ECEN 5682 Theory and Practice of Error Control Codes

Cyclic Codes

Peter Mathys

University of Colorado

Spring 2007

Cyclic Codes

Linear cyclic codes have a lot of structure that can be exploited for highly efficient encoder and decoder implementations. The class of linear cyclic codes includes BCH (Bose-Chaudhuri-Hocquenghem) codes, RS (Reed-Solomon) codes, and the Golay codes. It also includes CRC (cyclic redundancy check) error detection codes which are widely used in data communications and data storage.

Definition: A q-ary linear (n, k) blockcode $\mathcal{C}$ is called a cyclic code if every cyclic shift of a codeword in $\mathcal{C}$ is also a codeword in $\mathcal{C}$.

Note: It is possible to define and construct non-linear codes with the property that every cyclic shift of a codeword is another codeword. However, these codes have less structure and are hardly used in practice.

Example: Let $\mathbf{c} = (c_0, c_1, \ldots, c_{n-1})$ be a codeword of $\mathcal{C}$. If $\mathcal{C}$ is a cyclic code, then the cyclic shift to the right by one place of $\mathbf{c}$, denoted $\mathbf{c}^{(1)}$, and defined by

$$\mathbf{c}^{(1)} = (c_{n-1}, c_0, c_1, \ldots, c_{n-2}) ,$$

must also be in $\mathcal{C}$. Repeated application yields the codeword cyclically shifted to the right by i places

$$\mathbf{c}^{(i)} = (c_{n-i}, c_{n-i+1}, \ldots, c_{n-1}, c_0, c_1, \ldots, c_{n-i-1}) ,$$

which must also be in $\mathcal{C}$. Note that a cyclic shift to the right by i places is equivalent to a cyclic shift to the left by $n - i$ places.

Note: A cyclic right shift of a codeword $\mathbf{c}$ over F by i places corresponds to multiplying the corresponding code polynomial $c(x)$ by x^i in the quotient ring $F[x]/(x^n - 1)$, i.e.,

$$\mathbf{c}^{(i)} \qquad \longleftrightarrow \qquad R_{x^n-1}(x^i\, c(x)) .$$

Theorem: The monic nonzero code polynomial of minimum degree in a linear cyclic code $\mathcal{C}$ is unique.

Proof: Suppose $g(x) = x^r + g_{r-1}\,x^{r-1} + \ldots + g_0$ and $g'(x) = x^r + g'_{r-1}\,x^{r-1} + \ldots + g'_0$ are both such monic nonzero code polynomials of minimum degree. Because the code is linear, $g(x) - g'(x) = (g_{r-1} - g'_{r-1})\,x^{r-1} + \ldots + (g_0 - g'_0)$ must also be a codeword. Since it has degree at most $r-1$, this contradicts the statement that $g(x)$ and $g'(x)$ are nonzero code polynomials of smallest degree, unless $g(x) = g'(x)$. QED

Definition: The unique monic nonzero code polynomial of minimum degree in a linear cyclic code $\mathcal{C}$ is denoted by $g(x)$ and called the *generator polynomial* of $\mathcal{C}$.

Theorem: Let $g(x)$ be the generator polynomial of a linear cyclic code $\mathcal{C}$ of blocklength n and assume $\deg g(x) = r < n$. Then a polynomial $c(x)$ with $\deg c(x) < n$ is a code polynomial iff it is a multiple of $g(x)$, i.e., iff $c(x) = a(x)\, g(x)$, where $\deg a(x) < n - r$.

Proof: Suppose $c(x)$ is a multiple of $g(x)$. Then

$$\begin{aligned} c(x) = a(x)\, g(x) &= (a_{n-r-1}\, x^{n-r-1} + \ldots + a_1\, x + a_0)\, g(x) \\ &= a_{n-r-1}\, x^{n-r-1} g(x) + \ldots + a_1\, x g(x) + a_0\, g(x)\,, \end{aligned}$$

i.e., $c(x)$ is a linear combination of the code polynomials $g(x), xg(x), \ldots, x^{n-r-1}g(x)$, and thus $c(x) \in \mathcal{C}$.

Now suppose that $c(x) \in \mathcal{C}$. Use the division algorithm to write that $c(x) = a(x)\, g(x) + r(x)$, with $\deg r(x) < \deg g(x)$. Thus $r(x) = c(x) - a(x)\, g(x)$. But both $c(x) \in \mathcal{C}$ and $a(x)\, g(x) \in \mathcal{C}$ which implies that $r(x) \in \mathcal{C}$ since $\mathcal{C}$ is linear. This is a contradiction to the fact that $g(x)$ is the smallest degree nonzero code polynomial in $\mathcal{C}$, unless $r(x) = 0$. Thus $c(x) = a(x)\, g(x)$. QED

Corollary: Let $g(x)$ be the generator polynomial of a linear cyclic (n, k) code $\mathcal{C}$. Then $\deg g(x) = n - k$.

Proof: Follows immediately from above theorem.

Theorem: A linear cyclic code $\mathcal{C}$ of blocklength n with generator polynomial $g(x)$ exists iff $g(x) \,|\, (x^n - 1)$.

Proof: Using the division algorithm, write $x^n - 1 = h(x)\, g(x) + r(x)$, with $\deg r(x) < \deg g(x)$. Then

$$0 = R_{x^n-1}(x^n - 1) = R_{x^n-1}(h(x)\, g(x) + r(x)) = R_{x^n-1}(h(x)\, g(x)) + r(x),$$

which implies $r(x) = -R_{x^n-1}(h(x)\, g(x))$. That is, $r(x)$ is a code polynomial with degree less than $\deg g(x)$ and the only such code polynomial is $r(x) = 0$. Therefore $g(x) \,|\, (x^n - 1)$. Since every polynomial that divides $x^n - 1$ can be used as a generator polynomial, this completes the proof. QED

Definition: The polynomial $h(x)$ in $x^n - 1 = h(x)\,g(x)$, where $g(x)$ is a generator polynomial, is called *parity-check polynomial*.

Note: The above theorem asserts that any factor of $x^n - 1$ over $\mathrm{GF}(q)$ with degree $n - k$ generates a q-ary linear cyclic (n, k) code. However, not all such factors generate good (i.e., large $d_{\min}$) codes.

Any code polynomial $c(x)$ of a linear cyclic (n, k) code $\mathcal{C}$ can be written as $c(x) = a(x)\,g(x)$, $\deg a(x) < k$, where $g(x)$ is the generator polynomial of $\mathcal{C}$ and $\deg g(x) = n - k$. This polynomial multiplication can be expressed in vector-matrix form as

$$c(x) = \begin{bmatrix} a_0 & a_1 & \cdots & a_{k-1} \end{bmatrix} \begin{bmatrix} g(x) \\ x\,g(x) \\ \vdots \\ x^{k-1}\,g(x) \end{bmatrix}.$$

Setting $\mathbf{c} = (c_0, c_1, \ldots, c_{n-1})$ and $\mathbf{a} = (a_0, a_1, \ldots, a_{k-1})$, a codeword $\mathbf{c}$ can thus be written as $\mathbf{c} = \mathbf{a}\,\mathbf{G}$, where the generator matrix $\mathbf{G}$ is given by

$$\mathbf{G} = \begin{bmatrix} g_0 & g_1 & \cdots & g_{n-k} & & & & \\ & g_0 & g_1 & \cdots & g_{n-k} & & \mathbf{0} & \\ & & \ddots & \ddots & \ddots & \ddots & & \\ & \mathbf{0} & & g_0 & g_1 & \cdots & g_{n-k} & \\ & & & & g_0 & g_1 & \cdots & g_{n-k} \end{bmatrix}.$$

Next, recall that $g(x) \,|\, (x^n - 1)$ and that $x^n - 1 = g(x)\, h(x)$, where $h(x)$ is the parity-check polynomial of $\mathcal{C}$. This leads to the following theorem.

Theorem: Let $\mathcal{C}$ be a linear cyclic (n, k) code with parity-check polynomial $h(x) = \sum_{i=0}^{k} h_i x^i$. Then the parity check matrix $\mathbf{H}$ of $\mathcal{C}$ is given by

$$\mathbf{H} = \begin{bmatrix} h_k & h_{k-1} & \cdots & h_0 & & & \\ & h_k & h_{k-1} & \cdots & h_0 & & \mathbf{0} \\ & & \ddots & \ddots & \ddots & \ddots & \\ & \mathbf{0} & & h_k & h_{k-1} & \cdots & h_0 \\ & & & & h_k & h_{k-1} & \cdots & h_0 \end{bmatrix}.$$

Proof: It needs to be shown that $\mathbf{G}\,\mathbf{H}^T = \mathbf{0}$. Define $v_m = \sum_{i=0}^{m} h_i\, g_{m-i}$. Then

$$\mathbf{G}\,\mathbf{H}^T = \begin{bmatrix} v_k & v_{k+1} & \cdots & v_{n-1} \\ v_{k-1} & v_k & \cdots & v_{n-2} \\ \vdots & \vdots & & \vdots \\ v_1 & v_2 & \cdots & v_{n-k} \end{bmatrix}.$$

But

$$g(x)\,h(x) = \sum_{i=0}^{n-k} g_i\,x^i \sum_{j=0}^{k} h_j\,x^j = \sum_{i=0}^{n-k}\sum_{j=0}^{k} g_i h_j\,x^{i+j}$$
$$= \sum_{m=0}^{n}\sum_{j=0}^{m} h_j g_{m-j}\,x^m = \sum_{m=0}^{n} v_m\,x^m = x^n - 1\,.$$

The last equality implies that $v_m = 0$ for $0 < m < n$ and thus $\mathbf{G}\,\mathbf{H}^T = \mathbf{0}$ as claimed. QED

Note that the top row of $\mathbf{H}$ can be expressed in polynomial notation as

$$\tilde{h}(x) = x^k\,h(x^{-1}) = h_0 x^k + h_1 x^{k-1} + \ldots + h_k\,.$$

Since $\mathbf{H}$ is the generator matrix of the dual code, it is not difficult to prove the following

Theorem: Let $\mathcal{C}$ be a linear cyclic (n, k) code with generator polynomial $g(x)$. Then the dual code $\mathcal{C}^\perp$ is cyclic as well and is generated by $\tilde{h}(x) = x^k\,h(x^{-1})$, where $h(x) = (x^n - 1)/g(x)$.

Proof: Left as an exercise.

It is often convenient to use a systematic encoder. Computing $c(x) = u(x)\,g(x)$, where $u(x)$ is the information polynomial, will in general not yield $c(x)$ in systematic form. However, the following encoding rule will ensure systematic codewords of the form $\mathbf{c} = (b_0, b_1, \ldots, b_{n-k-1}, u_0, u_1, \ldots, u_{k-1})$.

Systematic encoding rule: Multiply the information polynomial $u(x)$ by x^{n-k} and add $b(x) = \sum_{i=0}^{n-k-1} b_i\, x^i$ such that $c(x) = x^{n-k}\, u(x) + b(x)$ is divisible by $g(x)$, i.e., such that

$$R_{g(x)}[c(x)] = R_{g(x)}[x^{n-k}\, u(x) + b(x)] = R_{g(x)}[x^{n-k}\, u(x)] + b(x) = 0.$$

This requires that

$$b(x) = -R_{g(x)}[x^{n-k}\, u(x)]\,.$$

Note that the systematic encoding rule produces exactly the same set of codewords as any nonsystematic encoding rule. The only difference is that the mapping from $u(x)$ to $c(x)$ is different.

Definition: Let $c(x)$ be a code polynomial and let $v(x)$ be a received (possibly corrupted) code polynomial. Then $e(x) = v(x) - c(x)$ is called the *error polynomial*. It has nonzero coefficients in those locations where channel errors occured.

Construction of BCH and RS Codes

BCH (Bose-Chaudhuri-Hocquenghem) codes and RS (Reed-Solomon) codes are two large classes of cyclic multiple error correcting codes that are used in many practical applications. For small to moderate blocklengths (up to a few thousand symbols) both classes of codes contain good codes (but usually not the best codes known). The main reason for the popularity of the BCH and RS codes is the (relative) ease with which they can be generated and decoded (mostly processing hard decisions) using shift register circuits.

Definition: Given q, $m \geq 1$, and n such that $n \,|\, (q^m - 1)$, let $\beta \in \mathrm{GF}(q^m)$ be any primitive n-th root of unity. Then, for any integer h, the corresponding *BCH code of designed minimum distance* d is the cyclic code of blocklength n with symbols in $\mathrm{GF}(q)$ with generator polynomial

$$g(x) = \mathrm{lcm}\left\{f_h(x), f_{h+1}(x), \ldots, f_{h+d-2}(x)\right\} \ ,$$

where $f_j(x)$ is the minimal polynomial over $\mathrm{GF}(q)$ of β^j. That is, $g(x)$ has the following $d-1$ consecutive powers of β as roots

$$\left\{\beta^h, \beta^{h+1}, \ldots, \beta^{h+d-2}\right\} \ .$$

Note that $g(x)$ usually has more roots than the ones shown above. Often $h = 1$ is selected and then the resulting code is called a *narrow-sense BCH code*. If $n = q^m - 1$, then the BCH code is called a *primitive BCH code*.

Note: In general k (the number of information symbols) is only known after the lcm (least common multiple) operation is performed.

Theorem: **BCH bound**. The BCH codes as defined above satisfy

$$d_{\min} \geq \text{designed minimum distance } d \, .$$

Proof: Will be given later.

Example: Binary $n = 15$, $d = 7$ (designed minimum distance) BCH code. Since $2^4 - 1 = 15 = n$, choose $\beta = \alpha$, where $\alpha \in \mathrm{GF}\,(2^4)$ is a primitive element (with order 15). Setting $h = 1$ and using $d = 7$, a generator polynomial is needed that has the following 6 consecutive powers of α as roots:

$$\{\alpha^1, \alpha^2, \alpha^3, \alpha^4, \alpha^5, \alpha^6\} \, .$$

To make sure that $g(x)$ has all its coefficients in $\mathrm{GF}(2)$ use

$$g(x) = \mathrm{lcm}\{f_1(x), f_2(x), f_3(x), f_4(x), f_5(x), f_6(x)\} = f_1(x)\, f_3(x)\, f_5(x),$$

where

$$\begin{aligned} f_1(x) &= \text{minimal polynomial of } \{\alpha^1, \alpha^2, \alpha^4, \alpha^8\}, \\ f_3(x) &= \text{minimal polynomial of } \{\alpha^3, \alpha^6, \alpha^{12}, \alpha^9\}, \\ f_5(x) &= \text{minimal polynomial of } \{\alpha^5, \alpha^{10}\}. \end{aligned}$$

Since $\deg[f_1(x)] = \deg[f_3(x)] = 4$ and $\deg[f_5(x)] = 2$, $\deg[g(x)] = n - k = 4 + 4 + 2 = 10$ and thus $k = 5$.

To actually compute $g(x)$, the elements of $\mathrm{GF}(2^4)$ have to be generated. Using the primitive polynomial $p(x) = x^4 + x + 1$, the following elements and minimal polynomials are obtained:

Element	$\alpha^3\ \alpha^2\ \alpha^1\ \alpha^0$	Order	Minimal Polynomial
α^0	0 0 0 1	1	$x+1$
α^1	0 0 1 0	15	x^4+x+1
α^2	0 1 0 0	15	x^4+x+1
α^3	1 0 0 0	5	$x^4+x^3+x^2+x+1$
α^4	0 0 1 1	15	x^4+x+1
α^5	0 1 1 0	3	x^2+x+1
α^6	1 1 0 0	5	$x^4+x^3+x^2+x+1$
α^7	1 0 1 1	15	x^4+x^3+1
α^8	0 1 0 1	15	x^4+x+1
α^9	1 0 1 0	5	$x^4+x^3+x^2+x+1$
α^{10}	0 1 1 1	3	x^2+x+1
α^{11}	1 1 1 0	15	x^4+x^3+1
α^{12}	1 1 1 1	5	$x^4+x^3+x^2+x+1$
α^{13}	1 1 0 1	15	x^4+x^3+1
α^{14}	1 0 0 1	15	x^4+x^3+1

The generator polynomial can now be written as

$$\begin{aligned} g(x) &= (1+x+x^4)(1+x+x^2+x^3+x^4)(1+x+x^2) \\ &= 1+x+x^2+x^4+x^5+x^8+x^{10}\,. \end{aligned}$$

The generator matrix for this binary $(15, 5, 7)$ BCH code is

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

To encode the message polynomial $u(x) = 1 + x^2 + x^3$, for example, one can either compute $c(x) = u(x)g(x)$, or $\mathbf{c} = \mathbf{u}\,\mathbf{G}$, where $\mathbf{u} = (1011)$. The result is

$$c(x) = 1+x+x^4+x^6+x^{11}+x^{12}+x^{13} \quad \leftrightarrow \quad \mathbf{c} = (110010100001110).$$

Example: Shortened binary $n = 14$, $d = 7$ BCH code. Start from the $(15, 5, 7)$ code given above with generator polynomial

$$g(x) = 1 + x + x^2 + x^4 + x^5 + x^8 + x^{10} .$$

But now restrict the message polynomial $u(x)$ to have degree less than or equal to 3 so that $c(x) = u(x)g(x)$ will always have $\deg[c(x)] \leq 13$ (and thus $n = 14$). The generator matrix of this shortened binary $(14, 4, 7)$ BCH code is

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

i.e., the same as before, but with the last column and the last row deleted. Note that shortened BCH codes are not cyclic codes anymore in general, but they can be encoded and decoded as if they were cyclic codes.

RS codes can be viewed as a subset of the BCH codes and defined as follows.

Definition: For *RS codes* use the same construction as for BCH codes, but set $m = 1$. Thus, $\beta \in \mathrm{GF}(q)$ and the codewords are over $\mathrm{GF}(q)$.

Note: For RS codes $f_j(x)$ and β^j are both over the same field and thus $f_j(x) = x - \beta^j$. Therefore, **for RS codes**,

$$g(x) = (x - \beta^h)(x - \beta^{h+1}) \cdots (x - \beta^{h+d-2}) .$$

This implies, $\deg g(x) = n - k = d - 1$ from which $k = n - d + 1$ follows. Often $h = 1$ is chosen.

Theorem: RS codes as defined above satisfy

$$d_{\min} = \text{designed minimum distance } d .$$

Proof: From the BCH bound $d_{\min} \geq d$. But for RS codes $d = n - k + 1$ and the Singleton bound states that for any linear code $d_{\min} \leq n - k + 1$. QED

Corollary: RS codes are MDS codes.

Note: Shortened RS codes (n and k reduced by the same amounts) and punctured RS codes (n and $r = n - k$ reduced by the same amounts) are also MDS codes, but they are no longer cyclic codes.

Example: RS code over $\mathrm{GF}(2^4)$ with blocklength $n = 15$ and minimum distance $d_{\min} = 5$. Set $\beta = \alpha$, where $\alpha \in \mathrm{GF}(2^4)$ is a primitive element. Then, setting $h = 1$, compute the generator polynomial as (based on generating $\mathrm{GF}(2^4)$ using the primitive polynomial $x^4 + x + 1$)

$$\begin{aligned} g(x) &= (x - \beta)(x - \beta^2)(x - \beta^3)(x - \beta^4) \\ &= x^4 + \alpha^{13} x^3 + \alpha^6 x^2 + \alpha^3 x + \alpha^{10} . \end{aligned}$$

The resulting code is a $(15, 11, 5)$ RS code over $\mathrm{GF}(2^4)$.

Decoding of BCH and RS Codes

One of the key features of BCH and RS codes is that they are quite easy to decode (up to the designed minumum distance d and using hard decisions from the channel output). Let

$$\begin{aligned} c(x) &= c_0 + c_1\,x + \ldots + c_{n-1}\,x^{n-1}\,, && \text{transmitted codeword,} \\ e(x) &= e_0 + e_1\,x + \ldots + e_{n-1}\,x^{n-1}\,, && \text{error pattern,} \\ v(x) &= v_0 + v_1\,x + \ldots + v_{n-1}\,x^{n-1}\,, && \text{received codeword,} \end{aligned}$$

and assume $v(x) = c(x) + e(x)$. Suppose that the error pattern contains ν errors at locations $\ell_1, \ell_2, \ldots, \ell_\nu$, with $0 \leq \ell_j < n$. Thus

$$e(x) = e_{\ell_1}\,x^{\ell_1} + e_{\ell_2}\,x^{\ell_2} + \ldots + e_{\ell_\nu}\,x^{\ell_\nu}\,,$$

where e_{ℓ_j} are the error values. Note that initially ℓ_j, e_{ℓ_j}, and ν are unknown.

The first step towards determining ℓ_j, e_{ℓ_j}, and ν is to compute the syndromes of the error pattern.

Definition: For an arbitrary cyclic code $\mathcal{C}$ of length n over $\mathrm{GF}(q)$ whose generator polynomial $g(x)$ has zeros $\gamma_1, \gamma_2, \ldots, \gamma_r$, $\gamma_i \in \mathrm{GF}(q^m)$, the *syndromes* $S_i \in \mathrm{GF}(q^m)$ are defined by

$$S_i = v(\gamma_i) = \underbrace{c(\gamma_i)}_{=0} + e(\gamma_i) = \sum_{j=0}^{n-1} e_j \gamma_i^j , \quad i = 1, 2, \ldots, r .$$

To simplify the notation for BCH and RS codes, assume that the generator polynomial $g(x)$ has $2t$ consecutive roots $\beta^1, \beta^2, \ldots, \beta^{2t}$, where $\mathrm{ord}(\beta) = n$. This corresponds to a cyclic t-error correcting code of designed minimum distance $2t + 1$ with parameter $h = 1$. In this case the syndromes are

$$S_i = v(\beta^i) = e(\beta^i) = e_{\ell_1} \beta^{i\ell_1} + e_{\ell_2} \beta^{i\ell_2} + \ldots + e_{\ell_\nu} \beta^{i\ell_\nu}, \quad 1 \le i \le 2t.$$

Note: S_i are the frequency components (GDFT coefficients) of the error vector.

Definition: To obtain a more concise notation, define

$$X_j = \beta^{\ell_j}\,, \quad j = 1, 2, \ldots, \nu\,, \qquad \text{error locators,}$$
$$Y_j = e_{\ell_j}\,, \quad j = 1, 2, \ldots, \nu\,, \qquad \text{error values.}$$

Note: Since β is a primitive n-th root of unity, all error locators are distinct.

With this notation, the syndromes become

$$S_i = v(\beta^i) = Y_1 X_1^i + Y_2 X_2^i + \ldots + Y_\nu X_\nu^i\,,$$

for $i = 1, 2, \ldots, 2t$ and $0 \leq \nu \leq t$. This is a set of $2t$ (nonlinear) equations in the unknowns

$$X_1, \ldots, X_\nu, \quad Y_1, \ldots, Y_\nu, \quad \nu\,.$$

If $\nu \leq t$, then a unique solution exists. However, the original set of nonlinear equations is too difficult to solve directly.

Definition: Define the *error locator polynomial* $\sigma(z)$ as

$$\begin{aligned}\sigma(z) &= (1 - X_1 z)(1 - X_2 z) \cdots (1 - X_\nu z) \\ &= \sigma_0 + \sigma_1 z + \ldots + \sigma_\nu z^\nu , \qquad \sigma_0 = 1 .\end{aligned}$$

Note: The roots of $\sigma(z)$ are X_j^{-1}, $1 \le j \le \nu$, i.e., the **inverses** of the error locators X_j. Clearly, if $\sigma_0, \sigma_1, \ldots, \sigma_\nu$ are known, then the zeros of $\sigma(z)$ and thus the error locators X_j can be found.

The coefficients σ_i and the error locators X_j are related by the following equations (the σ_i are known as *elementary symmetric functions* of the X_j)

$$\begin{aligned}\sigma_0 &= 1 , \\ \sigma_1 &= -(X_1 + X_2 + \ldots + X_\nu) , \\ \sigma_2 &= X_1 X_2 + X_1 X_3 + \ldots + X_{\nu-1} X_\nu , \\ &\vdots \\ \sigma_\nu &= (-1)^\nu X_1 X_2 \cdots X_\nu .\end{aligned}$$

Theorem: **Generalized Newton Identities**. For all integers μ, the syndromes S_i satisfy the recurrence

$$S_{\mu+\nu} + \sigma_1 S_{\mu+\nu-1} + \ldots + \sigma_\nu S_\mu = 0 \,.$$

In particular, for $\mu = 1, 2, \ldots, \nu$,

$$\begin{bmatrix} S_\nu & S_{\nu-1} & \ldots & S_1 \\ S_{\nu+1} & S_\nu & \ldots & S_2 \\ \vdots & \vdots & & \vdots \\ S_{2\nu-1} & S_{2\nu-2} & \ldots & S_\nu \end{bmatrix} \begin{bmatrix} \sigma_1 \\ \sigma_2 \\ \vdots \\ \sigma_\nu \end{bmatrix} = \begin{bmatrix} -S_{\nu+1} \\ -S_{\nu+2} \\ \vdots \\ -S_{2\nu} \end{bmatrix} .$$

Theorem: The matrix of syndromes

$$\mathbf{S}_m = \begin{bmatrix} S_m & S_{m-1} & \cdots & S_1 \\ S_{m+1} & S_m & \cdots & S_2 \\ \vdots & \vdots & & \vdots \\ S_{2m-1} & S_{2m-2} & \cdots & S_m \end{bmatrix},$$

is invertible if $m = \nu \leq t$ and it is singular if $\nu < m \leq t$, where ν is the actual number of errors in the received codeword $v(x)$.

Thus, one strategy to find the actual number ν of errors in a received codeword is to compute $\det(\mathbf{S}_m)$ for $m = t, t-1, t-2$, etc, until the value of $\det(\mathbf{S}_m)$ becomes zero for the first time. Once ν is known, then $\mathbf{S}_\nu^{-1}$ can be computed and the result used to find the coefficients $\sigma_1, \sigma_2, \ldots, \sigma_\nu$ of the error locator polynomial $\sigma(z)$. This procedure is known as the *Peterson-Gorenstein-Zierler (PGZ) decoder*.

A more elegant and efficient procedure, known as the *Berlekamp-Massey (BM) algorithm* uses the generalized Newton identities as the basis for synthesizing a linear feedback shift register (LFSR) circuit that finds both ν and $\sigma_1, \sigma_2, \ldots, \sigma_\nu$.

Once $\sigma(z)$ is known (either from the PGZ decoder or the BM algorithm), the next step is to find the roots of $\sigma(z)$, which are the inverse error locators $X_j^{-1} = \beta^{-\ell_j}$. For polynomials of degree 5 or more, no general closed-form solution method exists for finding the roots. The most common procedure, known as *Chien search*, is to set $z = \beta^0, \beta^{-1}, \ldots, \beta^{-(n-1)}$ and successively try whether any of them is a root of $\sigma(z)$. In practice this is not quite as inefficient as it sounds initially, because BCH and RS codes are often decoded sequentially, symbol by symbol, and thus one root trial operation is performed for each decoded code symbol.

After $\sigma(z)$ and its roots are found, decoding is complete in the case of binary codewords, because all error values in the error locations will be one. For non-binary BCH codes and RS codes, however, the error values $Y_j = e_{\ell_j}$ still need to be determined. The definition of the syndromes can be used to write

$$\begin{bmatrix} S_1 \\ S_2 \\ \vdots \\ S_\nu \end{bmatrix} = \begin{bmatrix} X_1^1 & X_2^1 & \dots & X_\nu^1 \\ X_1^2 & X_2^2 & \dots & X_\nu^2 \\ \vdots & \vdots & & \vdots \\ X_1^\nu & X_2^\nu & \dots & X_\nu^\nu \end{bmatrix} \begin{bmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_\nu \end{bmatrix}.$$

The matrix on the right-hand side is a Vandermonde matrix and thus can be inverted, if ν and the X_j are known, to obtain the Y_j.