 1, 1, 1, 1, 1, 1)$.

When transmitting with C*, assume that the vector*

$$\mathbf{p} = (0, \alpha^5, 0, 1, \alpha^6, 0, 1)$$

is received. Use the Euclidean Algorithm to find an error vector $\mathbf{e}$ *and a decoded codeword* $\mathbf{c}$.