Chapter 8

Reed-Solomon and Bose-Chaudhuri-Hocquenghem Codes

Several classes of codes were discussed in the previous chapters, including Hamming codes and the related dual simplex codes, repetition codes and the related dual parity-check codes, CRC codes for error detection, Fire codes as well as further codes for the correction of burst errors. However, up until now we have not found an analytical construction method for really good codes.

In this chapter we will introduce RS and BCH codes, which define two classes of very powerful codes, found around 1960. Even today these codes belong to the best known codes, which are used more and more often for various coded communication systems. The advantages of these codes are summarized below.

- Both code classes can be constructed in an analytically closed way. The RS codes are MDS codes, hence they satisfy the Singleton bound with equality.
- The minimum distance is known and can be easily used as a design parameter. For the RS codes the exact complete weight distribution is known.
- The RS as well as the BCH codes are both very powerful, if the block length is not too big.
- The codes can be adapted to the error structure of the channel, where RS codes are particularly applicable to burst errors and BCH codes are applicable to random single errors.
- Decoding according to the BMD method is easy to accomplish. There are other codes with simpler decoders, but they do not have the other advantages of RS and BCH codes.
- With little additional effort simple soft-decision information (erasure location information) can be used in the decoder.

- The RS and BCH codes are the foundation for understanding many other code classes, which are not mentioned here.

In Section 8.1 we will discuss RS codes as the most important class of MDS codes where also a spectral representation is used from which we will then derive the description with the generator polynomial. In Section 8.2 we will introduce BCH codes as special RS codes where the range of the time domain is restricted to a prime field. The decoding can also be described by spectral methods in a very compact and clear way, although we will need quite some mathematical effort. This will take place in Sections 8.3 to 8.6. The RS and BCH codes will be defined for the general case of $q = p^m$ as well as for the so-called *primitive block length*

$$n = p^m - 1 = q - 1.$$

Thus for RS codes the block length is one symbol smaller than the cardinality of the symbols. In Section 8.7 we will examine the special case of $q = 2^m$ for BCH codes, which is important in practice, and in Section 8.8, we will discuss modifications to the primitive block length.

Generally, we presuppose a Galois field $\mathbb{F}_{p^m}$ with a primitive element z and for the spectral transformation we will use $a(x) \circ\!\!-\!\!\bullet A(x)$ again.

8.1 Representation and Performance of Reed-Solomon (RS) Codes

We will first introduce RS codes by some kind of bandwidth condition in the frequency domain. In Subsection 8.1.2 this will lead us to the derivation of our usual representation with polynomials and matrices. In Subsection 8.1.3, we will get into the weight distribution for MDS codes and thus also for RS codes. In Subsection 8.1.4, the error-correction and error-detection performance will be analytically calculated. These results will be demonstrated by various error probability curves in Subsection 8.1.5.

8.1.1 Definition of RS Codes in the Frequency Domain

Definition 8.1 (RS code). *For arbitrary prime p and integer m and an arbitrary so-called* designed distance $d = d_{\min}$*, a* Reed-Solomon code *with primitive block length is defined as an $(n, k, d_{\min})_q = (p^m - 1, p^m - d, d)_q$ code. Typically, $d = 2t + 1$ is assumed as odd, thus*

$$n - k = d - 1 = 2t \tag{8.1.1}$$

for the number of parity-check symbols. The code consists of all time-domain words $(a_0, \ldots, a_{n-1}) \leftrightarrow a(x)$ with coefficients in $\mathbb{F}_{p^m}$, such that the corresponding

frequency-domain words $(A_0, \ldots, A_{n-1}) \leftrightarrow A(x)$ *are zero in a cyclic sequence of* $n - k = d - 1$ *consecutive positions. These positions are also called* parity frequencies. *An exact comprehensive description requires a further parameter* l*, leading to the form*

$$\mathcal{C} = \Big\{ a(x) \;\Big|\; A(x) = R_{x^n-1}[x^{l+d-1} B(x)] \;\; \textit{with} \;\; \deg\, B(x) \le n - d \Big\}. \tag{8.1.2}$$

Thus $A_l = A_{l+1} = \cdots = A_{l+d-2} = 0$ *for the* $d - 1$ *consecutive parity frequencies and for the remaining* $n - d + 1$ *positions* A_i *can take on arbitrary values. For* $l = 1$*,*

$$\mathcal{C} = \Big\{ a(x) \;\Big|\; a(z^1) = a(z^2) = \cdots = a(z^{d-1}) = 0 \Big\} \tag{8.1.3}$$

and for $l = n + 1 - d$*,*

$$\mathcal{C} = \Big\{ a(x) \;\Big|\; \deg\, A(x) \le n - d \Big\}. \tag{8.1.4}$$

Of course, the other important design parameter apart from p and m is $d = d_{\min}$. However, the value of l is less important since it only means a shift in the frequency domain and thus a "modulation" in the time domain according to (7.5.8). Yet, the right choice of l can bring about certain simplifications for realization, as for example symmetrical generator polynomials. The number of codewords is

$$|\mathcal{C}| = q^k = q^{q-d} = p^{m(p^m - d)}. \tag{8.1.5}$$

Example 8.1. Let $q = 2^m$ and $n = 2^m - 1$. For $d = 2^{m-1}$ the code rate is

$$R \;=\; \frac{k}{n} \;=\; \frac{2^m - 2^{m-1}}{2^m - 1} \;\approx\; \frac{1}{2}.$$

For $m = 8$ we have a $(255, 128, 128)_{256}$ code, which can also be conceived as a binary $(2040, 1024, 128)_2$ code. The number of codewords is $256^{128} = 2^{1024} \approx 10^{308}$, so for a primitive polynomial of degree 8 we already have an enormous number of codewords. Any two byte-viewed codewords differ in at least 128 bytes (or symbols), however, for the binary interpretation the difference is only 128 bits, because two bytes are different, as soon as only one bit of the two 8-bit groups is different. ■

Theorem 8.1. *RS codes are cyclic MDS codes. Hence the designed distance is equal to the minimum distance and* $d_{\min} = d = n - k + 1 = p^m - k$*.*

Proof. "Linearity": is apparent.

"Cyclic": Let $a(x) \in \mathcal{C}$ and $c(x) = R_{x^n-1}[xa(x)]$ be the cyclic shift. According to Theorem 7.8, $C_i = z^i A_i$, thus $C_i = 0 \Leftrightarrow A_i = 0$. Therefore $c(x) \in \mathcal{C}$.

"MDS": Let $d_{\min}$ be the actual minimum distance and $d = n - k + 1$ be the designed distance. We are to prove that $d_{\min} = d$. According to the Singleton

bound of Theorem 4.7, $d_{\min} \leq n-k+1 = d$. Consider $\mathcal{C}$ as in (8.1.4) with $\deg A(x) \leq n-d$. Therefore $A(x) \neq 0$ has a maximum of $n-d$ roots, hence there are a maximum of $n-d$ values z^{-i} with $a_i = -A(z^{-i}) = 0$. Thus $w_H(a(x)) \geq d$ and $d_{\min} \geq d$. For any other l we also have $d_{\min} = d$, since the "modulation" in the time domain does not affect the weights. ■

8.1.2 Polynomial and Matrix Description

Theorem 8.2. *For the RS code as given in (8.1.3) the generator polynomial and the parity-check polynomial are*

$$g(x) = \prod_{i=1}^{d-1}(x - z^i) \quad , \quad h(x) = \prod_{i=d}^{n}(x - z^i). \tag{8.1.6}$$

Proof. For each codeword, $a(z^i) = 0$ with $1 \leq i \leq d-1$. Thus $(x - z^i)$ is a divisor of $a(x)$. So $a(x)$ must be a multiple of $g(x)$. Since $g(x)$ has the degree $d-1 = n-k$, $g(x)$ is the generator polynomial. For the parity-check polynomial $h(x)$, $g(x)h(x) = x^n - 1$ must be valid, which is the case according to (7.2.13). ■

Systematic encoding can be performed with $g(x)$ or $h(x)$ according to the methods discussed in Section 6.4.

A further alternative is given directly by the spectral description, since non-systematic encoding can be accomplished by inverse Fourier transform where the A_i values are presumed as information symbols in the frequency domain. For this realization, the encoder consists of an additive chain of k multipliers operated in separate feedback loops, where the chain is to be run through n times as illustrated in Figure 7.1. So $nk = n^2R$ operations are required and the adder chain has to be computed sequentially. Usually, this IDFT method does not have any significant advantages over the four encoding methods discussed in Section 6.4. However, the IDFT could be simplified in certain cases by using FFT (Fast Fourier Transform) methods.

The equation $A_i = a(z^i) = \sum_{\mu=0}^{n-1} a_\mu z^{i\mu} = 0$ with $1 \leq i \leq d-1$ as in (8.1.3) directly yields a $(n-k, n) = (d-1, n)$-dimensional parity-check matrix $\boldsymbol{H}$:

$$(a_0, \ldots, a_{n-1}) \cdot \underbrace{\begin{pmatrix} 1 & z^1 & z^2 & \ldots & z^{n-1} \\ 1 & z^2 & z^4 & \ldots & z^{2(n-1)} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & z^{d-1} & z^{(d-1)2} & \ldots & z^{(d-1)(n-1)} \end{pmatrix}^T}_{\boldsymbol{H}^T} = (0, \ldots, 0). \tag{8.1.7}$$

However, we do not necessarily need the parity-check matrix for decoding.

Example 8.2. Let $p = 2$, $m = 3$ and therefore $n = 7$. The arithmetic of $\mathbb{F}_{2^3}$ is the same as in Example 7.4. For the designed distance $d = 3$, we have a $(7, 5, 3)_8$ RS code $\mathcal{C} = \{(a_0, \ldots, a_6) \mid A_1 = A_2 = 0\}$. According to Theorem 8.2,

$$\begin{aligned} g(x) &= (x - z)(x - z^2) = x^2 + z^4 x + z^3, \\ h(x) &= (x - z^3)(x - z^4)(x - z^5)(x - z^6)(x - z^7) \\ &= x^5 + z^4 x^4 + x^3 + z^5 x^2 + z^5 x + z^4. \end{aligned}$$

For the generator matrix, according to (6.2.7),

$$\boldsymbol{G} = \begin{pmatrix} z^3 & z^4 & 1 & & & & \\ & z^3 & z^4 & 1 & & & \\ & & z^3 & z^4 & 1 & & \\ & & & z^3 & z^4 & 1 & \\ & & & & z^3 & z^4 & 1 \end{pmatrix}.$$

For the parity-check matrix, according to (6.3.3),

$$\boldsymbol{H}_1 = \begin{pmatrix} 1 & z^4 & 1 & z^5 & z^5 & z^4 & \\ & 1 & z^4 & 1 & z^5 & z^5 & z^4 \end{pmatrix}$$

or according to (8.1.7)

$$\boldsymbol{H}_2 = \begin{pmatrix} 1 & z^1 & z^2 & z^3 & z^4 & z^5 & z^6 \\ 1 & z^2 & z^4 & z^6 & z^1 & z^3 & z^5 \end{pmatrix}.$$

By using elementary row operations $\boldsymbol{H}_1$ can be transformed into $\boldsymbol{H}_2$: multiply row 2 in $\boldsymbol{H}_1$ by z^2 and then add row 2 to row 1:

$$\boldsymbol{H}_3 = \begin{pmatrix} 1 & z^1 & z^2 & z^3 & z^4 & z^5 & z^6 \\ 0 & z^2 & z^6 & z^2 & 1 & 1 & z^6 \end{pmatrix}.$$

Multiplying row 2 with z^2 and adding row 1 to row 2 finally yields $\boldsymbol{H}_2$. This code corrects one error in the 8-ary symbols or in the binary 3-tuples. ■

8.1.3 The Weight Distribution of MDS Codes

Theorem 8.3. *The weight distribution as given in Definition 4.7 can be calculated in an analytically closed way for any arbitrary $(n, k, d_{\min})_q$ MDS code and therefore also for every RS code:*

$$A_r = \binom{n}{r}(q-1)\sum_{j=0}^{r-d_{\min}}(-1)^j\binom{r-1}{j}q^{r-d_{\min}-j} \tag{8.1.8}$$

$$= \binom{n}{r}\sum_{j=0}^{r-d_{\min}}(-1)^j\binom{r}{j}(q^{r-d_{\min}+1-j}-1). \tag{8.1.9}$$

The number of codewords of minimum Hamming weight is

$$A_{d_{\min}} = A_{n-k+1} = \binom{n}{d_{\min}}(q-1).$$

Proof. The proof is mainly based on some lengthy combinatorial considerations, similar to the proof of the MacWilliams identity. Two different forms of the proof of Theorem 8.3 can be found, for instance, in [17, 83]. Here, we will prove the equality of (8.1.8) and (8.1.9), however, we will do without the actual proof of Theorem 8.3 as we did for Theorem 5.8. The identity

$$\binom{r}{j} = \binom{r-1}{j} + \binom{r-1}{j-1}$$

is used to rewrite (8.1.9)

$$\begin{aligned}
A_r &= \binom{n}{r} \sum_{j=0}^{r-d_{\min}} (-1)^j \left[\binom{r-1}{j} + \binom{r-1}{j-1} \right] (q^{r-d_{\min}+1-j} - 1) \\
&= \binom{n}{r} \sum_{j=0}^{r-d_{\min}} (-1)^j \binom{r-1}{j} (q^{r-d_{\min}+1-j} - 1) \\
&\quad + \binom{n}{r} \sum_{j=0}^{r-d_{\min}-1} (-1)^{j+1} \binom{r-1}{j} (q^{r-d_{\min}-j} - 1) \\
&= \binom{n}{r} \sum_{j=0}^{r-d_{\min}} (-1)^j \binom{r-1}{j} (q^{r-d_{\min}+1-j} - q^{r-d_{\min}-j})
\end{aligned}$$

The last equation is identical to (8.1.8). ■

The weight distribution of the $(31, 32-2t, 2t+1)_{31}$ RS codes is shown in Figure 8.1 for some values of t, together with the binary $(31, 26, 3)_2$ Hamming code. In the logarithmically scaled A_r-axis, $A_r = 0$ is symbolized by the value 0.1. Obviously, words of almost maximum weight dominate for RS codes, for instance, the number of words of almost maximum weight with $A_{30} = A_{31} = 1.67 \cdot 10^{43}$ almost corresponds to the total number $|\mathcal{C}| = 32^{29} = 4.46 \cdot 10^{43}$ of all codewords. In other words, almost all codewords of length 31 also have a Hamming weight of 30 or 31. In contrast, for a binary code there can only be one codeword of maximum weight. For the binary Hamming code, most of the words have a medium weight; the maximum is $\max A_l = A_{15} = A_{16} = 8.28 \cdot 10^6$, whereas the code itself has the size $|\mathcal{C}| = 2^{26} = 6.71 \cdot 10^7$.

8.1.4 Calculation of Error-Correction and Error-Detection Probabilities for RS Codes

We presume a hard-decision q-ary abstract DMC with the symbol-error probability p_e as in Definition 1.2. So $1-p_e$ is the probability for the correct reception

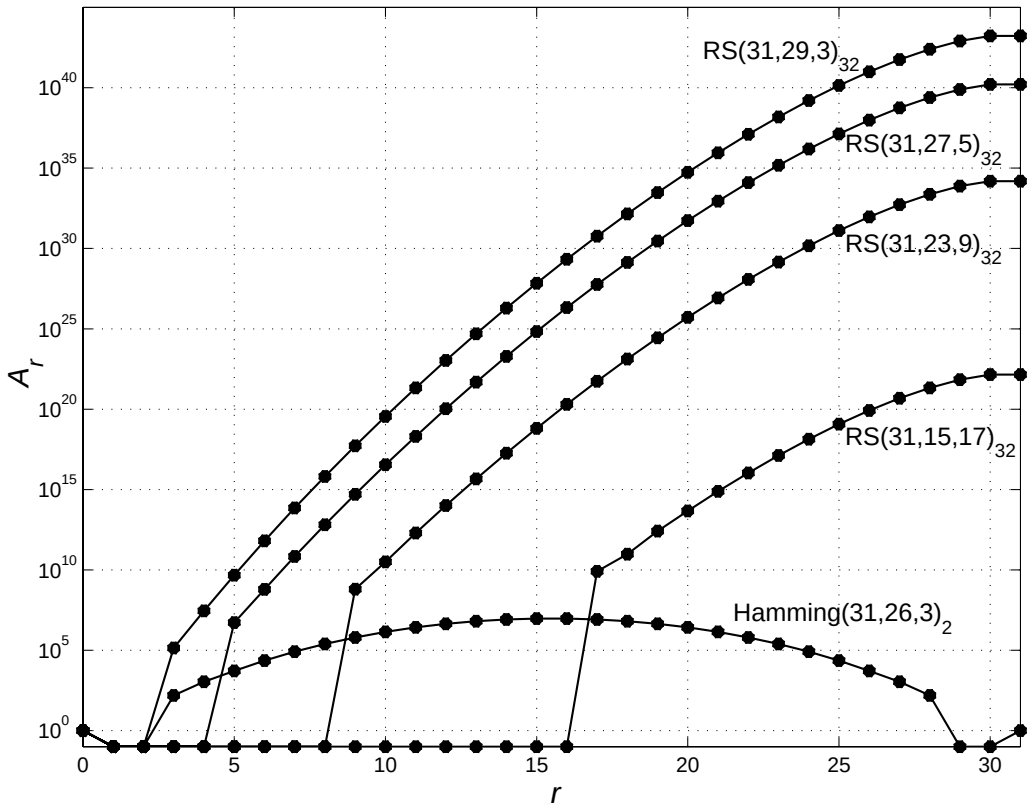

Figure 8.1. Weight distributions of RS and Hamming codes

of a symbol, p_e is the probability for an arbitrary error and $p_e/(q-1)$ is the probability for a specific error. The transmission over the DMC is modeled as $\boldsymbol{y} = \boldsymbol{a} + \boldsymbol{a}$ with the transmitted codeword $\boldsymbol{a} \in \mathcal{C}$ and the error pattern $\boldsymbol{e}$, as known from Chapters 4 and 6.

For a bounded-minimum-distance (BMD) decoder we distinguish between the following post-decoding error probabilities, which all refer to codewords, not to symbols. This is also illustrated in Figure 8.2.

(1) The *probability for correct decoding* is the probability that the received word $\boldsymbol{y}$ is contained in the decoding sphere of radius $t = \lfloor (d_{\min} - 1)/2 \rfloor$ around the transmitted codeword $\boldsymbol{a}$. By using Theorem 4.15 we get

$$P_{cd} = P(\boldsymbol{y} \in K_t(\boldsymbol{a}) \mid \boldsymbol{a}) = P(w_H(\boldsymbol{e}) \leq t) = \sum_{r=0}^{t} \binom{n}{r} p_e^r (1-p_e)^{n-r}. \quad (8.1.10)$$

(2) The *word-error probability* is the probability that $\boldsymbol{y}$ is not contained in the decoding sphere of radius t around $\boldsymbol{a}$:

$$P_w = 1 - P_{cd} = P(w_H(\boldsymbol{e}) > t) = \sum_{r=t+1}^{n} \binom{n}{r} p_e^r (1-p_e)^{n-r}. \quad (8.1.11)$$

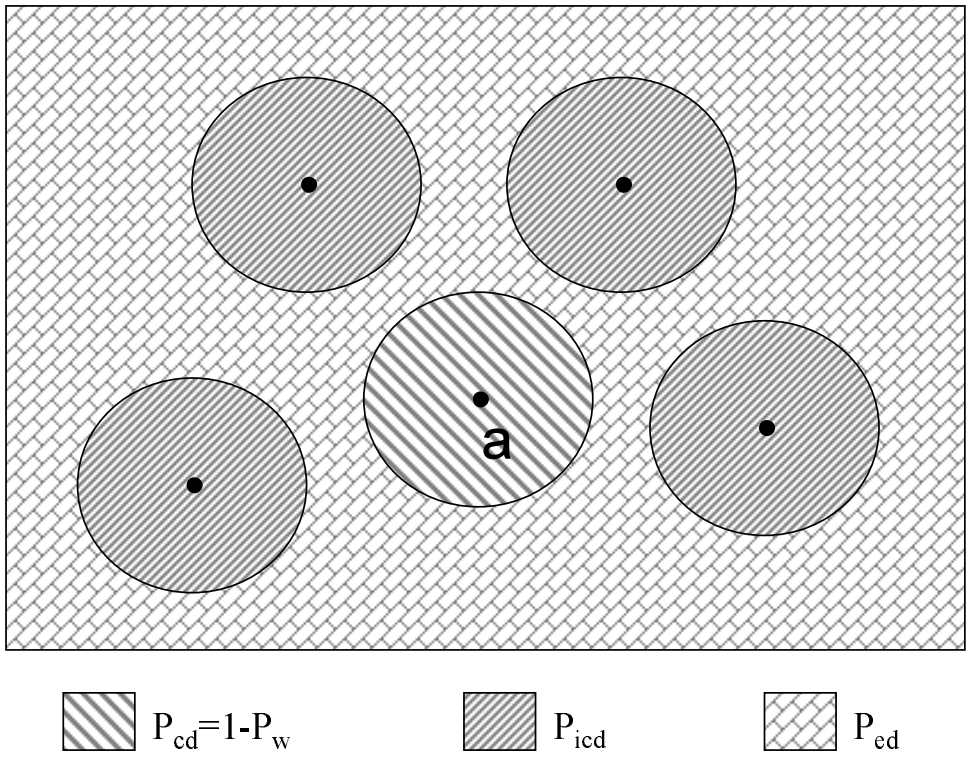

Figure 8.2. Illustration of some post-decoding error probabilities

(3) The *probability of incorrect decoding* is the probability that $\boldsymbol{y}$ is contained in the decoding sphere of radius t around another codeword $\boldsymbol{b}$ unequal to the transmitted codeword $\boldsymbol{a}$:

$$P_{icd} = P\left(\boldsymbol{e} \in \bigcup_{\boldsymbol{b}\in\mathcal{C}\setminus\{\boldsymbol{0}\}} K_t(\boldsymbol{b})\right). \tag{8.1.12}$$

(4) The *probability of error detection* is the probability that $\boldsymbol{y}$ is not contained in any of the decoding spheres of radius t around the codewords:

$$P_{ed} = P\left(\boldsymbol{e} \notin \bigcup_{\boldsymbol{b}\in\mathcal{C}} K_t(\boldsymbol{b})\right). \tag{8.1.13}$$

(5) The *probability of undetected error* was already introduced in Section 4.6 as the probability that $\boldsymbol{y}$ is identical to another codeword, yet unequal to the transmitted codeword:

$$P_{ue} = P(\boldsymbol{e} \in \mathcal{C} \setminus \{\boldsymbol{0}\}). \tag{8.1.14}$$

This probability is only mentioned to complete the list, since it is only relevant purely for error-detection decoding without error-correction.

Obviously,

$$P_{cd} + P_{icd} + P_{ed} = 1, \quad \text{or equivalently} \quad P_{ed} = P_w - P_{icd}. \tag{8.1.15}$$

For the orders of the error probabilities we typically have

$$P_{ue} \ll P_{icd} \ll P_w \approx P_{ed}. \tag{8.1.16}$$

Furthermore, there are also the post-decoding bit error probability P_b and the post-decoding error probability P_{cs} of the encoded q-ary symbols, which are smaller than P_w by a maximum factor of $k \cdot \log_2 q$ and k, respectively, thus $P_w/(k \log 2q) \leq P_b \leq P_w$ and $P_w/k \leq P_{cs} \leq P_w$. In Theorem 4.15 we stated an upper bound for P_{cs}

$$P_{cs} \leq P_{cs,\text{bound}} = \sum_{r=t+1}^{n} \min\left\{1, \frac{r+t}{k}\right\} \binom{n}{r} p_e^r (1-p_e)^{n-r}. \tag{8.1.17}$$

For the derivation of the approximation of P_{cs} in the proof of Theorem 4.15 we used the fact that the number of symbol errors per decoded word is limited to $r + t$ given that the received word contains r errors. This is still exactly true, since on the one hand if the received word lies between the decoding spheres, no decoding takes place and therefore no further symbol errors can occur. On the other hand if the received word is in a wrong decoding sphere, there is a maximum of t further symbol errors. A slightly different bound can be found in [151]. Another approach to the problem of bit-error rate calculations is presented in [131].

In contrast to determining the word-error rate P_w, which is very simple, the calculation of P_{icd}, P_{ed} and P_{ue} requires the knowledge of the weight distribution, which, fortunately, according to Theorem 8.3, can be calculated quite easily for all RS codes. This is why we will discuss the calculation of P_{icd} and P_{ed} in this chapter although it is really only an MDS specificity and does not relate to the algebraic structure of RS codes in any way. In addition, error detection, i.e., detecting that the received word lies between the decoding spheres, can be easily implemented by RS decoders, as we will see later in this chapter.

Theorem 8.4. *We presuppose a q-ary hard-decision DMC with the symbol-error probability p_e. For an $(n, k, d_{\min})_q$ MDS code with the weight distribution $A_0, \ldots, A_n$ and $t = \lfloor (d_{\min} - 1)/2 \rfloor$ the post-decoding probability of incorrect decoding of a received word can be exactly calculated in closed form:*

$$P_{icd} = \sum_{h=d_{\min}}^{n} A_h \sum_{s=0}^{t} \sum_{l=h-s}^{h+s} N(l, s, h) \cdot p(l), \tag{8.1.18}$$

where

$$p(l) = \left(\frac{p_e}{q-1}\right)^l (1-p_e)^{n-l} \tag{8.1.19}$$

and

$$N(l,s,h)=\sum_{r=r_1}^{r_2}\binom{h}{h-s+r}\binom{s-r}{l-h+s-2r}\binom{n-h}{r}(q-2)^{l-h+s-2r}(q-1)^r \tag{8.1.20}$$

with $r_1=\max\{0,l-h\}$ *and* $r_2=\lfloor(l-h+s)/2\rfloor$. *The term* $N(l,s,h)$ *denotes the number of error patterns of weight* l *that are at Hamming distance* s *to a specific codeword* $\boldsymbol{b}$ *of weight* h. *The definition of* $N(l,s,h)$ *is independent of the choice of* $\boldsymbol{b}$. *According to (1.3.10),* $p(l)$ *denotes the probability of a specific error pattern of weight* l.

Proof. For a specific $\boldsymbol{b}\in\mathcal{C}$ with $w_H(\boldsymbol{b})=h$, let

$$N(l,s,h)=\Big|\{\boldsymbol{e}\in\mathbb{F}_q^n \mid w_H(\boldsymbol{e})=l \text{ and } d_H(\boldsymbol{e},\boldsymbol{a})=s\}\Big|,$$

where $h-s\le l\le h+s$ must be valid for $N(l,s,h)>0$. The probability P_{icd} of incorrect decoding is the probability that $\boldsymbol{e}$ is contained in a decoding sphere of radius t around a codeword unequal to the all-zero codeword:

$$\begin{aligned}
P_{icd}&=\sum_{\boldsymbol{b}\in\mathcal{C}\setminus\{\boldsymbol{0}\}}P(\boldsymbol{e}|\boldsymbol{e}\text{ is decoded to }\boldsymbol{b})\\
&=\sum_{\boldsymbol{b}\in\mathcal{C}\setminus\{\boldsymbol{0}\}}\sum_{s=0}^{t}P(\boldsymbol{e}|d_H(\boldsymbol{e},\boldsymbol{b})=s)\\
&=\sum_{\boldsymbol{b}\in\mathcal{C}\setminus\{\boldsymbol{0}\}}\sum_{s=0}^{t}\sum_{l=0}^{n}P(\boldsymbol{e}|w_H(\boldsymbol{e})=l \text{ and } d_H(\boldsymbol{e},\boldsymbol{b})=s)\\
&=\sum_{\boldsymbol{b}\in\mathcal{C}\setminus\{\boldsymbol{0}\}}\sum_{s=0}^{t}\sum_{l=0}^{n}N(l,s,w_H(\boldsymbol{a}))\cdot p(l)\\
&=\sum_{h=d_{\min}}^{n}A_h\sum_{s=0}^{t}\sum_{l=h-s}^{h+s}N(l,s,h))\cdot p(l).
\end{aligned}$$

To complete the proof we are to show the expression (8.1.20) for $N(l,s,h)$. We modify $\boldsymbol{b}$ of Hamming weight h in s positions into $\boldsymbol{e}$ of Hamming weight l. Then we count the number of possibilities which give us the value of $N(l,s,h)$. To modify $\boldsymbol{b}$ into $\boldsymbol{e}$ we divide $\boldsymbol{b}$ into five sectors:

$$\begin{array}{lccccc}
\boldsymbol{a}= & (\neq 0\ldots\neq 0 & \neq 0\ldots\neq 0 & \neq 0\ldots\neq 0 & 0\ldots\ldots 0 & 0\ldots\ldots 0\ )\\
& \text{different} & \text{identical} & \text{different} & \text{different} & \text{identical}\\
\boldsymbol{e}= & (\underbrace{0\ldots\ldots 0}_{v} & \underbrace{\neq 0\ldots\neq 0}_{w} & \underbrace{\neq 0\ldots\neq 0}_{j} & \underbrace{\neq 0\ldots\neq 0}_{r} & \underbrace{0\ldots\ldots 0}_{g}\ )
\end{array}$$

using $g = n - v - w - j - r$ as an abbreviation. The following must be valid:

$$\begin{aligned} h &= w_H(\boldsymbol{a}) = v + w + j \\ l &= w_H(\boldsymbol{e}) = w + j + r \\ s &= d_H(\boldsymbol{a}, \boldsymbol{a}) = v + j + r. \end{aligned} \tag{8.1.21}$$

Let v, w, j, r be arbitrary. For the first h positions, i.e., for the first three sectors, there are $\binom{h}{w}$ possibilities to choose the w identical positions. For the j modifications in the remaining $h - w = v + j$ positions there are $\binom{h-w}{j}(q-2)^j$ ways to modify the non-zero symbols into other non-zero symbols. For the last $n - h = r + g$ positions, i.e., for the last two sectors, there are $\binom{n-h}{r}(q-1)^r$ ways of choosing the r non-zero symbols. All of this leads us to

$$N(l, s, h|v, w, j, r) = \binom{h}{w} \cdot \binom{h-w}{j}(q-2)^j \cdot \binom{n-h}{r}(q-1)^r.$$

The equations (8.1.21) imply that

$$h - s + r = w, \quad l - h + s - 2r = j$$

and therefore

$$N(l, s, h|r) = \binom{h}{h-s+r}\binom{s-r}{l-h+s-2r}\binom{n-h}{r}(q-2)^{l-h+s-2r}(q-1)^r$$

and finally

$$N(l, s, h) = \sum_{r=r_1}^{r_2} N(l, s, h|r).$$

Further considerations lead to $r_1 = \max\{0, l - h\}$ and $r_2 = \lfloor (l - h + s)/2 \rfloor$, so that the bottom numbers of the binomial coefficients are always smaller than or equal to the upper numbers and both numbers are always non-negative. ■

The proof above follows the methods in [95, 131]. Slightly different but equivalent analytical expressions of P_{icd} are derived in [17, 144].

Theorem 8.5. *For the probability of incorrect decoding for a q-ary hard-decision DMC with the symbol-error probability p_e the following approximations can be made. For small p_e,*

$$\frac{P_w}{P_{icd}} \approx \frac{t! \cdot (q-1)^t}{(n-2t)(n-2t+1)\cdots(n-t+1)} \approx t! \cdot \left(\frac{q-1}{n - \frac{3}{2}t}\right)^t \geq t!. \tag{8.1.22}$$

For small p_e and the primitive block length $n = q - 1$ we even have

$$\frac{P_w}{P_{icd}} \approx t!. \tag{8.1.23}$$

For large p_e,

$$P_{icd} \approx q^{-(n-k)} \cdot \sum_{r=0}^{t} \binom{n}{r} (q-1)^r. \tag{8.1.24}$$

Proof. For small p_e,

$$P_w = \sum_{r=t+1}^{n} \binom{n}{r} p_e^r (1-p_e)^{n-r} \approx \binom{n}{t+1} p_e^{t+1}.$$

In the expression for P_{icd} according to (8.1.18), the summands are only dominant at the minimum value of l, because $p(l)$ decreases rapidly for increasing l. According to (8.1.18), minimum l implies minimum h and maximum s, thus $h = 2t+1$, $s = t$, $l = t+1$ as well as $r_1 = r_2 = 0$. Therefore

$$\begin{aligned} P_{icd} &= A_{2t+1} \binom{2t+1}{t+1} \binom{t}{0} \binom{n-2t-1}{0} (q-2)^0 (q-1)^0 p(t+1) \\ &\approx (q-1) \binom{n}{2t+1} \binom{2t+1}{t+1} \left(\frac{p_e}{q-1} \right)^{t+1}, \end{aligned}$$

where A_{2t+1} was determined according to Theorem 8.3 and the approximation $1 - p_e \approx 1$ was used for $p(l)$. The given form of the quotient P_w/P_{icd} is achieved by simple remodeling and the product in the denominator, which consists of t factors, is replaced by the power of t of the middle factor. The inequality on the right hand side of (8.1.22) follows directly from $q - 1 \geq n \geq n - 3t/2$.

For large p_e the worst case is $p_e = (q-1)/q$, since for $p_e/(q-1) = 1-p_e = 1/q$, $p(l) = 1/q^n$ which is independent of l. Thus every specific error pattern occurs with the same probability, so the channel is completely random, which allows the following approximation:

$$\begin{aligned} P_{icd} &= P(\boldsymbol{y} \text{ is contained in the wrong decoding sphere}) \\ &= P(\boldsymbol{y} \text{ is contained in an arbitrary decoding sphere}) \\ &\qquad - P(\boldsymbol{y} \text{ is contained in the correct decoding sphere}) \\ &\approx \frac{\text{cardinality of all decoding spheres of all codewords}}{\text{number of possible words}} \\ &\qquad - P(\boldsymbol{y} \text{ is contained in the correct decoding sphere}) \\ &= \frac{q^k \cdot \sum_{r=0}^{t} \binom{n}{r} (q-1)^r}{q^n} - \sum_{r=0}^{t} \binom{n}{r} \underbrace{p_e^r (1-p_e)^{n-r}}_{= (q-1)^r/q^n} \\ &= \left(\frac{q^k}{q^n} - \frac{1}{q^n} \right) \sum_{r=0}^{t} \binom{n}{r} (q-1)^r. \end{aligned}$$

The approximation $q^k - 1 \approx q^k$ finally leads us to the result of (8.1.22). ■

Both approximations of Theorem 8.5 are very precise, which we can easily verify by the performance curves in the next subsection.

8.1.5 Performance Results for RS Codes over Random-Error Channels

In this subsection we will only take a look at RS codes over the binary modulated AWGN channel with hard decisions, so we assume statistically independent random single errors and the results are displayed over E_b/N_0. One of the great advantages of RS codes is their capability of correcting burst errors, however, we will consider the corresponding performance results not in this chapter but later on in Section 12.?.

The following figures show the performance of RS codes over $q = 2^m$ for various parameter constellations, where the post-decoding error rates are displayed over E_b/N_0 and p_e in the upper and lower subfigures, respectively. Generally, the word-error rate P_w is always given as in (8.1.11) and the probability of incorrect decisions P_{icd} as in (8.1.18), usually over E_b/N_0 in the upper subfigures as well as over the q-ary symbol error rate prior to the decoder p_e in the bottom subfigures. For the binary AWGN channel, i.e., for BPSK modulation, p_e is determined by E_b/N_0 as

$$p_e = 1 - \left(1 - Q\left(\sqrt{\frac{2RE_b}{N_0}}\right)\right)^m , \tag{8.1.25}$$

using (1.3.16), with $m = \log_2 q$ and $M = 1$, as well as (1.3.23).

The following upper subfigures over E_b/N_0 contain not only the graphs for the word-error rate P_w and P_{icd} but also the bit-error rate $p_{b,unc} = Q(\sqrt{2E_b/N_0})$ for uncoded binary signaling. Concerning the bottom subfigures over p_e, the uncoded bit-error rate $p_{b,unc}$ could only be shown over the q-ary symbol-error rate p_e if all RS codes were considered over the same $q = 2^m$, because of $p_{b,unc} = 1-(1-p_e)^{1/m}$, of course. In actual fact, $p_{b,unc}$ over p_e is only given in the bottom part of Figure 8.6.

The first three figures show RS codes with primitive block lengths $n = q-1$ over $q = 32$, $q = 64$ and $q = 256$ for $t = 1, 2, 4, 8$ and partly also for $t = 16$ and $t = 32$, with $k = n - 2t$ accordingly. Note that for $n = 255$ the block length for binary interpretation already amounts to 2040 bits. Independent of q or n, increasing t obviously yields improvements of some dB with regards to E_b/N_0.

Also, P_{icd} rapidly decreases with increasing t, so the probability of undetected errors becomes quite small. For $n = 255$ and $t = 8$, $P_{icd} \leq 2.1 \cdot 10^{-5}$ according to (8.1.23), which can be clearly seen in Figure 8.5. Furthermore $P_{icd} \leq 2.6 \cdot 10^{-14}$ at $t = 16$ as well as $P_{icd} \leq 3.8 \cdot 10^{-37}$ at $t = 32$. For the relation P_w/P_{icd}, the approximation (8.1.22) at $n = 255$ gives us the values $1, 2, 26, 59291$ for $t = 1, 2, 4, 8$ which can also be seen in Figure 8.5.

In Figure 8.6, just for $n = 255$, P_w is not combined with P_{icd}, but instead with the post-decoding q-ary symbol-error rate P_{cs}, however, it is not P_{cs} itself but its upper bound $P_{cs,\text{bound}}$ as given in (8.1.17) that is shown. The probability P_{cs} can only be smaller than P_w by a maximum factor of k, yet, the upper bound only reduces it by a factor of $(2t + 1)/(255 - 2t)$, i.e., $3/253 \approx 0.01$ at $t = 1$ and $65/191 \approx 0.34$ at $t = 32$. Although the post-decoding bit-error or symbol-error rates differ by a factor between 10 and 100, this difference becomes less important as the slope gets steeper.

Comparing Figures 8.2, 8.3 and 8.4 for fixed t, we can see that increasing q means a minor deterioration of P_w and P_{icd}. This is not surprising, because as the code rate $R = (n - 2t)/n$ increases, the number of correctable symbols remains the same. Although the block length also increases in symbols as well as in bits, according to the channel coding theorem, the increase can not compensate for the growing R.

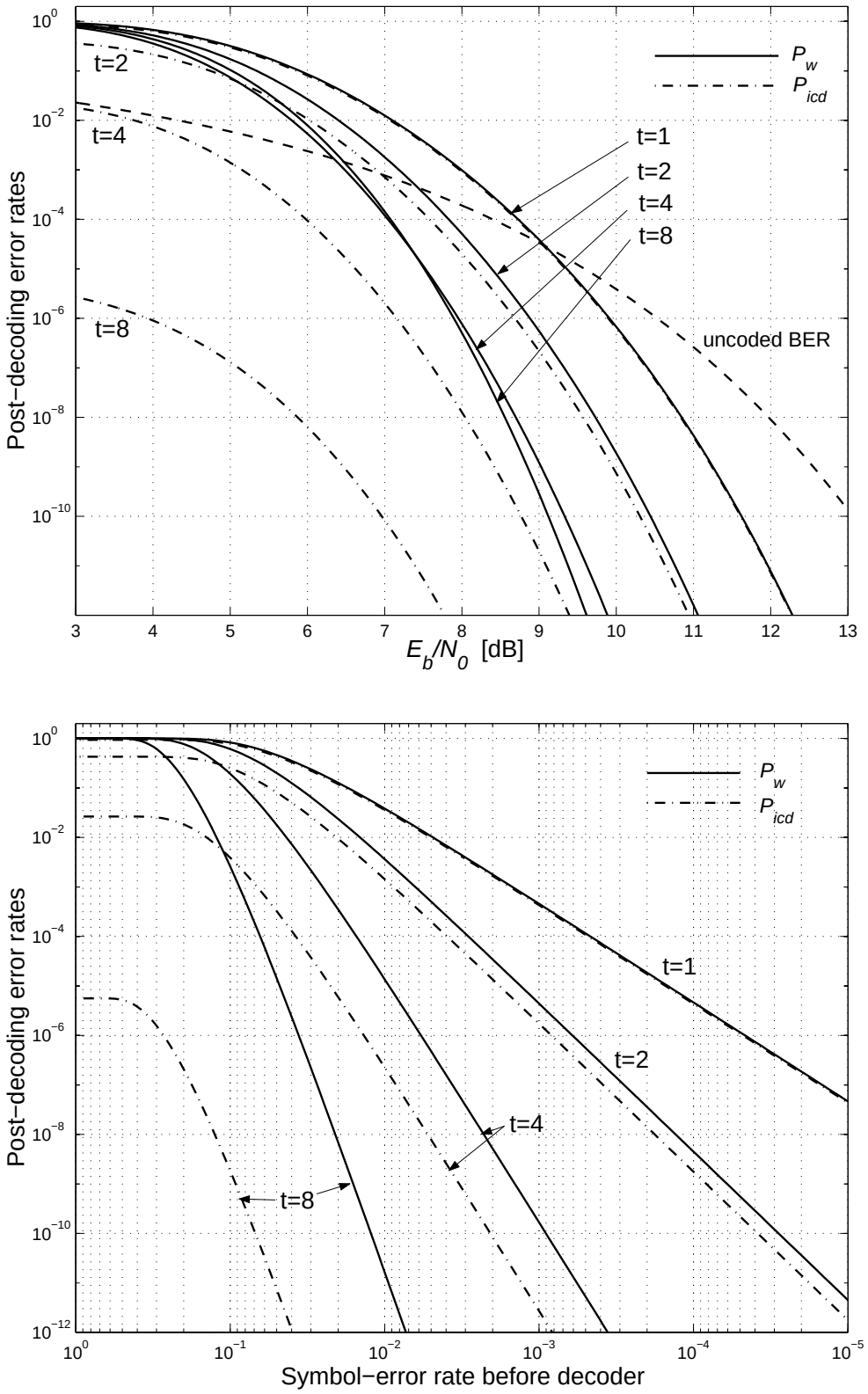

Figure 8.3. P_w and P_{icd} of $(31, 31-2t, 2t+1)_{32}$-RS codes for several t

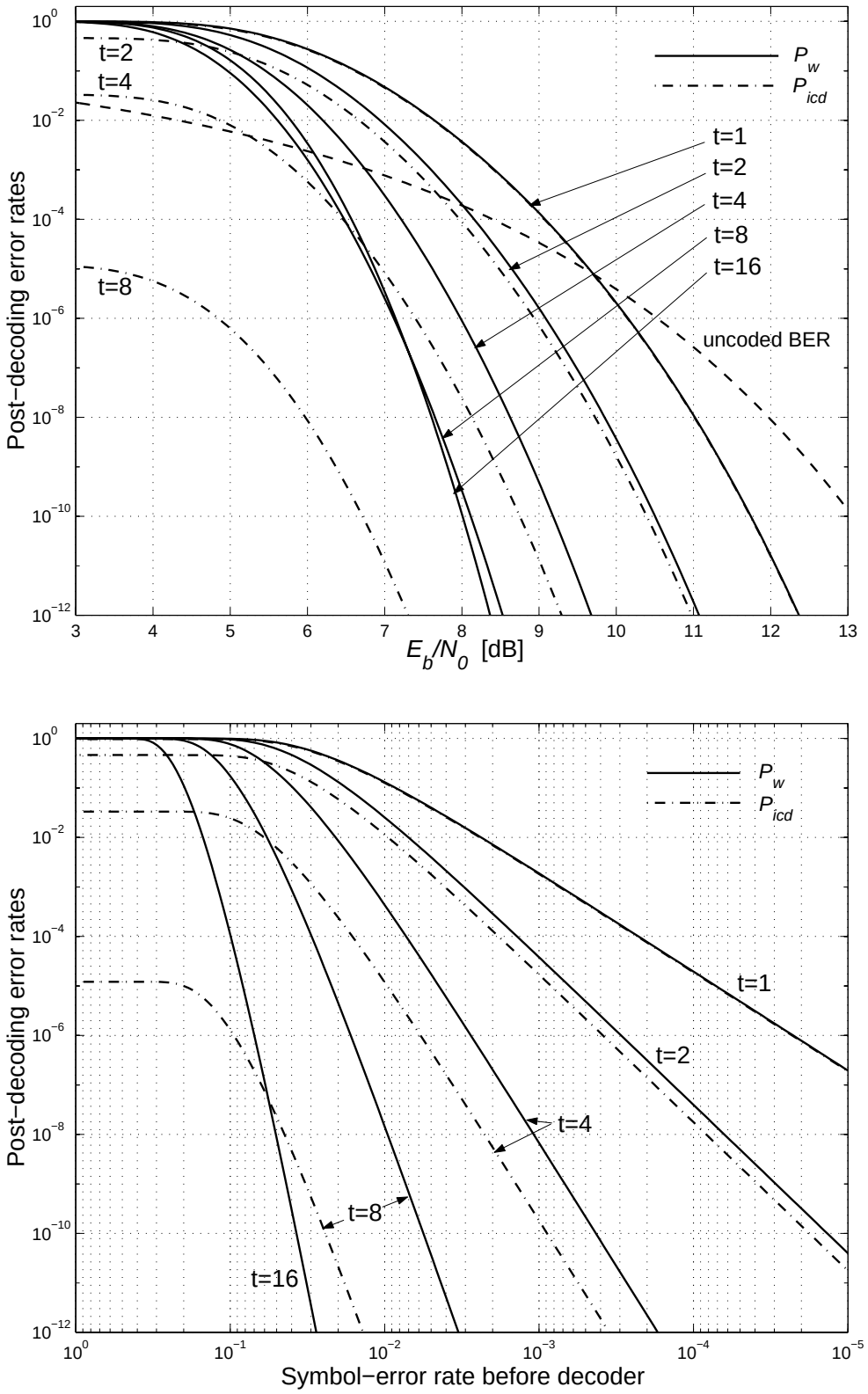

Figure 8.4. P_w and P_{icd} of $(63, 63-2t, 2t+1)_{64}$-RS codes for several t

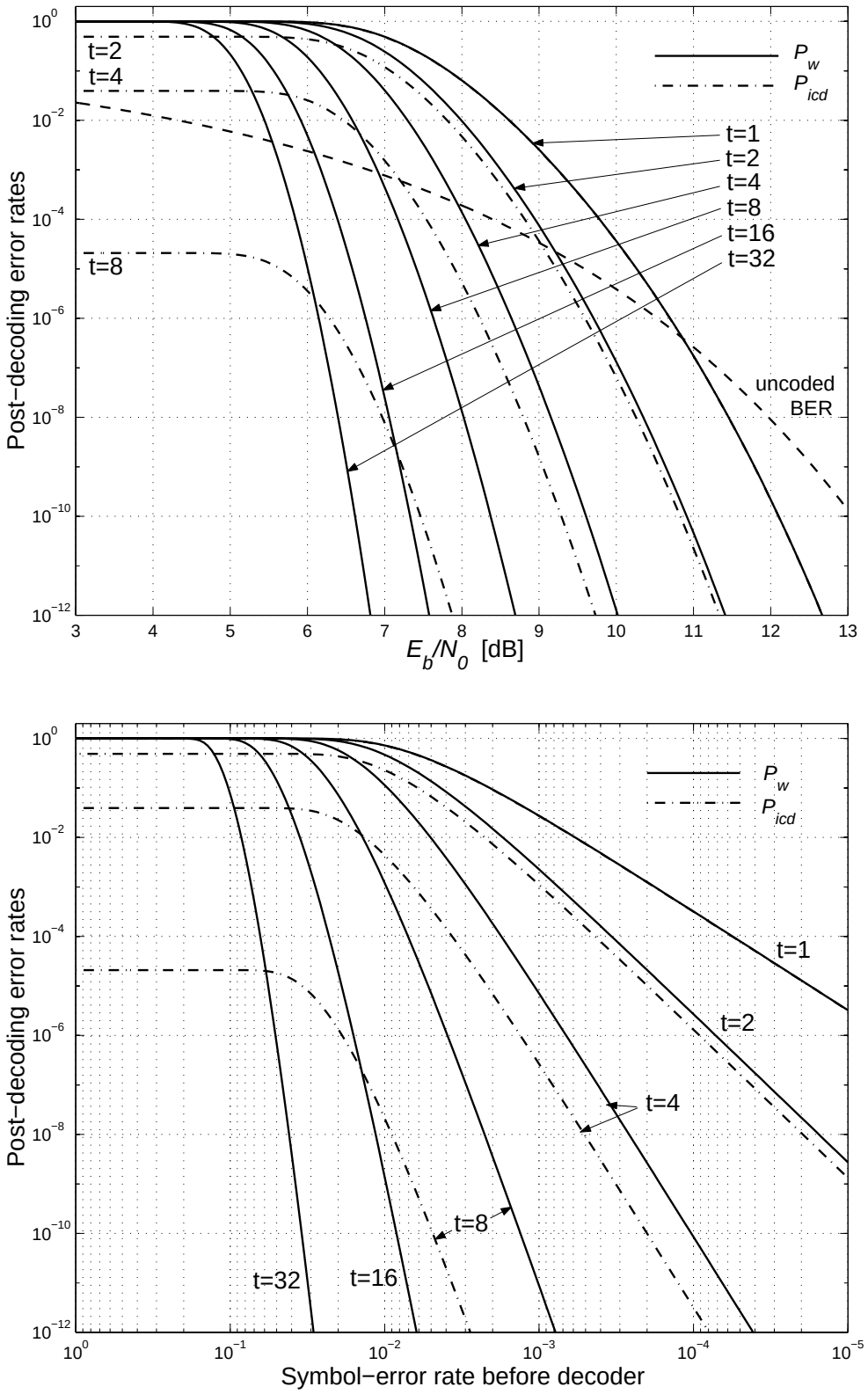

Figure 8.5. P_w and P_{icd} of $(255, 255-2t, 2t+1)_{256}$-RS codes for several t

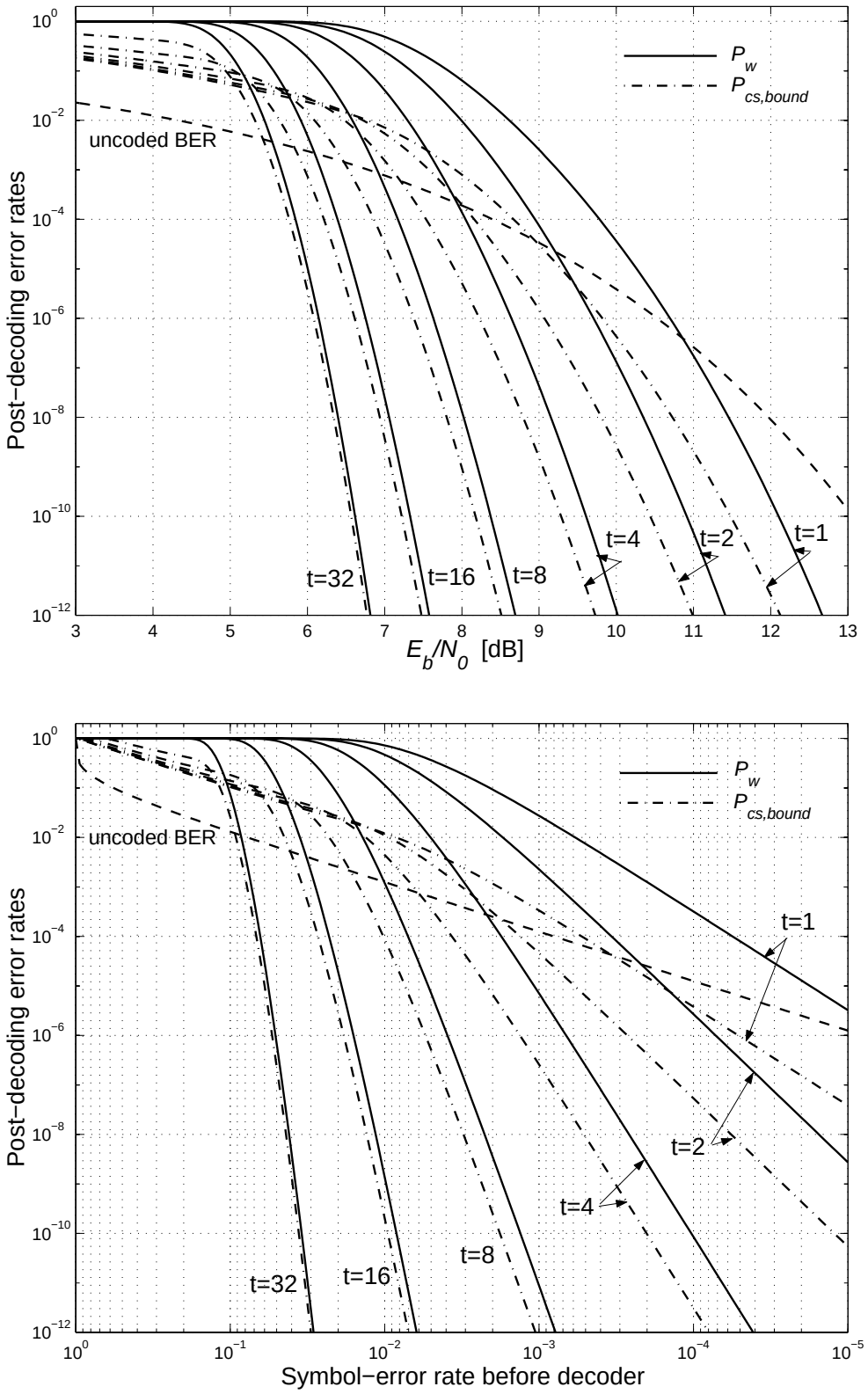

Figure 8.6. P_w and P_b of $(255, 255-2t, 2t+1)_{256}$-RS codes for several t

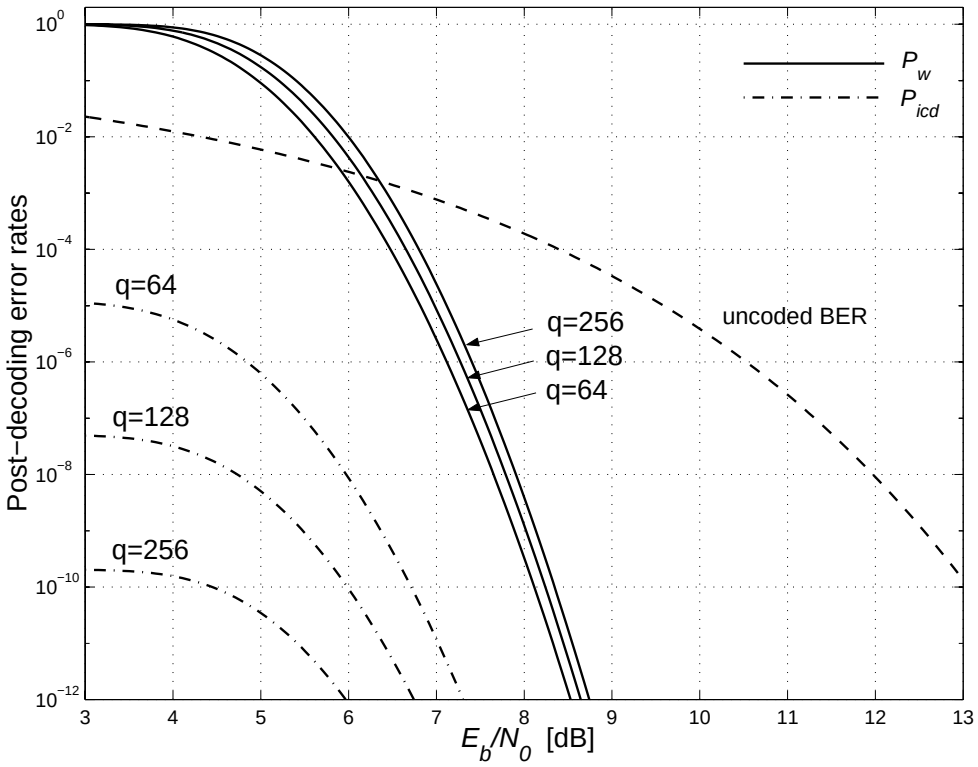

Figure 8.7. P_w and P_{icd} of $(63, 47, 17)_q$-RS codes for several q

In Figure 8.7 we consider shortened RS codes with $n = 63 < q = 64, 128, 256$. As we will see later in Theorem 8.14, shortened MDS codes are still MDS codes. Thus $n-k = 2t$ is still valid and P_w and P_{icd} can be calculated as usual. For fixed $n = 63$, $k = 47$ and $t = 8$, we increase q and therefore also the binary block length. The effect is a minor deterioration of P_w over E_b/N_0, which is solely caused by the increasing q-ary symbol-error rate p_e, as a q-ary symbol requires more binary transmissions for increasing q and is therefore more vulnerable to errors. However, over p_e, P_w is fully independent of q, which is obvious looking at (8.1.11). In contrast to P_w, P_{icd} undergoes a significant improvement for increasing q.

In Figure 8.8 we examine the shortening under different circumstances. We have $q = 256$ and $t = 8$ fixed, but the block length is $n = 53\lambda + 2t$, so a code block transports either 1, 2, 3 or 4 ATM cells [**?**]. Small λ implies a smaller block length and a decreasing code rate and thus worse bandwidth efficiency, however, the power efficiency, i.e., P_w over E_b/N_0, remains almost the same. Over p_e (bottom subfigure) instead of over E_b/N_0 (upper subfigure), the dependence on the code rate does not exist, so small λ is the most suitable, because the code rate is small without paying the penalty of worsening signal-to-noise ratio.

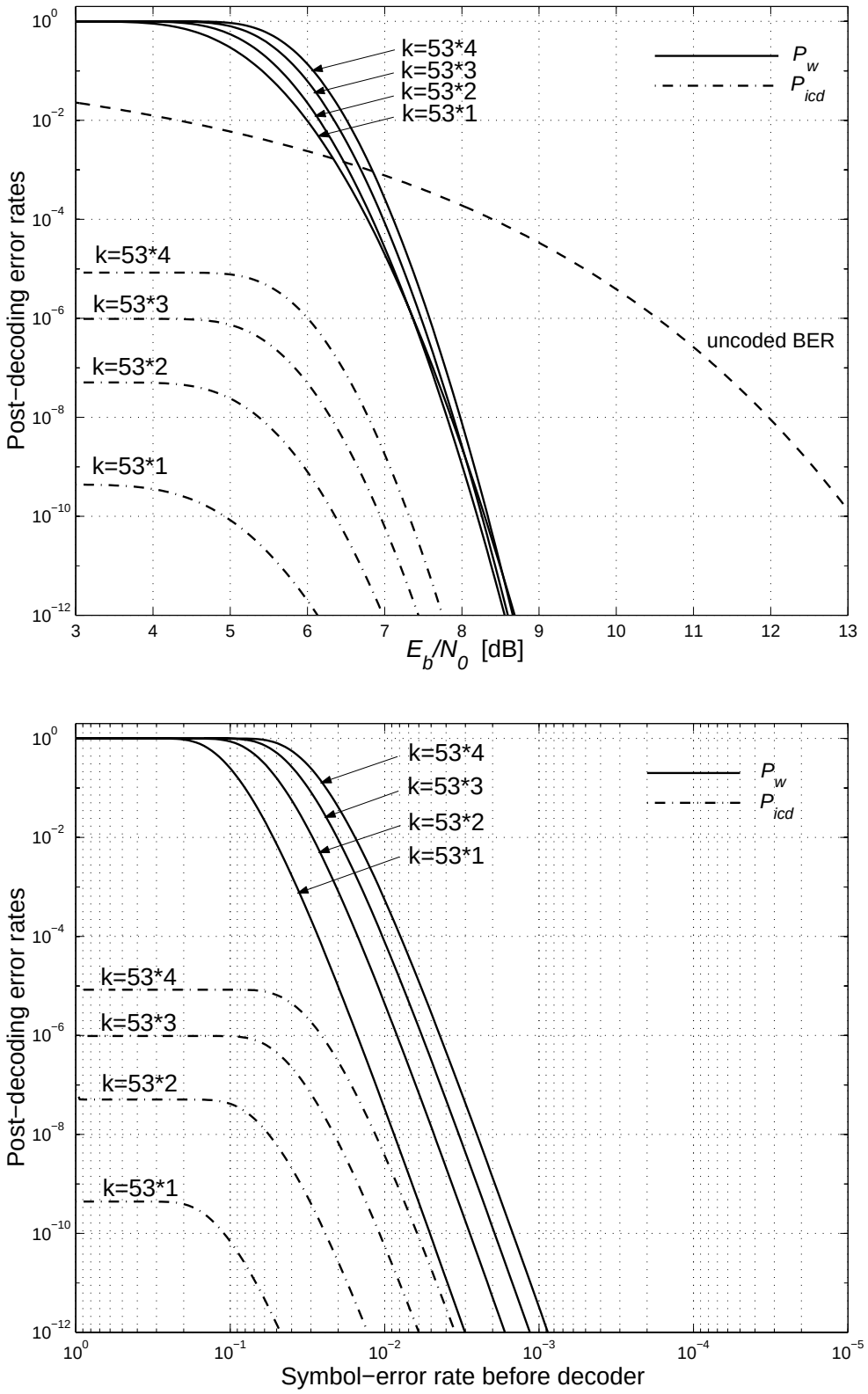

Figure 8.8. P_w and P_{icd} of $(53\lambda + 16, 53\lambda, 17)_{256}$-RS codes for $\lambda = 1, 2, 3, 4$

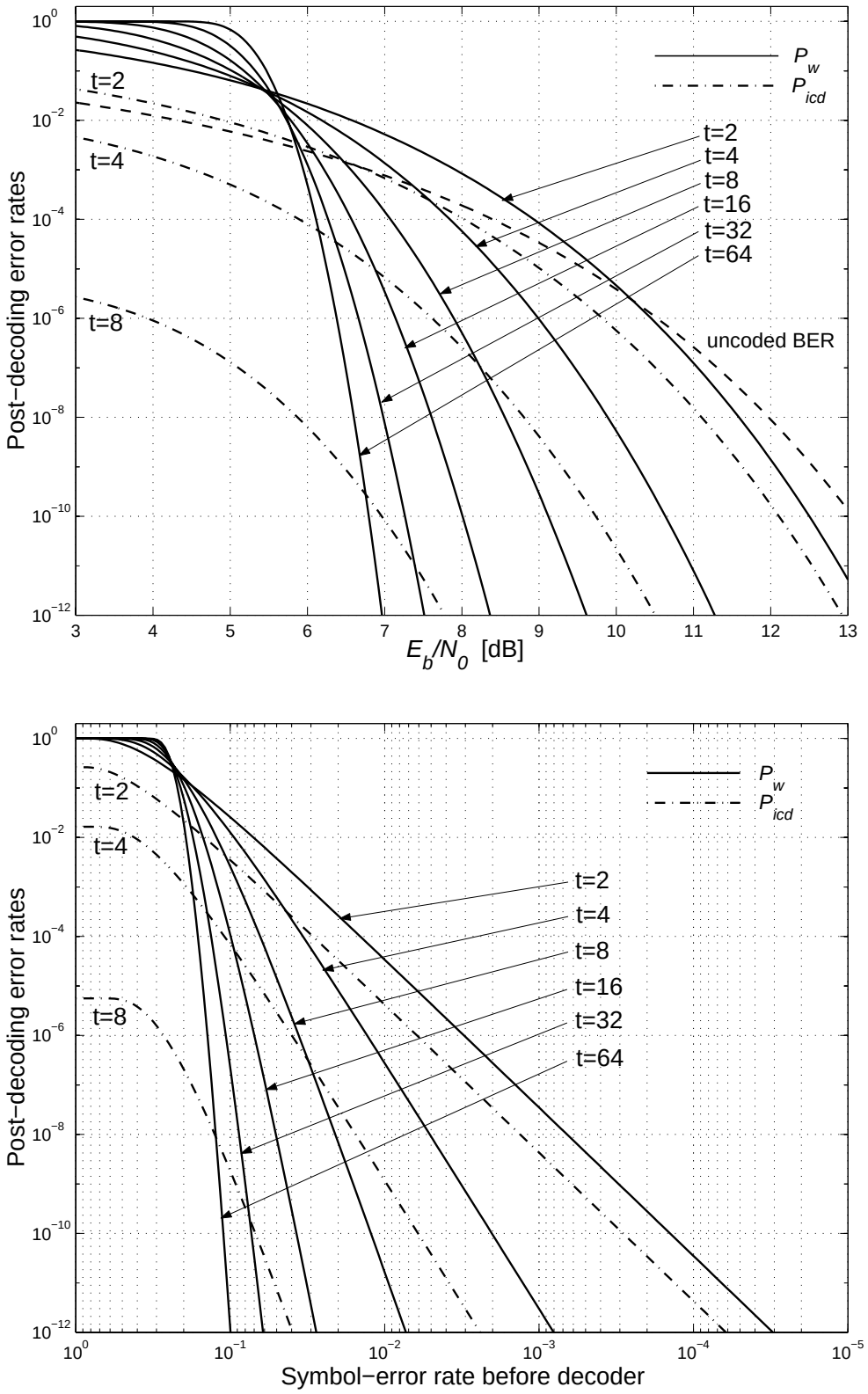

Figure 8.9. P_w and P_{icd} of $(4t-1, 2t-1, 2t+1)_{4t}$-RS codes for several t

Finally, Figure 8.9 shows a comparison of non-shortened RS codes again. We consider RS codes over $q = 8, 16, 32, 64, 128, 256$ with code rate $R = 1/2$ in the form $(4t - 1, 2t - 1, 2t + 1)_{4t}$. Obviously, we gain a lot thanks to increasing block length and thus to increasing complexity. However, at $t = 64$ the performance curve is already very steep with less than 1 dB between almost error free transmission and total failure. We will discuss the practical side of such curves in Section 10.?.

8.2 Representation and Performance of Bose-Chaudhuri-Hocquenghem (BCH) Codes

We will first introduce BCH codes as a subset of RS codes. Some examples and interesting special cases are considered in Subsection 8.2.2. Tables and performance curves of very high practical relevance are presented in Subsection 8.2.3. Finally, the asymptotic properties of BCH codes and further features are considered in Subsection 8.2.4.

8.2.1 Definition of BCH Codes as Subsets of RS Codes

BCH codes emerge from RS codes by adopting the Fourier transform in $\mathbb{F}_{p^m}$ and the block length $n = p^m - 1$, but the code symbols are taken from the prime field $\mathbb{F}_p$ instead of $\mathbb{F}_{p^m}$. For the typical case of $p = 2$ the code symbols are bits and not groups of bits, so BCH codes are normal binary codes for $p = 2$.

Definition 8.2 (BCH codes). *For arbitrary prime p and integer m and an arbitrary designed distance d, a* Bose-Chaudhuri-Hocquenghem code *is defined as an $(n, k, d_{\min})_p$ code with the primitive block length $n = p^m - 1 = q - 1$ and the relations*

$$d_{\min} \geq d \quad , \quad k \leq n + 1 - d_{\min} \leq n + 1 - d. \tag{8.2.1}$$

Thus $n - k \geq 2t$ for $d = 2t + 1$. The code consists of all time-domain words $(a_0, \ldots, a_{n-1}) \leftrightarrow a(x)$ with coefficients in $\mathbb{F}_p$, such that the corresponding frequency-domain words $(A_0, \ldots, A_{n-1}) \leftrightarrow A(x)$ are zero in a cyclic sequence of at least $d - 1$ consecutive positions. Usually the parity frequencies are presumed at $1, \ldots, d - 1$ and the resulting code

$$\mathcal{C} = \left\{ a(x) \in \mathbb{F}_p[x] \;\middle|\; a(z^1) = \cdots = a(z^{d-1}) = 0 \right\} \tag{8.2.2}$$

is then called a narrow-sense BCH code. *As for the RS codes, the exact description of general BCH codes requires a further parameter l, where the parity frequencies are defined by $A_l = A_{l+1} = \cdots = A_{l+d-2} = 0$ for the $d-1$ consecutive positions.*

From the given designed distance d we do not directly obtain the information block length k or the code rate R nor the actual minimum distance $d_{\min}$, but have to calculate these parameters first. The parameter l for determining the parity frequencies is of greater importance for BCH codes than for RS codes, since for BCH codes l might even influence $d_{\min}$. However, typically the narrow-sense BCH codes as in (8.2.2) are used, because then the number of parity bits is usually at its minimum.

In contrast to (8.1.3) with $a(x) \in \mathbb{F}_{p^m}[x]$ for RS codes, $a(x) \in \mathbb{F}_p[x]$ is required in (8.2.2) for BCH codes, both with degrees $\leq n-1$. So a BCH code is a subset of an RS code, thus $d_{\min,BCH} \geq d_{\min,RS} = d$ and the Singleton bound implies (8.2.1). The relation $d_{\min} \geq d$ is also called *BCH bound*. Usually, $d_{\min}$ is not calculated exactly, since the gap between $d_{\min}$ and d might remain unknown for most applications. In some cases even $d_{\min} = d$ is valid (see [83, 105]). The exact calculation of the information block length k (or the dimension of the code) is performed by the generator polynomial as follows.

Theorem 8.6. *Let z be a primitive element of the Galois field $\mathbb{F}_{p^m}$. For the designed distance d, an $(n, k, d_{\min})_p$ BCH code with $d_{\min} \geq d$ is created by the generator polynomial*

$$\begin{aligned} g(x) &= \mathrm{LCM}\Big(f_{[z^1]}(x), \ldots, f_{[z^{d-1}]}(x)\Big) \\ &= \prod_{b \in \mathcal{M}} (x-b) \quad \textit{with} \quad \mathcal{M} = \bigcup_{i=1}^{d-1} [z^i], \end{aligned} \tag{8.2.3}$$

where $[z^i] = \{z^{ip^0}, z^{ip^1}, z^{ip^2}, \ldots\}$ denotes the set of conjugates as introduced in Definition 7.3, i.e., all conjugates of z^i with respect to $\mathbb{F}_p$. The corresponding minimal polynomial is

$$f_{[z^i]}(x) = \prod_{b \in [z^i]} (x-b) = \prod_{r=0}^{|[z^i]|-1} (x - z^{ip^r}). \tag{8.2.4}$$

So, the generator polynomial $g(x)$ is the least common multiple (LCM) of the minimal polynomials of $z^1, \ldots, z^{d-1}$. Thus, for the information block length

$$\begin{aligned} k = n - \deg\, g(x) &= n - \left| \bigcup_{i=1}^{d-1} [z^i] \right| \\ &\geq n - \sum_{i=1}^{d-1} |[z^i]|. \end{aligned} \tag{8.2.5}$$

Proof. The coefficients of the minimal polynomials, and thus also of the generator polynomial, are taken from the prime field $\mathbb{F}_p$. So with the information polynomial, the codeword polynomial also has its coefficients in $\mathbb{F}_p$. For each

combination of two minimal polynomials of (8.2.3), the polynomials are either relatively prime or identical. As for the RS codes, the equation $a(z^i) = A_i = 0$ implies that $g(x)$ must consist of the linear factors $(x - z^i)$ or that $g(x)$ must be a multiple of $\prod_{i=1}^{d-1}(x - z^i)$. According to Theorem 7.9, the requirement that the coefficients of $a(x)$ must be in $\mathbb{F}_p$ is equivalent to $a(z^i)^p = a(z^{ip})$ for $0 \le i \le n-1$. Thus for $1 \le i \le d-1$,

$$\begin{aligned} 0 &= a(z^i) \\ 0 &= a(z^i)^p &&= a(z^{ip}) \\ 0 &= a(z^{ip})^p &&= a(z^{ip^2}) \\ 0 &= a(z^{ip^2})^p &&= a(z^{ip^3}) \quad \ldots \end{aligned}$$

So all $(x - z^{ip^r})$ with $1 \le i \le d-1$ and $r = 0, 1, 2, \ldots$ must be linear factors of $g(x)$, but only once for each factor. According to (8.2.3), $g(x)$ is reducible into just these factors. ■

For this representation, the BCH codes are defined as subsets of RS codes which almost makes the BCH bound $d_{\min} \ge d$ seem trivial. However, there are other concepts for introducing BCH codes. For example, if the BCH codes are defined by the generator polynomial $g(x)$ with coefficients in $\mathbb{F}_p$ and its roots are $z^1, \ldots, z^{d-1}$, then the proof of the BCH bound is quite time-consuming [144].

To make things a little easier, we introduced BCH codes with a slight restriction in this chapter, as we did with the algebraic foundations in Chapter 7 where we only discussed the extension of a prime field $\mathbb{F}_p$ to a Galois field $\mathbb{F}_{p^m}$. However, the concept of field extensions over primitive polynomials can be generalized to the extension of $\mathbb{F}_{p^m}$ to $\mathbb{F}_{p^{mr}}$, where r is an integer, which gives us BCH codes with the primitive block length $p^{mr} - 1$ and p^m-ary symbols. For the binary interpretation with $p = 2$, the block length is simply $m(2^{mr} - 1)$. These BCH codes

- are especially suitable for channels with burst errors of length m, in other words, the coding scheme can be adjusted to the error structure of the channel quite precisely;
- include the RS codes as a special case for $r = 1$, hence, BCH codes are a generalization of RS codes and vice versa.

However, since almost all BCH codes used in practice are binary with the block length $2^m - 1$ and $r = 1$, the restriction chosen in Definition 8.2 seems to be reasonable under practical aspects. Thus BCH codes are still special RS codes.

The weight distribution of most BCH codes is not known analytically. BCH codes for correcting up to 3 errors are an exception however, since formulas to calculate the weight distribution are well known [79, 144].

8.2.2 Examples and Special Cases

Example 8.3. As in Example 8.2, let $p = 2$, $m = 3$ and thus $n = 7$. For the designed distance $d = 3$ we have a narrow-sense BCH code with the generator polynomial

$$g(x) = \mathrm{LCM}\Big(f_{[z^1]}(x), f_{[z^2]}(x)\Big) = f_{[z]}(x) = x^3 + x + 1,$$

since, according to Example 7.4, $[z^1] = \{z^1, z^2, z^4\} = [z^2]$ with $f_{[z]}(x) = x^3 + x + 1$. The BCH code turns out to be a cyclic $(7, 4, 3)_2$ Hamming code, when compared to 6.3. For better understanding, we will now take a closer look at the representation in the frequency domain. According to Definition 8.2, $A_1 = A_2 = 0$ and according to Theorem 7.9, $A_i^2 = A_{2i \bmod 7}$, as already used for the proof of Theorem 8.6. Thus,

$$\begin{array}{ll}
A_0 = A_{2\cdot 0} = A_0^2 & \text{implies that } A_0 \in \{0, 1\} \\
A_3 = A_{2\cdot 5} = A_5^2 & \\
A_6 = A_{2\cdot 3} = A_3^2 & A_3 \text{ determines } A_6 \\
A_5 = A_{2\cdot 6} = A_6^2 = A_3^4 & A_3 \text{ determines } A_5 \\
A_1 = A_{2\cdot 4} = A_4^2 & A_1 = 0 \text{ per definition} \\
A_2 = A_{2\cdot 1} = A_1^2 & A_2 = 0 \text{ per definition} \\
A_4 = A_{2\cdot 2} = A_2^2 & \text{implies that } A_4 = 0.
\end{array}$$

Therefore for the codewords in the frequency domain

$$\mathcal{C} \quad \circ\!\!-\!\!\bullet \quad \{(A_0, 0, 0, A_3, 0, A_3^4, A_3^2) \mid A_0 \in \mathbb{F}_2, A_3 \in \mathbb{F}_8\}.$$

With $A_0 \in \mathbb{F}_2$ (1 bit) and $A_3 \in \mathbb{F}_{2^3}$ (3 bits), 4 bits can be given arbitrarily in the frequency domain, which, of course, correspond to the $k = 4$ information bits in the time domain. Table 8.1 lists the corresponding frequency words to the 16 codewords of the cyclic Hamming (or BCH) code of Example 6.1. Obviously $A_0 \in \mathbb{F}_2$, $A_3 \in \mathbb{F}_8$, $A_6 = A_3^2$ and $A_5 = A_6^2$ are satisfied and each of the 16 frequency words of the form $(A_0, 0, 0, A_3, 0, A_3^4, A_3^2)$ is presented in the table. For the cyclic shift from row to row, A_i is always multiplied by z^i. ■

Example 8.4. Comparison of the error-correction ability of RS and BCH codes. There exists a $(15, 11, 5)_{16}$ RS code that can correct 2 symbol errors. In the binary interpretation we have a $(60, 44, 5)_2$ code, where

$$\begin{array}{ll}
\boldsymbol{e} = 0000\ 1111\ 1111\ 0000\ 0000\ \ldots & \text{is correctable} \\
\boldsymbol{e} = 0000\ 0001\ 1111\ 1000\ 0000\ \ldots & \text{is not correctable} \\
\boldsymbol{e} = 0000\ 0001\ 0010\ 1000\ 0000\ \ldots & \text{is not correctable.}
\end{array}$$

Each burst error up to length 5 can be corrected, for burst errors of length 6, 7 and 8, 75%, 50% and 25% can be corrected, respectively. However, 3 random

single errors can not be corrected. So the minimum distance for the binary interpretation is still 5, however, in special cases some slightly larger values are possible but not guaranteed.

According to Table 8.2, there exists a $(63, 45, 7)_2$ BCH code with comparable code rate, which can always correct 3 random single errors. For a channel with burst errors, the RS code is better than the BCH code. However, for a channel with statistically independent random single errors the BCH code is better than the RS code. ■

Special case. In Theorem 6.10 we defined CRC codes by the generator polynomial $g(x) = (1 + x)p(x)$ with a primitive polynomial $p(x)$. For $p(x) = f_{[z]}(x)$ CRC codes turn out to be special BCH codes with the form

$$g(x) = \mathrm{LCM}\Big(f_{[z^0]}(x), f_{[z^1]}(x)\Big) = (x + 1)p(x) \tag{8.2.6}$$

for the designed distance $d = 3$. Thus $d_{\min} \geq 3$. In the proof of Theorem 6.10 $d_{\min}$ was determined to be even, so now we can exactly prove that $d_{\min} \geq 4$ or $d_{\min} = 4$. ■

For binary BCH codes with $d = 2t + 1$ the advantages of the form given in (8.2.2) become immediately clear: $a(z^t) = 0$ implies that $a(z^{d-1}) = a(z^{2t}) = a(z^t)^2 = 0$. For narrow-sense BCH codes with $l = 1$ (beginning at z^1) at least the set $z^1, \ldots, z^{2t}$ of roots is required. In contrast to shifted parity frequencies with $l = 0$ (beginning at z^0) at least the set $z^0, \ldots, z^{2t}$ of roots is required,

Table 8.1. Time-domain codewords and corresponding frequency-domain words for the $(7, 4, 3)_2$ Hamming (or BCH) code

$a(x)$							$A(x)$						
0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	1	1	1	1	1	1	1	0	0	0	0	0	0
1	1	0	1	0	0	0	1	0	0	z^4	0	z^2	z
0	1	1	0	1	0	0	1	0	0	1	0	1	1
0	0	1	1	0	1	0	1	0	0	z^3	0	z^5	z^6
0	0	0	1	1	0	1	1	0	0	z^6	0	z^3	z^5
1	0	0	0	1	1	0	1	0	0	z^2	0	z	z^4
0	1	0	0	0	1	1	1	0	0	z^5	0	z^6	z^3
1	0	1	0	0	0	1	1	0	0	z	0	z^4	z^2
1	1	1	0	0	1	0	0	0	0	z^6	0	z^3	z^5
0	1	1	1	0	0	1	0	0	0	z^2	0	z	z^4
1	0	1	1	1	0	0	0	0	0	z^5	0	z^6	z^3
0	1	0	1	1	1	0	0	0	0	z	0	z^4	z^2
0	0	1	0	1	1	1	0	0	0	z^4	0	z^2	z
1	0	0	1	0	1	1	0	0	0	1	0	1	1
1	1	0	0	1	0	1	0	0	0	z^3	0	z^5	z^6

thus a further factor $x-1$ occurs in the generator polynomial which reduces the dimension k unnecessarily. This only really makes sense for CRC codes, as seen previously.

Special case: We consider binary BCH codes for the correction of one error, so $t = 1$, $d = 3$ and $n = 2^m - 1$. Again, let $p(x)$ be a primitive polynomial and z be a primitive element of $\mathbb{F}_{p^m}$. Because of $z^{2^m} = z^{n+1} = z$, we have

$$[z] = \{z, z^2, z^4, z^8, \ldots, z^{2^{m-1}}\} \quad , \quad |[z]| = m$$

for the set of conjugates of z. Thus $[z] = [z^2]$, so $p(x) = f_{[z]}(x) = f_{[z^2]}(x)$ for the minimal polynomials. According to Theorem 8.6, the generator polynomial is

$$g(x) = \mathrm{LCM}\Big(f_{[z^1]}(x), f_{[z^2]}(x)\Big) = f_{[z]}(x) = \prod_{i=0}^{m-1}(x - z^{2^i}). \tag{8.2.7}$$

The last equation follows from (7.3.6). Now, according to (8.2.5),

$$k = n - \deg\, g(x) = (2^m - 1) - m,$$

so the binary BCH codes for correcting one error are $(2^m - 1, 2^m - m - 1, 3)_2$ codes, which correspond to the cyclic binary Hamming codes of order m, as the comparison to Theorem 5.10 reveals. The binary Hamming codes can also be described by

$$\mathcal{C} = \{\boldsymbol{a} \in \mathbb{F}_p^n \mid a(z) = 0\}$$

because for coefficients in the prime field, $a(z) = 0$ directly implies that $a(z^2) = a(z)^2 = 0$. ■

Parity-check matrices of BCH codes have a special feature which we will examine for the case of $p = 2$ and $d = 2t + 1$. As for RS codes in (8.1.7), each row of $\boldsymbol{H}$ also has the form $z^0, z^i, z^{i\cdot 2}, \ldots, z^{i\cdot(n-1)}$ for BCH codes, where i takes on each of the $n - k$ exponents in $[z^1] \cup \ldots \cup [z^{2t}]$. However, with the condition $a(x) \in \mathbb{F}_2[x]$, i only has to take on the values $1, 2, 3, \ldots, 2t$. Because of

$$z^2 \in [z^1],\ z^4 \in [z^1],\ z^6 \in [z^3],\ z^8 \in [z^1],\ z^{10} \in [z^5], \ldots$$

i can be further restricted to the t values $1, 3, 5, \ldots, 2t - 1$, i.e., the dimension of the parity-check matrix is reduced from $(n - k, n)$ to (t, n) or to (mt, n) for the binary interpretation. However, since mt can be greater than $n - k$ (see Theorem 8.8), the reduced binary parity-check matrix may still have linearly dependent rows, as we will demonstrate in Example 8.5.

Example 8.5. The next subsection contains the Tables 8.1 and 8.2, which have been mentioned so often and list the existing BCH codes and their error correction ability. The following examples show how these tables were created. Therefore we will now consider various BCH codes with $n = 15$, where the

arithmetic of $\mathbb{F}_{2^4}$ is based on the primitive element z with $z^4+z+1=0$ as in Example 7.5.

(1) The designed distance $d=3$ implies the generator polynomial

$$g(x)=\mathrm{LCM}\Big(f_{[z^1]}(x),f_{[z^2]}(x)\Big)=f_{[z]}(x)=x^4+x+1,$$

because $[z^1]=\{z^1,z^2,z^4,z^8\}=[z^2]$. The result is the $(15,11,3)_2$ Hamming code. Since $d=3\le d_{\min}\le w_H(g(x))=3$, $d=d_{\min}$ as well as $n-k=mt$.

(2) The designed distance $d=5$ implies the generator polynomial

$$g(x)=\prod_{b\in\mathcal{M}}(x-b)\quad\text{with}\quad\mathcal{M}=[z^1]\cup[z^2]\cup[z^3]\cup[z^4].$$

We have $[z^1]=\{z^1,z^2,z^4,z^8\}=[z^2]=[z^4]$ and $[z^3]=\{z^3,z^6,z^{12},z^{24}=z^9\}$, thus $\mathcal{M}=[z]\cup[z^3]$. Therefore the generator polynomial is

$$\begin{aligned}g(x)&=f_{[z]}(x)\cdot f_{[z^3]}(x)\\&=\underbrace{(x-z^1)(x-z^2)(x-z^4)(x-z^8)}_{x^4+x+1}\cdot\underbrace{(x-z^3)(x-z^6)(x-z^9)(x-z^{12})}_{x^4+x^3+x^2+x+1}\\&=x^8+x^7+x^6+x^4+1,\end{aligned}$$

which gives us a $(15,7,5)_2$ code. Since $d=5\le d_{\min}\le w_H(g(x))=5$, $d=d_{\min}$ as well as $n-k=mt$ again. The parity-check matrix is reduced from

$$\boldsymbol{H}=\begin{pmatrix}1&z^1&z^{1\cdot2}&\dots&z^{1\cdot14}\\1&z^2&z^{2\cdot2}&\dots&z^{2\cdot14}\\1&z^3&z^{3\cdot2}&\dots&z^{3\cdot14}\\1&z^4&z^{4\cdot2}&\dots&z^{4\cdot14}\end{pmatrix}$$

to

$$\boldsymbol{H}=\begin{pmatrix}1&z^1&z^{1\cdot2}&\dots&z^{1\cdot14}\\1&z^3&z^{3\cdot2}&\dots&z^{3\cdot14}\end{pmatrix}.$$

For the binary interpretation this is a $(2\cdot4,15)=(n-k,n)$-dimensional matrix.

(3) The designed distance $d=7$ implies the generator polynomial

$$g(x)=\prod_{b\in\mathcal{M}}(x-b)\quad\text{with}\quad\mathcal{M}=[z^1]\cup[z^2]\cup[z^3]\cup[z^4]\cup[z^5]\cup[z^6].$$

Since $[z^1]=[z^2]=[z^4]$ and $[z^3]=[z^6]$, $\mathcal{M}=[z^1]\cup[z^3]\cup[z^5]$ thus

$$\begin{aligned}g(x)=&\underbrace{f_{[z]}(x)\cdot f_{[z^3]}(x)}_{x^8+x^7+x^6+x^4+1}\cdot\underbrace{f_{[z^5]}(x)}_{x^2+x+1}\\&=x^{10}+x^8+x^5+x^4+x^2+x+1,\end{aligned}$$

which gives us a $(15, 5, 7)_2$ code with $d = d_{\min}$ as well as $n-k = 10 < 4 \cdot 3 = mt$. The reduced parity-check matrix is

$$\boldsymbol{H} = \begin{pmatrix} 1 & z^1 & z^{1\cdot 2} & \dots & z^{1\cdot 14} \\ 1 & z^3 & z^{3\cdot 2} & \dots & z^{3\cdot 14} \\ 1 & z^5 & z^{5\cdot 2} & \dots & z^{5\cdot 14} \end{pmatrix},$$

which is a $(12, 15)$-dimensional matrix for binary interpretation. A $(15, 5)_2$ code has a $(10, 15)$-dimensional binary parity-check matrix, so that two linearly dependent rows of the 12 rows of the binary $\boldsymbol{H}$ can be eliminated to get the standard form.

(4) The designed distance $d = 9$ implies the generator polynomial

$$g(x) = \prod_{b \in \mathcal{M}} (x - b) \quad \text{with} \quad \mathcal{M} = \bigcup_{i=1}^{8} [z^i].$$

Since $[z^1] = [z^2] = [z^4] = [z^8]$ and $[z^3] = [z^6]$, $\mathcal{M} = [z^1] \cup [z^3] \cup [z^5] \cup [z^7]$ and therefore

$$g(x) = \underbrace{f_{[z]}(x) \cdot f_{[z^3]}(x) \cdot f_{[z^5]}(x)}_{x^{10}+x^8+x^5+x^4+x^2+x+1} \cdot \underbrace{f_{[z^7]}(x)}_{x^4+x^3+1} = \frac{x^{15}-1}{x-1}$$

$$= x^{14} + x^{13} + x^{12} + x^{11} + x^{10} + x^9 + x^8 + x^7 + x^6 + x^5 + x^4 + x^3 + x^2 + x + 1.$$

which gives us a $(15, 1, 15)_2$-repetition code, i.e., 7 instead of simply 4 errors can be corrected. So there are binary BCH codes of length 15 simply for correcting 1, 2, 3 or 7 errors.

These 3 codes (except for the repetition code) can also be taken from the following Tables 8.1 and 8.2. If the parity frequencies are moved, codes with different generator polynomial may emerge. ■

8.2.3 Tables and Performance Results on BCH Codes

Table 8.2 lists the binary narrow-sense BCH codes for the correction of t errors with all primitive block lengths from $n = 7$ to $n = 1023$ [95, 105, 144, 151]. Of course, $2t + 1 \le d \le d_{\min} \le n - k + 1$.

Table 8.3 shows the generator polynomials in octal coding for some of the BCH codes listed in Table 8.2. For the $(15, 7)_2$ BCH code with $t = 2$, for example, we have $g(x) = 721_{\text{octal}} = 111\ 010\ 001_{\text{dual}} = x^8 + x^7 + x^6 + x^4 + 1$.

Table 8.2. t-error-correcting $(n, k)_2$ BCH codes (from [105])

n	k	t	n	k	t	n	k	t	n	k	t	n	k	t
7	4	1	255	199	7	511	322	22	1023	913	11	1023	443	73
				191	8		313	23		903	12		433	74
15	11	1		187	9		304	25		893	13		423	75
	7	2		179	10		295	26		883	14		413	77
	5	3		171	11		286	27		873	15		403	78
				163	12		277	28		863	16		393	79
31	26	1		155	13		268	29		858	17		383	82
	21	2		147	14		259	30		848	18		378	83
	16	3		139	15		250	31		838	19		368	85
	11	5		131	18		241	36		828	20		358	86
	6	7		123	19		238	37		818	21		348	87
				115	21		229	38		808	22		338	89
63	57	1		107	22		220	39		798	23		328	90
	51	2		99	23		211	41		788	24		318	91
	45	3		91	25		202	42		778	25		308	93
	39	4		87	26		193	43		768	26		298	94
	36	5		79	27		184	45		758	27		288	95
	30	6		71	29		175	46		748	28		278	102
	24	7		63	30		166	47		738	29		268	103
	18	10		55	31		157	51		728	30		258	106
	16	11		47	42		148	53		718	31		248	107
	10	13		45	43		139	54		708	34		238	109
	7	15		37	45		130	55		698	35		228	110
				29	47		121	58		688	36		218	111
127	120	1		21	55		112	59		678	37		208	115
	113	2		13	59		103	61		668	38		203	117
	106	3		9	63		94	62		658	39		193	118
	99	4					85	63		648	41		183	119
	92	5	511	502	1		76	85		638	42		173	122
	85	6		493	2		67	87		628	43		163	123
	78	7		484	3		58	91		618	44		153	125
	71	9		475	4		49	93		608	45		143	126
	64	10		466	5		40	95		598	46		133	127
	57	11		457	6		31	109		588	47		123	170
	50	13		448	7		28	111		578	49		121	171
	43	14		439	8		19	119		573	50		111	173
	36	15		430	9		10	127		563	51		101	175
	29	21		421	10					553	52		91	181
	22	23		412	11	1023	1013	1		543	53		86	183
	15	27		403	12		1003	2		533	54		76	187
	8	31		394	13		993	3		523	55		66	189
				385	14		983	4		513	57		56	191
255	247	1		376	15		973	5		503	58		46	219
	239	2		367	16		963	6		493	59		36	223
	231	3		358	18		953	7		483	60		26	239
	223	4		349	19		943	8		473	61		16	247
	215	5		340	20		933	9		463	62		11	255
	207	6		331	21		923	10		453	63			

Table 8.3. Generator polynomials for t-error-correcting $(n, k)_2$ BCH codes (from [221])

n	k	t	$g(x)$ octal
7	4	1	13
15	11	1	23
	7	2	721
	5	3	2467
31	26	1	45
	21	2	3551
	16	3	107657
	11	5	5423325
	6	7	313365047
63	57	1	103
	51	2	12471
	45	3	1701317
	39	4	166623567
	36	5	1033500423
	30	6	157464165547
	24	7	17323260404441
	18	10	1363026512351725
127	120	1	211
	113	2	41567
	106	3	11554743
	99	4	3447023271
	92	5	624730022327
	85	6	130704476322273
	78	7	26230002166130115
	71	9	6255010713253127753
	64	10	1206534025570773100045
	57	11	335265252505705053517721
	50	13	54446512523314012421501421
255	247	1	435
	239	2	267543
	231	3	156720665
	223	4	75626641375
	215	5	23157564726421
	207	6	16176560567636227
	199	7	7633031270420722341
	191	8	2663470176115333714567
	187	9	52755313540001322236351
	179	10	22624710717340432416300455
	171	11	15416214212342356077061630637
	163	12	75004155100756025515747245146 01
	155	13	37575130054076650157225064646776 33
	147	14	16421301735371655253041653054410117 11
	139	15	46140173206017556157072273024745356744 5
	131	18	215713331471510151261250277442142024165471

For the entries of Tables 8.1 and 8.2 the designed distance $d = 2t + 1$ was presumed. The actual minimum distance $d_{\min}$ is unknown and may even be much larger. However, most of the decoding methods for BCH codes can not profit from a larger $d_{\min}$, since the decoding is only based on the positions of the $d-1 = 2t$ sequential parity frequencies. Yet, in some cases we can calculate $d_{\min}$ exactly, as Theorem 8.7 proves.

Theorem 8.7. *Suppose an $(n, k)_p$ BCH code with the primitive block length $n = p^m - 1$. If the designed distance d is a divisor of n, then $d_{\min} = d$.*

Proof. Let $n = d \cdot \beta$. The sum (5.1.2) of the finite geometric series can be denoted by

$$x^n - 1 = x^{d\beta} - 1 = (x^\beta - 1) \cdot \underbrace{\left(1 + x^\beta + x^{2\beta} + \cdots + x^{(d-1)\beta}\right)}_{= a(x)}.$$

For $i = 1, \ldots, d-1$, the relation $0 < i\beta < n$ is valid and therefore $z^{i\beta} \neq 1$. Since z^i is a root of $x^n - 1$ but not of $x^\beta - 1$, $a(z^i) = 0$ must be satisfied. Since $a(x) \in \mathbb{F}_p[x]$ and $\deg a(x) \leq n - 1$, $a(x)$ is a codeword of Hamming weight d. Thus $d_{\min} \leq d$. Finally, the BCH bound $d_{\min} \geq d$ leads to $d_{\min} = d$. ■

According to Table 8.2, $n - k = m \cdot t$ for small t. Typically, the required number of parity-check bits is upper bounded by the minimum number of errors to be corrected:

Theorem 8.8. *For a binary $(n, k)_2$ BCH code with the primitive block length $n = 2^m - 1$, which can correct at least t errors,*

$$n - k \leq m \cdot t. \tag{8.2.8}$$

Proof. For the designed distance $d = 2t + 1$, $n - k = \deg g(x)$ is equal to the cardinality of $\mathcal{M} = \bigcup_{i=1}^{d-1} [z^i] = \bigcup_{i=1}^{t} \left([z^{2i-1}] \cup [z^{2i}]\right)$. In $\mathbb{F}_{2^m}$, z^i and z^{2i} are conjugates, thus $[z^i] = [z^{2i}]$. So the conjugacy class $[z^{2i}]$ can be omitted and $\mathcal{M}$ is reduced to $\mathcal{M} = \bigcup_{i=1}^{t} [z^{2i-1}]$. According to (6.3.5), $|[z^{2i-1}]| \leq m$ and therefore $|\mathcal{M}| \leq t \cdot m$. ■

According to (7.3.1) with $t \approx \frac{d_{\min}}{2}$ implies that $1 - R \leq \frac{m}{2} \cdot \frac{d_{\min}}{n}$. So if approximate equality was reached, then obviously $d_{\min}/n \to 0$ is implied for $n \to \infty$ or $m \to \infty$. Consequently, according to Section 3.4, BCH codes are asymptotically bad. A complete proof can be found in [83].

Figures 8.9, 8.10 and 8.11 show the bit-error rate over the binary modulated AWGN channel for binary BCH codes at rates of approximately 1/2, 1/4 and

3/4 for various block lengths between 15 and 1023. BMD decoding with hard decisions is assumed, such that the curves can be calculated according to (4.7.4). Obviously a larger block length implies a smaller error rate. The comparison of Figures 8.9, 8.10 and 8.11 suggests $R \approx 1/2$ as the best available code rate. To make this really clear, Figure 8.13 illustrates some BCH codes with various code rates and a constant block length of $n = 255$. According to the Shannon theory, small code rates are best (for example, according to Figure 3.6 a transition from $R = 1/2$ to $R = 1/10$ can achieve a gain of approximately 1 dB), but obviously the construction method for BCH codes requires medium code rates for best results.

Both parts of Figure 8.14 show the same BCH codes with $n = 255$ as Figure 8.13. However, the bottom part now restricts the bit-error rate to a maximum of 10^{-10} for easier comparison with Figures 8.9 to 8.11. In addition the top part of Figure 8.14 shows the word-error rates. According to Theorem 4.15 or (1.7.2), we have

$$P_b \approx \frac{d_{\min}}{k} \cdot P_w,$$

so the differences between P_b and P_w are the largest for small t in Figure 8.14. For Figures 8.9 to 8.11 the differences between P_b and P_w are much smaller since an increasing t means an increase of k, so this does not merit further explanation. The comparison of Figures 8.11 and Figure 8.15 will also show that these differences are negligible.

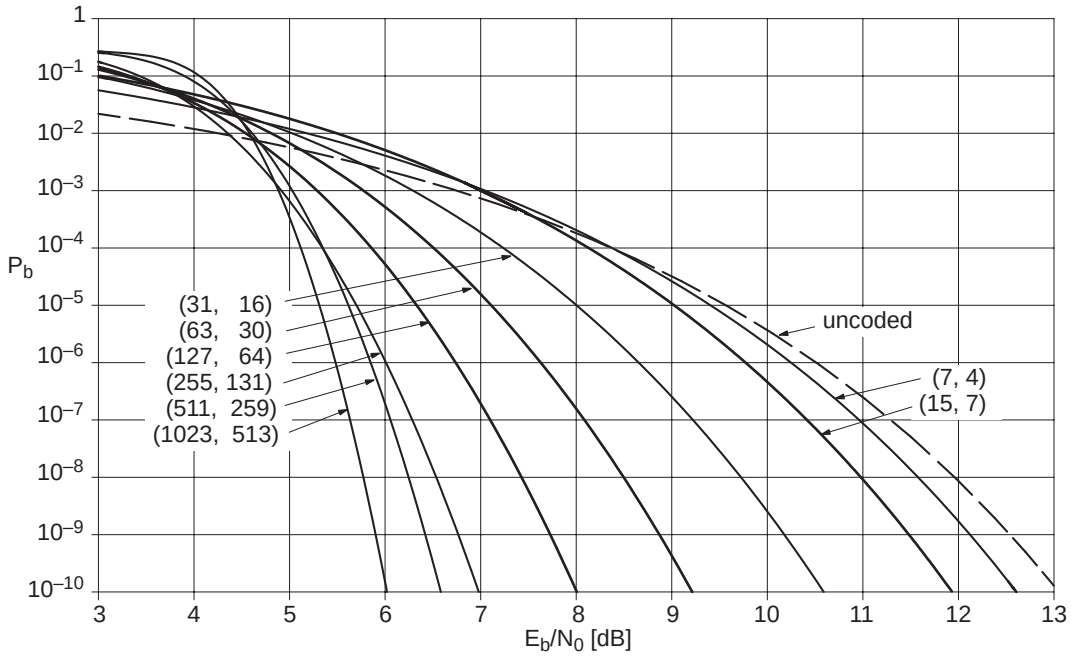

Figure 8.10. Bit-error probabilities of BCH codes with $R = 1/2$

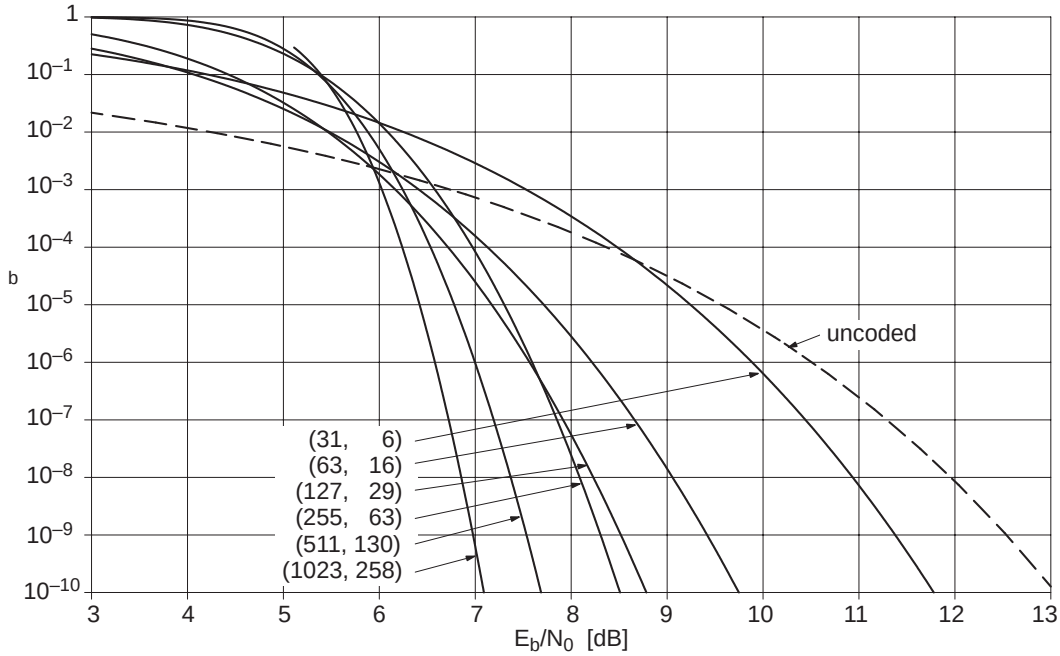

Figure 8.11. Bit-error probabilities of BCH codes with $R = 1/4$

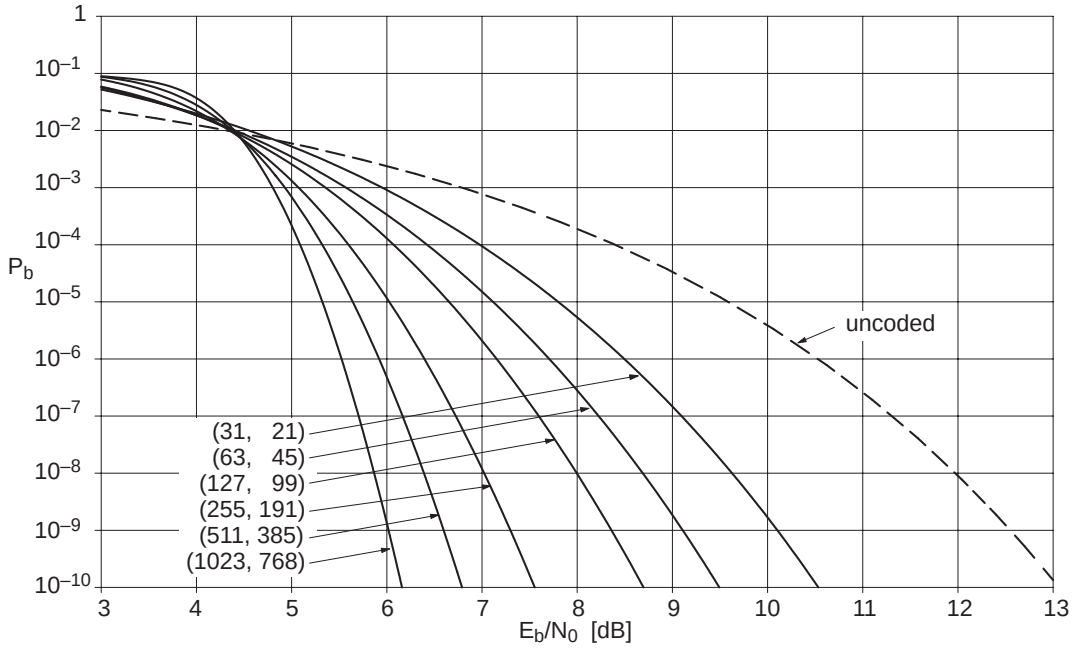

Figure 8.12. Bit-error probabilities of BCH codes with $R = 3/4$

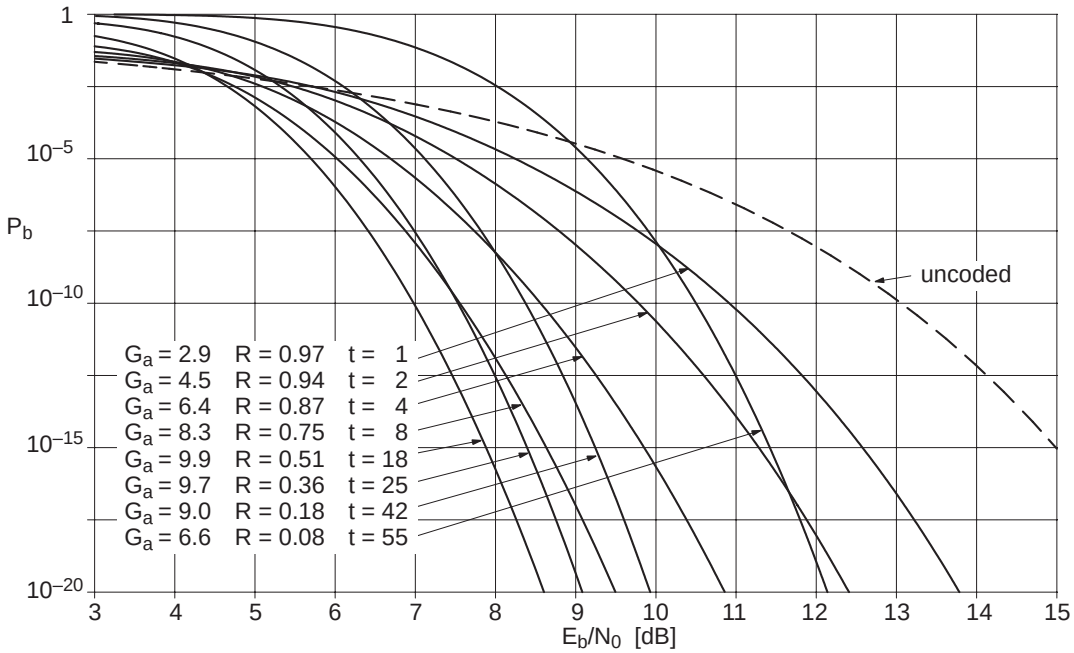

Figure 8.13. Bit-error probabilities of BCH codes with $n = 255$

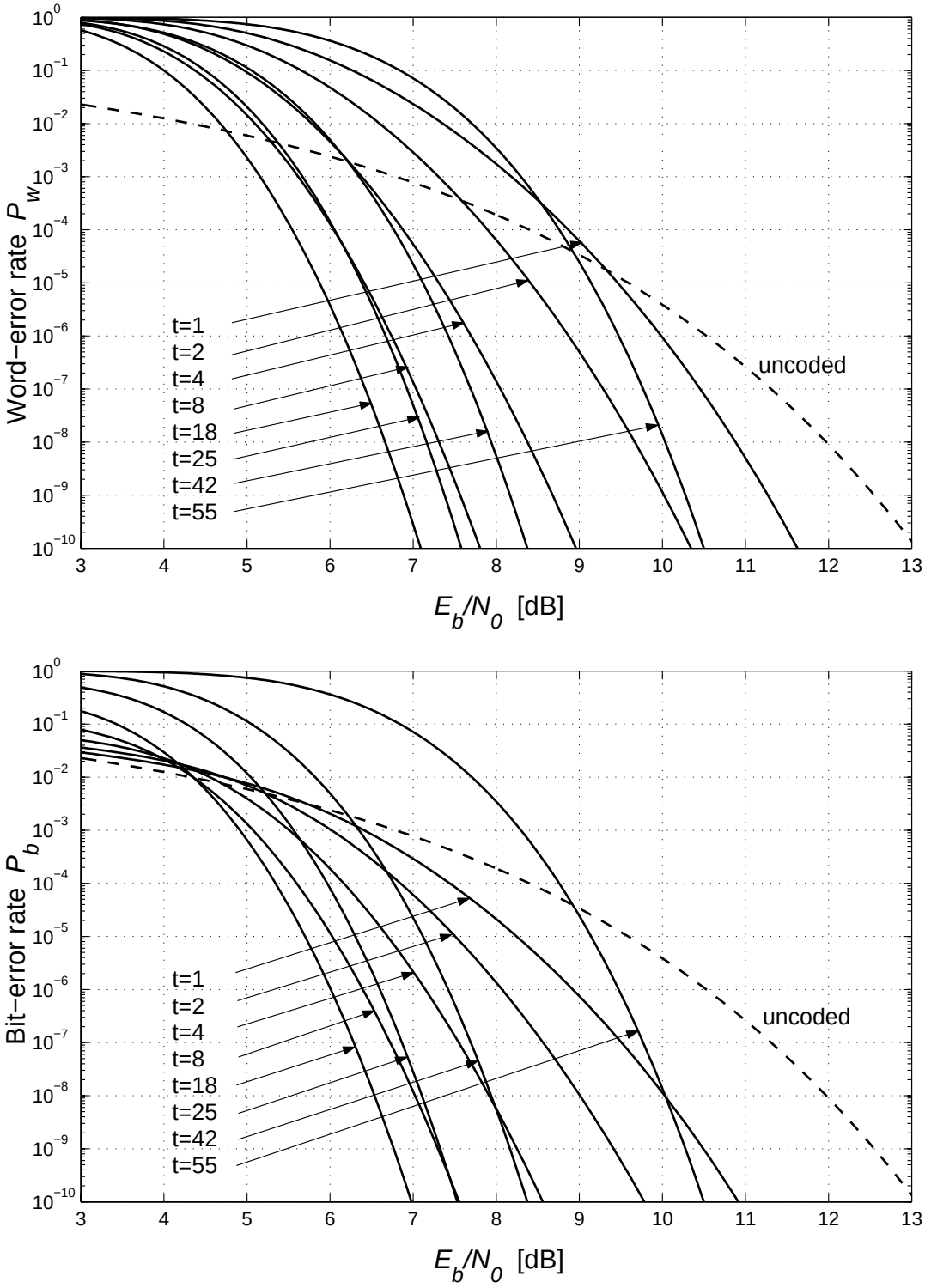

Figure 8.14. Word-error and bit-error probabilities of BCH codes with $n = 255$ (same codes as in Figure 8.13)

8.2.4 Further Considerations and Asymptotic Properties

Figure 8.15 shows a comparison between a binary BCH code and three RS codes, where $R \approx 3/4$ is the code rate for all cases. Again the binary AWGN channel with hard decisions is presupposed, so there are no burst errors, only random single errors. For the same nominal block length the RS code is better than the BCH code. However, when compared to the RS block length with binary interpretation, the BCH code is better, as expected on the grounds of the previous considerations and in particular Example 8.4.

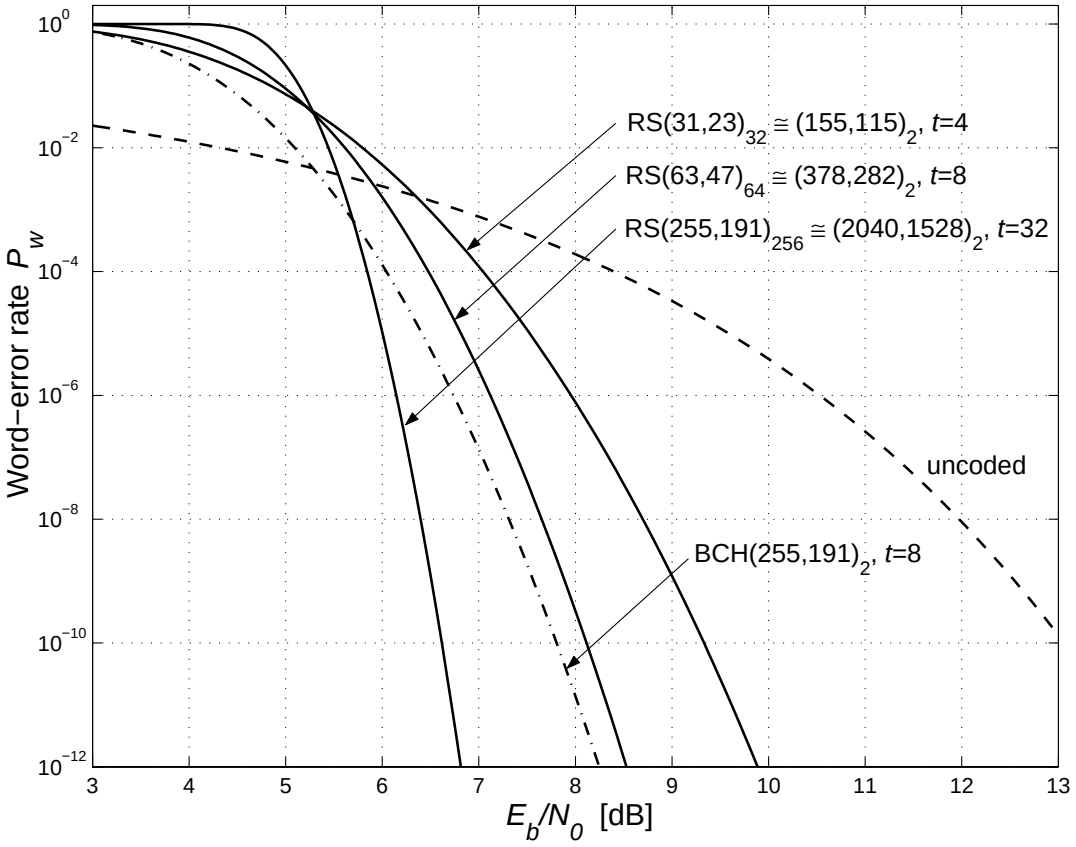

Figure 8.15. Comparison of a BCH code with some RS codes (all with $R \approx 3/4$)

Note that Figure 8.15 also shows the word-error rate and Figure 8.12 the bit-error rate; the small difference for the BCH code corresponds exactly to the different factors of (4.7.4) and (4.7.5) of Theorem 4.15.

Now, we will examine the influence block lengths have on the behaviour of BCH codes by taking a closer look at Tables 8.3 and 8.4. According to Table 8.4 the asymptotic coding gain $G_{\mathrm{a,hard}} = 10 \cdot \log_{10}(R(t+1))$ grows with increasing block length, as expected on the grounds of Figures 8.9 to 8.11. Yet, at the same time the distance rate d/n decreases, as shown in Table 8.5. The distance rate

Table 8.4. Asymptotic coding gains $G_{\text{a,hard}}$ of some BCH codes

n	$R \approx 1/2$	$R \approx 1/4$	$R \approx 3/4$
31	3.1	1.9	3.1
63	5.2	4.8	4.6
127	7.4	7.0	5.9
255	9.9	8.8	8.3
511	12.0	11.5	10.5
1023	14.6	14.3	13.1

Table 8.5. Distance rate d/n of some BCH codes

n	$R \approx 1/2$	$R \approx 1/4$	$R \approx 3/4$
31	0.23	0.48	0.16
63	0.21	0.37	0.11
127	0.17	0.34	0.07
255	0.15	0.24	0.07
511	0.12	0.22	0.06
1023	0.11	0.21	0.05
asympt.GV	0.11	0.21	0.04

of RS codes is much larger with $d_{\min} = 1 - R + 1/n \approx 1 - R$, and thus lies on the asymptotic Singleton bound, which is obvious since RS codes, being MDS codes, satisfy the Singleton bound with equality.

Figure 8.16 shows a comparison of the binary BCH codes with the asymptotic upper Elias bound and the asymptotic lower Gilbert-Varshamov (GV) bound of Figure 4.6. For the block lengths 31, 63 and 255 all BCH codes are listed, however, for 1023 only a small number of representative BCH codes is given. Short BCH codes lie way above the GV bound, and for $n = 31$ even above the upper Elias bound (which is, of course, no contradiction, since for a small n, codes can have a quite different behaviour than in the asymptotic case of $n \to \infty$). For a bigger n the distance rate comes close to the GV bound and for $n > 1023$ even goes below the GV bound. So even in this representation BCH codes turn out to be asymptotically bad.

Asymptotically bad codes with $d/n \to 0$ at a constant code rate and $n \to \infty$ may, of course, still have a coding gain going toward infinity, since

$$G_{\text{a,hard}} \approx 10 \cdot \log_{10}\left(\frac{R}{2} \cdot \frac{d}{n} \cdot n\right) \;\to\; \infty \quad \text{as} \quad n \to \infty$$

is possible, if d/n converges very slowly toward zero.

Although BCH codes are asymptotically bad their practical advantages still dominate: BCH codes exist for many parameter values, they are more powerful for short and medium block lengths than any other known code family, the costs of encoding and especially of decoding are relatively small. However, the disadvantages are:

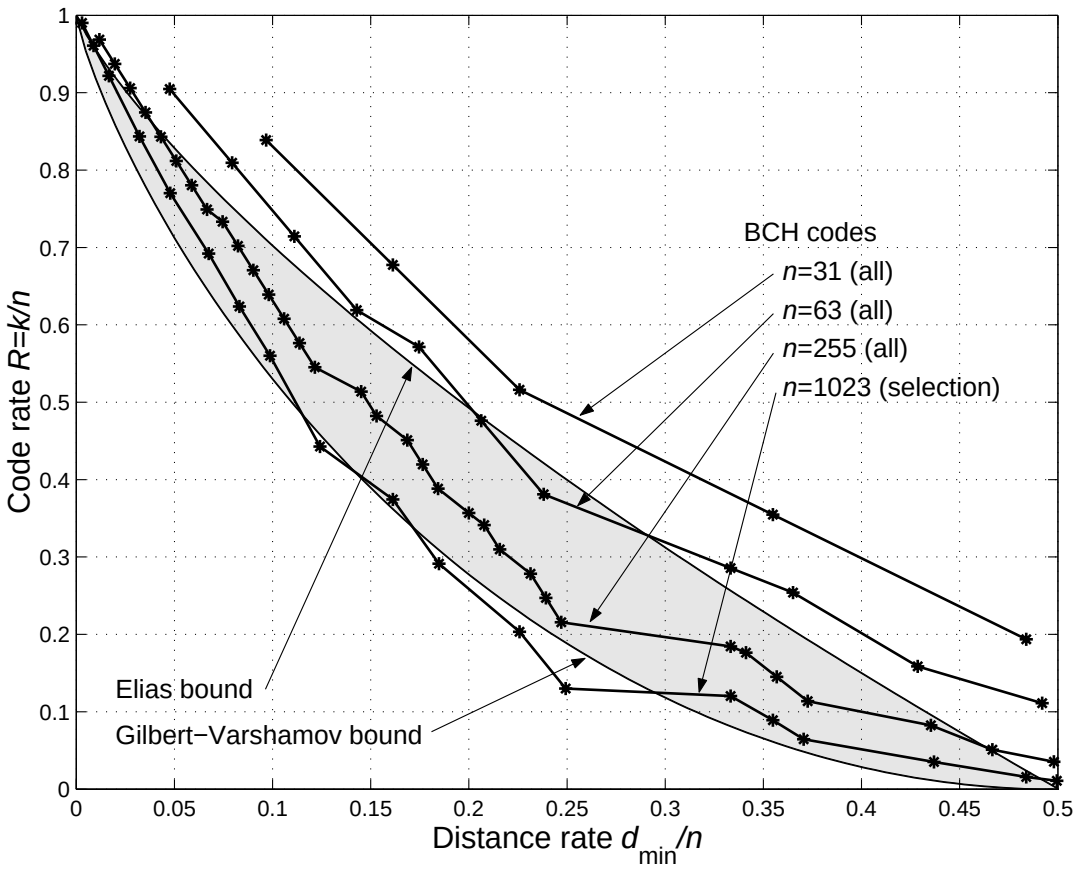

Figure 8.16. Comparison of some BCH codes with upper and lower asymptotic bounds

- BCH codes can only correct up to half the designed distance by the usual, less time-consuming decoding methods. It is of no use for decoding if the actual minimum distance is bigger. ML decoding is usually impossible, although it would mean enormous improvements [**?**].

- Furthermore the usual decoding methods only process and exploit hard decisions. So the expected gain of 2 to 3 dB for soft decisions can not be achieved. For example, according to Figure 8.10, the block length could be reduced by at least a factor of 8 with soft decisions.

The methods for increasing the coding gain of BCH codes found up to now require so much more effort for decoding, that they are uninteresting in practice, therefore we will not discuss them here.

8.3 Decoding Basics: Syndrome and Key Equation

8.3.1 The Syndrome

For RS and BCH codes, the ideal maximum-likelihood decoder can not be implemented with a reasonable amount of effort, instead only the bounded-minimum-distance (BMD) decoder as given in Definition 4.6 is feasible. Therefore, in the following, we will assume that the actual number of errors in the received word is no higher than the number of correctable errors. However, if there are more errors then no specific behaviour of the decoder is stipulated, since this case is assumed to be a failure of BMD decoder.

Although the derivation of the decoding algorithms is given for the general case it is primarily oriented on RS codes. Since, in contrast to RS decoding, there are enormous simplifications for BCH codes and in particular for the binary BCH codes; we will discuss this case separately in Section 8.7.

The received word $\boldsymbol{y}$ with hard decisions is the superposition of the transmitted codeword $\boldsymbol{a}$ with the error word $\boldsymbol{e}$ and because of linearity of the spectral transformation

$$\boldsymbol{y} = \boldsymbol{a} + \boldsymbol{e} \quad \circ\!\!-\!\!\bullet \quad \boldsymbol{Y} = \boldsymbol{A} + \boldsymbol{E}. \tag{8.3.1}$$

The symbols of the codeword and the error word are $q = p^m$-ary for RS codes and p-ary for BCH codes in the time domain and q-ary in the frequency domain. The term "error" in the time domain always refers to the q- or p-ary values. Whether the code symbols are transmitted directly as multilevel or as bit groups is irrelevant: if just one bit of a bit group is wrong, the whole bit group is considered to be false.

General presumptions for RS decoding: let $d = 2t + 1$ be the designed distance and $n = p^m - 1$ be the primitive block length. For the number τ of actual errors, we have

$$\tau \;\le\; t \left\{ \begin{array}{ll} = (n-k)/2 & \text{RS codes} \\ \le (n-k)/2 & \text{BCH codes} \end{array} \right\}. \tag{8.3.2}$$

Furthermore, let $l = 0$ in Definitions 8.1 and 8.2 for RS and BCH codes, respectively. So the $2t$ parity frequencies are at the lowest positions $0, 1, \ldots, 2t-1$. Then the codeword in the frequency domain is described by

$$a(x) \quad \circ\!\!-\!\!\bullet \quad A(x) \leftrightarrow \boldsymbol{A} = (\underbrace{0, \ldots, 0}_{2t \text{ zeros}}, \underbrace{A_{2t}, \ldots, A_{n-1}}_{n-2t \text{ positions}}). \tag{8.3.3}$$

For BCH codes some of the $n - 2t$ higher frequencies can also be constantly zero, however, this is of no use for decoding, since it would need a sequence of consecutive roots. The received word is

$$\boldsymbol{Y} = (\underbrace{E_0, \ldots, E_{2t-1}}_{= \boldsymbol{S}}, A_{2t} + E_{2t}, \ldots, A_{n-1} + E_{n-1}). \tag{8.3.4}$$

Definition 8.3. *The* syndrome $\boldsymbol{S} \leftrightarrow S(x)$ *is defined by the parity frequencies of the received word, in other words, by the* $2t$ *components of the Fourier transform of the received word at those positions where the codewords have zeros in the frequency domain:*

$$S(x) = \sum_{i=0}^{2t-1} S_i x^i \quad , \quad S_i = E_i = Y_i = y(z^i) = \sum_{\mu=0}^{n-1} y_\mu z^{i\mu}. \tag{8.3.5}$$

As in the Definitions 5.7 and 6.4, the syndrome is defined as a vector of $2t$ components which are all-zero if and only if the received word or the error pattern is a codeword. However, the syndrome defined here is not equal to the Fourier transform of the syndrome of Definition 6.4. The calculation of the syndrome actually corresponds to a partial DFT. Although this is the most time-consuming part of the decoder, many operations can be performed in parallel and sometimes also FFT (Fast Fourier Transform) methods can be used [17, 18]. For BCH codes we do not actually have to Fourier transform each of the $2t$ frequencies, since, according to Theorem 7.9, $S_{ip \bmod n} = S_i^p$ must be valid (we will discuss this in detail in Section 8.7).

If only error detection is required then the decoding is finished after calculating the syndrome and checking whether it is zero.

8.3.2 The Error-Locator Polynomial and the Key Equation

Knowing the syndrome means also knowing the lower $2t$ frequencies of the received word or the error pattern. We now need to find the remaining $n - 2t$ frequencies such that the Hamming weight $w_H(\boldsymbol{e}) = w_H(e(x))$ takes on its minimum in the time domain. If we were to swap the time for the frequency domain, this would be a classic task in communications engineering: complete an impulse such that its spectrum is as narrow as possible. The requirement of a minimum Hamming weight of $e(x)$ is now to be transformed into the requirement of the minimum order of an suitable polynomial, so that an algebraic evaluation is possible.

Definition 8.4. *Each error pattern* $e(x) \leftrightarrow (e_0, \ldots, e_{n-1})$ *is mapped to the* set of unknown error locations

$$I = \{i \mid 0 \le i \le n-1 \ \wedge \ e_i \neq 0\} \quad , \quad |I| = \tau \le t \tag{8.3.6}$$

which defines the error-locator polynomial

$$C(x) = \prod_{i \in I} (1 - xz^i) = 1 + C_1 x + \cdots + C_\tau x^\tau \tag{8.3.7}$$

of order τ*. For the error-free case with* $I = \emptyset$*, we assume* $C(x) = 1$*.*

Of course, I and $C(x)$ are unknown and depend only on the error pattern but not on the transmitted codeword. Obviously, $C(x)$ contains all the information about the error locations, since I can be reconstructed from $C(x)$ with the help of the so-called *Chien search*:

$$I = \{i \mid C(z^{-i}) = 0\}. \tag{8.3.8}$$

So the Chien search means checking through all z^{-i} to see if there is a root of $C(x)$. This leaves us with the main problem of calculating $C(x)$ from $S(x)$ for which we will now examine the dependence between both polynomials. Let $c(x) \circ\!\!-\!\!\bullet C(x)$. For $i \in I$, $c_i = -C(z^{-i}) = 0$, and for $i \notin I$, $e_i = 0$. Thus $c_i e_i = 0$ for $0 \leq i \leq n-1$ and, according to Theorem 7.8,

$$R_{x^n-1}[C(x)E(x)] = 0. \tag{8.3.9}$$

This can be equivalently expressed by a cyclic convolution

$$\sum_{\mu=0}^{\tau} C_\mu E_{(i-\mu) \bmod n} = 0 \quad \text{für} \quad i = 0, \ldots, n-1. \tag{8.3.10}$$

Now, we will reduce the condition (8.3.10) to the interval $i = \tau, \ldots, 2\tau - 1$, then all indices of E are in the range $0, \ldots, 2t-1$ so that $E_{(i-\mu) \bmod n}$ can be replaced by $S_{i-\mu}$ (note that $C_0 = 1$):

$$S_i + \sum_{\mu=1}^{\tau} C_\mu S_{i-\mu} = 0 \quad \text{für} \quad i = \tau, \ldots, 2\tau - 1. \tag{8.3.11}$$

This result is called the *key equation* or also *Newton identity*. The matrix form of (8.3.11) is

$$\begin{pmatrix} -S_\tau \\ -S_{\tau+1} \\ \vdots \\ -S_{2\tau-2} \\ -S_{2\tau-1} \end{pmatrix} = \underbrace{\begin{pmatrix} S_0 & S_1 & \cdots & S_{\tau-2} & S_{\tau-1} \\ S_1 & S_2 & \cdots & S_{\tau-1} & S_\tau \\ \vdots & \vdots & & \vdots & \vdots \\ S_{\tau-2} & S_{\tau-1} & \cdots & S_{2\tau-4} & S_{2\tau-3} \\ S_{\tau-1} & S_\tau & \cdots & S_{2\tau-3} & S_{2\tau-2} \end{pmatrix}}_{= \boldsymbol{S}_{\tau,\tau}} \cdot \begin{pmatrix} C_\tau \\ C_{\tau-1} \\ \vdots \\ C_2 \\ C_1 \end{pmatrix}. \tag{8.3.12}$$

This is a linear equation system with τ equations to determine the τ unknowns $C_1, \ldots, C_\tau$ and therefore to determine the error locations. Since all syndrome components are known, (8.3.12) is solvable, however, we require efficient calculation methods which only call for reasonable effort.

The equation system (8.3.12) can also be formed for the maximum number t of errors. In the following, we will show that the rank of the corresponding

matrix $\boldsymbol{S}_{t,t}$ is equal to the actual number τ of errors. For this proof we enumerate the set of error locations with indices $I = \{i_1, \ldots, i_\tau\}$ and with the abbreviation $Z_\mu = z^{i_\mu}$ we have the following for the syndrome:

$$S_r = \sum_{\mu=0}^{n-1} e_\mu z^{r\mu} = \sum_{\mu=1}^{t} e_{i_\mu} Z_\mu^r. \tag{8.3.13}$$

In the case of $\tau < t$, $t - \tau$ error magnitudes e_{i_μ} are set to zero. For the matrix $\boldsymbol{S}_{t,t}$ we have the following factorization, which can be easily verified:

$$\boldsymbol{S}_{t,t} = \underbrace{\begin{pmatrix} 1 & 1 & \cdots & 1 \\ Z_1 & Z_2 & \cdots & Z_t \\ \vdots & \vdots & & \vdots \\ Z_1^{t-1} & Z_2^{t-1} & \cdots & Z_t^{t-1} \end{pmatrix}}_{=\boldsymbol{Z}_{t,t}} \cdot \underbrace{\begin{pmatrix} e_{i_1} & & & \\ & e_{i_2} & & \\ & & \ddots & \\ & & & e_{i_t} \end{pmatrix}}_{=\boldsymbol{\Delta}_{t,t}} \cdot \underbrace{\begin{pmatrix} 1 & Z_1 & \cdots & Z_1^{t-1} \\ 1 & Z_2 & \cdots & Z_2^{t-1} \\ \vdots & \vdots & & \vdots \\ 1 & Z_t & \cdots & Z_t^{t-1} \end{pmatrix}}_{=\boldsymbol{Z}_{t,t}^T} . \tag{8.3.14}$$

The determinant of the matrix $\boldsymbol{Z}_{t,t}$ is the well known *Vandermonde determinant* [1, 17, 105]

$$\det(\boldsymbol{Z}_{t,t}) = \prod_{t \geq \mu > \nu \geq 1} (Z_\mu - Z_\nu). \tag{8.3.15}$$

For $0 \leq i_\nu < i_\mu \leq n-1$, $Z_\nu = z^{i_\nu}$ and $Z_\mu = z^{i_\mu}$ are, of course, unequal and therefore $\det(\boldsymbol{Z}_{t,t}) \neq 0$. Hence, the matrix $\boldsymbol{Z}_{t,t}$ is not singular and

$$\operatorname{rank}(\boldsymbol{S}_{t,t}) = \operatorname{rank}(\boldsymbol{\Delta}_{t,t}) = \tau \tag{8.3.16}$$

leads to the result as stated above. ■

Of course, we do not know the number τ of errors in the received word. Either τ can be determined by evaluating the rank of the matrix $\boldsymbol{S}_{t,t}$ or by using the following method. Again we use (8.3.12) with t instead of τ. If $\tau = t$, then $\boldsymbol{S}_{t,t}$ is non-singular and (8.3.12) is solvable with a unique solution. If, however, $\tau < t$ then $\boldsymbol{S}_{t,t}$ is singular and (8.3.12) is not solvable. Then we delete the bottom row and the right column of the matrix $\boldsymbol{S}_{t,t}$ and check the resulting matrix $\boldsymbol{S}_{t-1,t-1}$ for singularity. We repeat this method until we have found a maximum τ such that $\boldsymbol{S}_{\tau,\tau}$ is non-singular.

The solution to the key equation now gives us $I = \{i_1, \ldots, i_\tau\}$ as well as $Z_\mu = z^{i_\mu}$. The relation (8.3.13) leads to the equation system

$$\begin{pmatrix} S_0 \\ S_1 \\ \\ \vdots \\ \\ S_{\tau-1} \end{pmatrix} = \underbrace{\begin{pmatrix} 1 & 1 & \cdots & 1 \\ Z_1 & Z_2 & \cdots & Z_\tau \\ & & & \\ \vdots & \vdots & & \vdots \\ & & & \\ Z_1^{\tau-1} & Z_2^{\tau-1} & \cdots & Z_\tau^{\tau-1} \end{pmatrix}}_{= \boldsymbol{Z}_{\tau,\tau}} \cdot \begin{pmatrix} e_{i_1} \\ e_{i_2} \\ \\ \vdots \\ \\ e_{i_\tau} \end{pmatrix}, \tag{8.3.17}$$

from which we can calculate the error magnitudes directly, since $\boldsymbol{Z}_{\tau,\tau}$ is non-singular. This method, called *Peterson-Gorenstein-Zierler decoder*, is too time-consuming for practical decoding with big block lengths and big designed distances. However, the special band structure of the matrix $\boldsymbol{S}_{\tau,\tau}$ enables much better algorithms for calculating the solution of the key equation (see Section 8.5) as well as for calculating the error magnitudes (see Section 8.4). There are further simplifications for binary BCH codes, discussed separately in Section 8.7.

If, in contrast to the presumption, there are more than t errors, there are two possibilities:

(1) Usually the received word will be outside a decoding sphere, if the code is not perfect. In this case, an ideal BMD decoder must detect an uncorrectable error pattern, this is usually called *decoder failure* and contributes to P_{ed} as defined in (8.1.10). Such a situation is recognized by the Chien search that then finds less than deg $C(x)$ roots of $C(x)$ or that $C(x)$ in $\mathbb{F}_{p^m}$ can not be factorized into various linear factors or that the error magnitudes are zero at the calculated error locations.

However, it is also possible that for a received word outside a decoding sphere an apparently successful decoding is made, this is usually called *decoder malfunction* [164, 220]. Yet, the word determined by the decoder can not be a codeword, since the decoder only corrects a maximum of t errors. This case can be easily recognized by checking the parity frequencies of the calculated word.

(2) However, if the received word is in a decoding sphere of another codeword, then an unrecognized wrong correction takes place, this is usually called *decoder error* and contributes to P_{icd} as defined in (8.1.12).

Detailed examples of the various scenarios of decoding can be found in Problem 8.10. For the calculation or approximation of the error probability as in Theorem 4.15, each received word with more than t errors is considered to be decoded incorrectly. So we do not have to distinguish between received words in wrong decoding spheres and received words outside of any decoding sphere.

Example 8.6. We consider the key equation for the case of $\tau = t = 1$: let $I = \{r\}$ and $e(x) = e_r z^r$ and therefore $S_i = e(z^i) = e_r z^{ir}$. The key equation has the simple form $-S_1 = S_0 \cdot C_1$ and leads to the solution

$$C(x) = C_0 + C_1 x = 1 - \frac{S_1}{S_0} x = 1 - \frac{e_r z^r}{e_r} x = 1 - z^r x = \prod_{i \in I} (1 - x z^i).$$

The Chien search delivers $I = \{r\}$. For $\tau = 1 < 2 = t$, the matrix

$$\boldsymbol{S}_{2,2} = \begin{pmatrix} S_0 & S_1 \\ S_1 & S_2 \end{pmatrix} = \begin{pmatrix} e_r & e_r z^r \\ e_r z^r & e_r z^{2r} \end{pmatrix}$$

is obviously singular. ■

Example 8.7. We consider the $(7, 3, 5)_8$ RS code with the $\mathbb{F}_8$ arithmetic as in Example 6.4. For the codewords, $\boldsymbol{a} \circ\!\!-\!\!\bullet\, \boldsymbol{A} = (0, 0, 0, 0, A_4, A_5, A_6)$ is valid. For example, choose

$$\boldsymbol{e} = (0, z^5, 0, 0, 0, z^6, 0) \;\leftrightarrow\; e(x) = z^5 x + z^6 x^5,$$

then $S_i = e(z^i) = z^{5+i} + z^{6+5i}$, so

$$\begin{aligned} S_0 &= z^5 + z^6 = z \\ S_1 &= z^6 + z^4 = z^3 \\ S_2 &= 1 + z^2 = z^6 \\ S_3 &= z + 1 = z^3. \end{aligned}$$

The key equation for $t = 2$ is

$$\begin{pmatrix} -S_2 \\ -S_3 \end{pmatrix} = \begin{pmatrix} S_0 & S_1 \\ S_1 & S_2 \end{pmatrix} \begin{pmatrix} C_2 \\ C_1 \end{pmatrix}$$

and thus has the solution

$$\begin{aligned} \begin{pmatrix} C_2 \\ C_1 \end{pmatrix} &= \frac{1}{S_0 S_2 - S_1^2} \begin{pmatrix} S_2 & -S_1 \\ -S_1 & S_0 \end{pmatrix} \begin{pmatrix} -S_2 \\ -S_3 \end{pmatrix} = \frac{1}{1 + z^6} \begin{pmatrix} z^6 & z^3 \\ z^3 & z \end{pmatrix} \begin{pmatrix} z^6 \\ z^3 \end{pmatrix} \\ &= \frac{1}{z^2} \begin{pmatrix} z^5 + z^6 \\ z^2 + z^4 \end{pmatrix} = \begin{pmatrix} z^6 \\ z^6 \end{pmatrix}. \end{aligned}$$

The error-locator polynomial $C(x) = 1 + z^6 x + z^6 x^2 = (1 + z^1 x)(1 + z^5 x)$ again leads to $I = \{1, 5\}$. In the following, we will make frequent use of this example.

As a further example with $\tau = 3 > 2 = t$ we choose

$$\boldsymbol{e} = (z, z^5, 0, 0, 0, z^6, 0) \;\leftrightarrow\; e(x) = z + z^5 x + z^6 x^5.$$

For the syndrome we have $(S_0, S_1, S_2, S_3) = (0, 1, z^5, 1)$ which gives us the error-locator polynomial $C(x) = 1 + z^5 x + z x^2$. The Chien search does not find a root, thus $C(x)$ is irreducible in $\mathbb{F}_8$. Hence, an error pattern with $\tau > t$ is found which is uncorrectable for the BMD decoder. ■

8.4 Decoding Architectures

8.4.1 Frequency-Domain Error Correction

The error-locator polynomial $C(x)$ and the set of unknown error locations I are known after solving the key equation. Now, we have to determine the error magnitudes $e(x) \circ\!\!-\!\!\bullet E(x)$. According to (8.3.10),

$$E_i = -\sum_{\mu=1}^{t} C_\mu E_{i-\mu} \quad \text{for} \quad i = 2t, \ldots, n-1 \tag{8.4.1}$$
$$= \text{function of } (E_{i-1}, \ldots, E_{i-t}).$$

This facilitates the *principle of recursive extension*: the start values are $E_0 = S_0, \ldots, E_{2t-1} = S_{2t-1}$. One after the other, $E_{2t}, \ldots, E_{n-1}$ are computed. Technically, the recursive extension can be realized by a linear feedback shift register (LFSR, also called *autoregressive filter*) as shown in Figure 8.17. The length of the filter can be variable with τ or can be fixed to t. The start configuration is defined by the t highest frequencies $S_{2t-1}, \ldots, S_t$ of the syndrome. At the filter output, we have the values of $E_{2t}, \ldots, E_{n-1}$ sequentially.

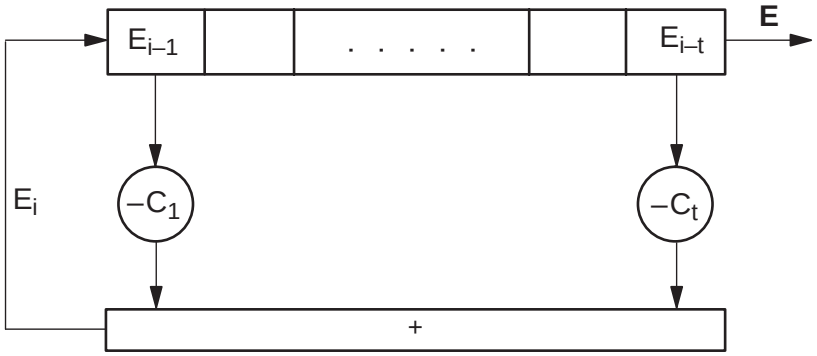

Figure 8.17. Computation of $E(x)$ by recursive extension

Figure 8.18 shows the whole method of decoding in the frequency domain. Although the correction takes place in the time domain, the calculation of the error magnitudes is performed in the frequency domain. Therefore we must compute the inverse Fourier transform of $E(x)$, so that $e(x)$ is available in the time domain. However, we only have to determine the components which were detected as being erroneous. To reduce the effort we will introduce a method in the next subsection which can directly calculate the desired error magnitudes in the time domain.

Example 8.8. Continuation of Example 8.7 with the $(7, 3, 5)_8$ RS code. The

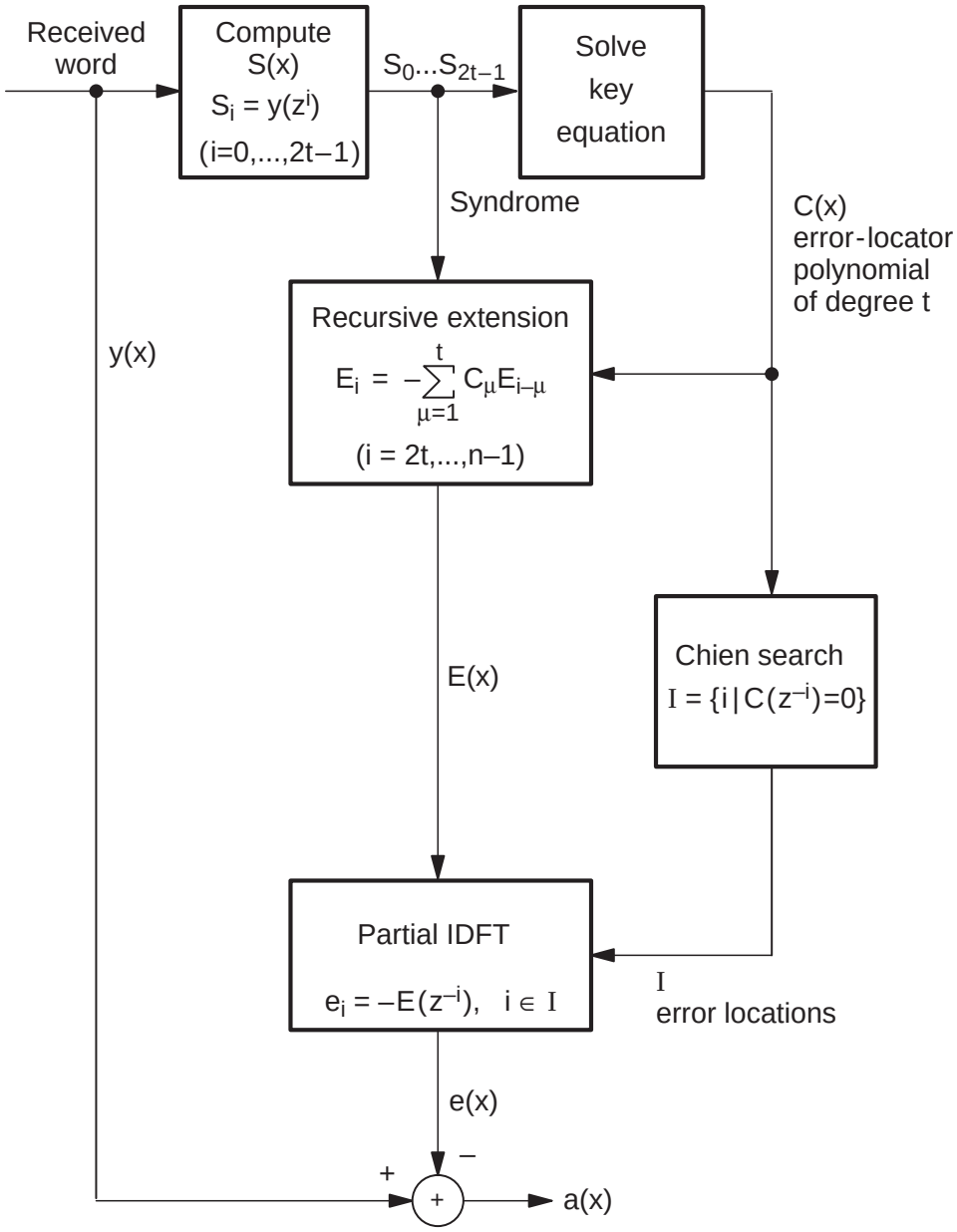

Figure 8.18. Error-correction decoder in frequency domain

parameters

$$(E_0, E_1, E_2, E_3) = (S_0, S_1, S_2, S_3) = (z, z^3, z^6, z^3),$$
$$(C_0, C_1, C_2) = (1, z^6, z^6),$$
$$I = \{1, 5\} \quad , \quad \tau = t = 2$$

are already known. Recursive extension leads to

$$\begin{aligned} E_4 &= C_1E_3 + C_2E_2 = z^6z^3 + z^6z^6 = z^3 \\ E_5 &= C_1E_4 + C_2E_3 = z^6z^3 + z^6z^3 = 0 \\ E_6 &= C_1E_5 + C_2E_4 = z^60 + z^6z^3 = z^2. \end{aligned}$$

Finally,

$$\begin{aligned} E(x) &= S_0 + S_1x + S_2x^2 + S_3x^3 + E_4x^4 + E_5x^5 + E_6x^6 \\ &= z + z^3x + z^6x^2 + z^3x^3 + z^3x^4 + z^2x^6. \end{aligned}$$

Since we have already determined that $I = \{1, 5\}$, all we need to do is calculate e_1 and e_5:

$$\begin{aligned} e_1 &= -E(z^{-1}) = E(z^6) = z + z^2 + z^4 + 1 + z^6 + z^3 = z^5 \\ e_5 &= -E(z^{-5}) = E(z^2) = z + z^5 + z^3 + z^2 + z^4 + 1 = z^6. \end{aligned}$$

Thus, the assumed error pattern $e(x) = z^5x + z^6x^5$ is the result. ■

8.4.2 Time-Domain Error Correction

The objective now is to determine the error magnitudes at the desired positions directly in the time domain without inverse Fourier transforms. As in the previous subsection, the error-locator polynomial $C(x)$ and the set of unknown error locations I are considered to be known after solving the key equation. According to (8.3.9), a so-called *error-evaluator polynomial* $T(x)$ exists with

$$C(x)E(x) = T(x)(x^n - 1). \tag{8.4.2}$$

$T(x)$ does not directly show the error magnitudes, but the error magnitudes can be calculated from $T(x)$ as follows. For the degree of $T(x)$,

$$\deg T(x) \;=\; \underbrace{\deg\, C(x)}_{=\tau} + \underbrace{\deg\, E(x)}_{\leq n-1} - \underbrace{\deg\, (x^n - 1)}_{=n} \;\leq\; \tau - 1.$$

So $T(x)$ kann be written as

$$T(x) = T_0 + T_1x + \cdots + T_{\tau-1}x^{\tau-1}. \tag{8.4.3}$$

The equation (8.4.2) is now written as

$$\underbrace{\sum_{i=0}^{\tau} C_ix^i \cdot \sum_{i=0}^{2t-1} E_ix^i}_{\text{from } x^0 \text{ onwards}} \;+\; \underbrace{C(x)\sum_{i=2t}^{n-1} E_ix^i}_{\text{from } x^{2t} \text{ onwards}} \;=\; \underbrace{-T(x)}_{\text{until } x^{\tau-1}} \;+\; \underbrace{x^nT(x)}_{\text{until } x^n},$$

where the range of the corresponding exponents is given below the braces. The comparison of the coefficients together with $\tau-1 \leq 2t$ implies that the coefficient $-T_j$ must correspond to the j-th coefficient in the left term. The left term must be the result of a convolutional operation:

$$-T_j = \sum_{\mu=0}^{j} C_\mu E_{j-\mu} \quad \text{for} \quad j = 0, \ldots, \tau - 1.$$

Since $0 \leq j - \mu \leq \tau - 1$, $E_{j-\mu}$ can be replaced by $S_{j-\mu}$:

$$T_j = -\sum_{\mu=0}^{j} C_\mu S_{j-\mu} \quad \text{for} \quad j = 0, \ldots, \tau - 1. \tag{8.4.4}$$

Since only $S_0, \ldots, S_{\tau-1}$ are used, $T(x)$ can be calculated by using (8.4.4). Now, with the error-locator polynomial $C(x)$ of degree τ and the error-evaluator polynomial $T(x)$ of degree $\tau-1$ we can determine the error magnitudes directly. The equation (8.4.2) gives us

$$e_i = -E(z^{-i}) = -\left[\frac{T(x)(x^n-1)}{C(x)}\right]_{x=z^{-i}} \quad \text{for} \quad i = 0, \ldots, n-1. \tag{8.4.5}$$

At the error locations $i \in I$, we have $C(z^{-i}) = 0$ (Chien search) and at the same positions $x^n - 1 = 0$ for $x = z^{-i}$. So the quotient (8.4.5) for $i \in I$ is of type "0/0". L'Hôspital's rule (A.1.3) and formal differentiation are also valid in Galois fields. For example, for the first derivative,

$$(x^n)' = nx^{n-1} = -x^{n-1},$$

since $n = p^m - 1$ with modulo p. Correspondingly,

$$C'(x) = C_1 + 2C_2x + \cdots + tC_tx^{t-1}, \tag{8.4.6}$$

where the coefficients $1, 2, \ldots, \tau$, that were derived from the exponents, have to be considered modulo p. Then l'Hôspital's rule implies that

$$e_i \;=\; -\left[\frac{T'(x)(x^n-1)+T(x)(-x^{n-1})}{C'(x)}\right]_{x=z^{-i}} \;=\; \left[\frac{T(x)x^{n-1}}{C'(x)}\right]_{x=z^{-i}}.$$

This leads to the so-called *Forney algorithm*

$$e_i = \frac{T(z^{-i})}{C'(z^{-i})} \cdot z^i \quad \text{for} \quad i \in I. \tag{8.4.7}$$

All in all, we get the decoding method shown in Figure 8.19 which only differs from Figure 8.18 in the calculation of $T(x)$ and the Forney algorithm.

For the solution of the key equation and therefore for the calculation of $C(x)$, all $2t$ components of the syndrome are required (at least for the maximum number of errors $\tau = t$). However, for the calculation of $T(x)$ only the τ lower components of the syndrome are required (accordingly only the τ upper components are required for the recursive extension in Figure 8.18).

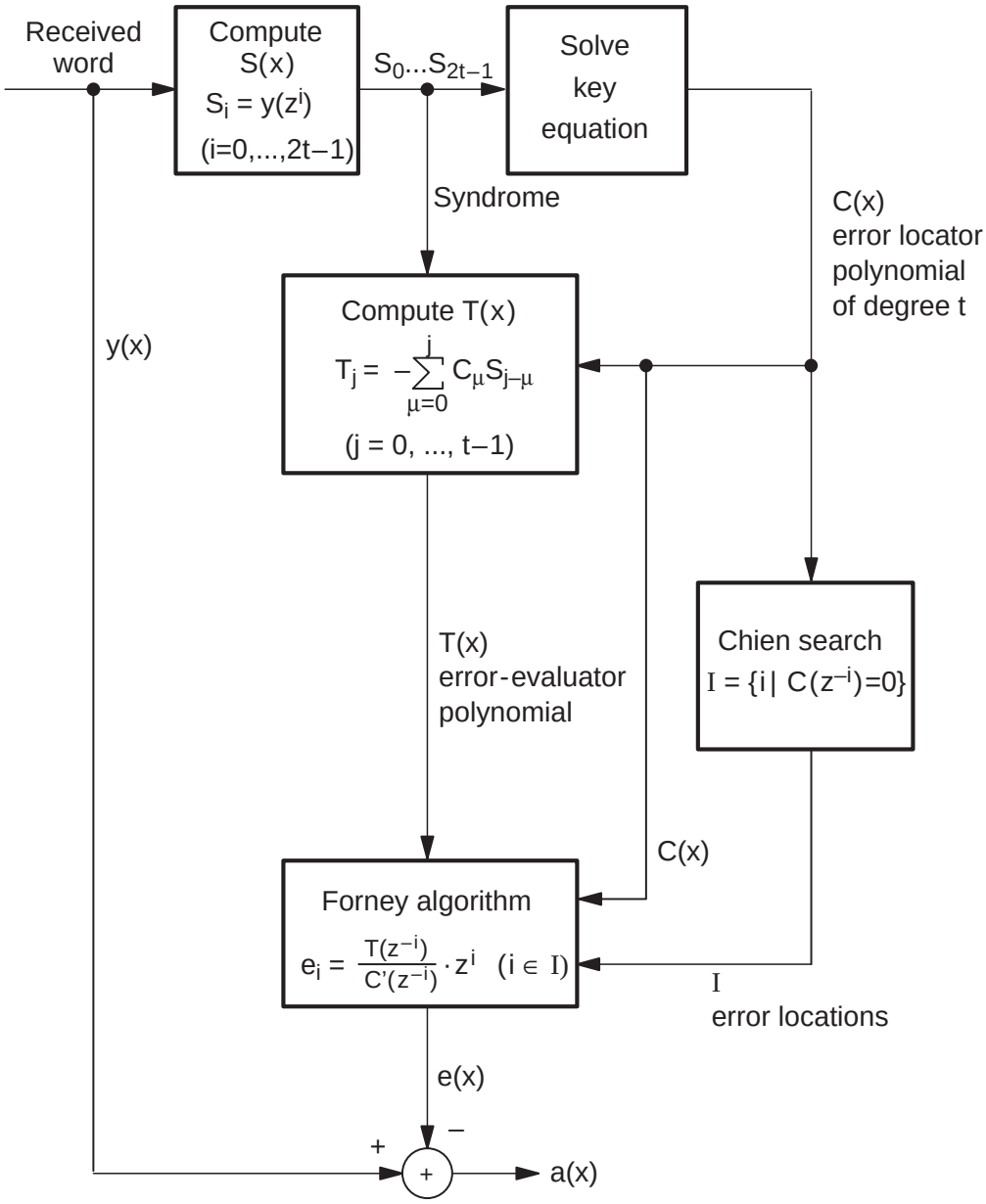

Figure 8.19. Error-correction decoder in time domain

Example 8.9. Continuation of Example 8.7 and 8.8 with the $(7,3,5)_8$ RS code. The following relations

$$\begin{aligned}
&(E_0, E_1, E_2, E_3) = (S_0, S_1, S_2, S_3) = (z, z^3, z^6, z^3),\\
&(C_0, C_1, C_2) = (1, z^6, z^6) \quad \leftrightarrow \quad C(x) = 1 + z^6 x + z^6 x^2,\\
&I = \{1, 5\} \quad , \quad \tau = t = 2
\end{aligned}$$

are known. The calculation of $T(x)$ gives us

$$\begin{array}{rcccccl}
T_0 & = & -C_0 S_1 & = & 1 \cdot z & = & z\\
T_1 & = & -C_0 S_2 - C_1 S_1 & = & 1 \cdot z^3 + z^6 \cdot z & = & z
\end{array}$$

and thus $T(x) = z + zx$. Differentiation leads to $C'(x) = z^6 + 2z^6x = z^6 \bmod 2$. Furthermore, the Forney algorithm leads to

$$e_i = \frac{T(z^{-i})}{C'(z^{-i})} z^i = \frac{z + z^{1-i}}{z^6} z^i = z^{2+i} + z^2 = \left\{ \begin{array}{ll} z^5 & i = 1 \\ z^6 & i = 5 \end{array} \right\}.$$

So the result is, once again, the assumed error pattern $e(x) = z^5x + z^6x^5$. For $i \notin I$, the Forney algorithm does not give us the correct result $e_i = 0$ but a wrong, non-zero value, i.e., (8.4.7) is only valid for the error locations $i \in I$. ■

8.5 Solving the Key Equation

8.5.1 The Berlekamp-Massey Algorithm

It was years after the discovery of RS and BCH codes that powerful algorithms were developed for the solution of the key equation (8.3.11) or (8.3.12). Some of these algorithms are based on the Euclidean algorithm (found in 1975) and will be introduced in the next subsection. A different method is the Berlekamp-Massey algorithm (BMA, found in 1968), which we examine here a without proof. Derivations of the BMA can be found in, e.g., [10, 17].

The equation (8.3.11) can be represented by a linear feedback shift register (LFSR) as shown in Figure 8.20. At the beginning ($i = t$) the register is initialized with $S_{t-1}, \ldots, S_0$. One after the other the values $S_t, \ldots, S_{2t-1}$ are created at register input. However, to solve the key equation we do not have to create the already known sequence of syndromes, but we are looking for the filter coefficients for the sequence of syndromes, where the filter is to be of minimum length. Thus solving the key equation is a problem of *synthesizing* an autoregressive filter.

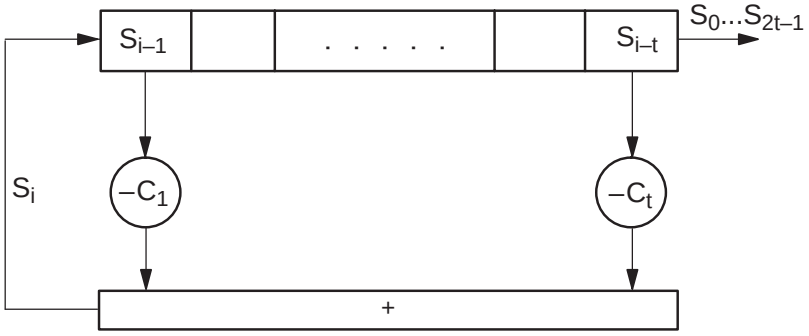

Figure 8.20. Generierung des Syndroms durch das Fehlerstellenpolynom

Figure 8.21 shows the Berlekamp-Massey algorithm. The input is the syndrome $(S_0, \ldots, S_{2t-1})$ and the output is the error-locator polynomial $C(x) = C^{(j)}(x)$ with the stop condition $j = 2t$.

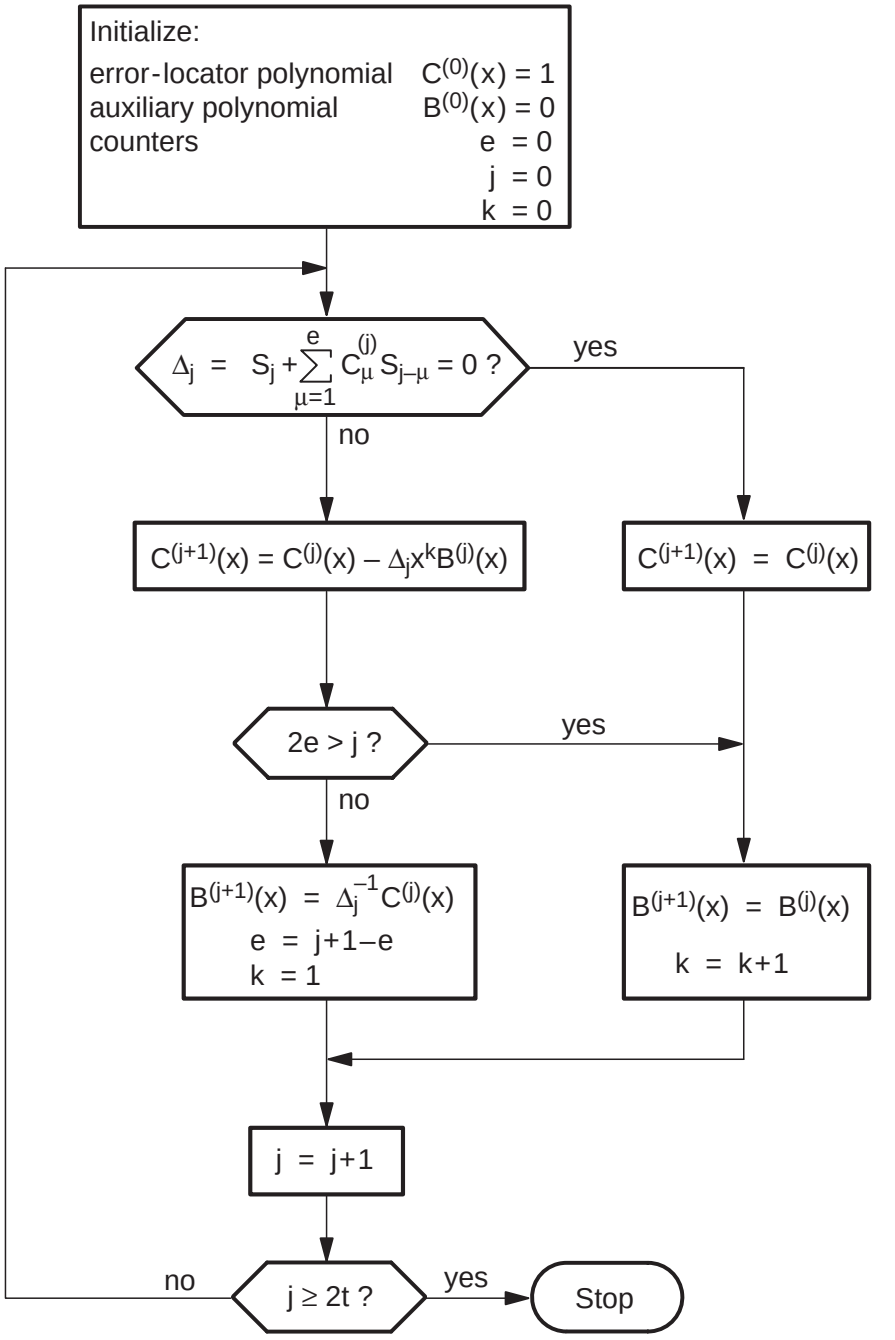

Figure 8.21. The Berlekamp-Massey algorithm

We present only a short descriptive explanation of the BMA. The counter e runs through the length of the filter and j runs through the length of the sequence of syndromes synthesized up until now. At $\Delta_j = 0$? a check is made to see whether we can create the next S_j using the current $C^{(j)}(x)$ and the current length e. If not then the filter coefficients are modified. Then at $2e_{\text{old}} \leq j$ the filter is extended: $e_{\text{new}} = j + 1 - e_{\text{old}} \geq e_{\text{old}} + 1$. Furthermore the auxiliary poly-

nomial $B(x)$ is used as a temporary buffer for the last-but-one $C(x)$. Without the extension of the filter, $B(x)$ remains unchanged. Finally, j is incremented in any case. During the whole algorithm, $j \geq e$ and $j \leq 2t-1$ are always valid so that at $\Delta_j = 0?$ only the known syndrome values $S_0, \ldots, S_{2t-1}$ are required.

An error pattern which is uncorrectable for the BMD decoder is detected, if, after the termination of the BMA, the degree of the calculated polynomial $C(x)$ is unequal to e or if $\deg C(x) = e > t$ or if the Chien search finds fewer than $\deg C(x)$ roots in the polynomial $C(x)$.

Example 8.10. Continuation of Example 8.7 with the $(7, 3, 5)_8$ RS code and $t = 2$ and $l = 0$. For the syndrome of the error pattern $e(x) = z^5x + z^6x^5$, $(S_0, S_1, S_2, S_3) = (z, z^3, z^6, z^3)$. The BMA works as follows:

e	k	j	Δ_j	$C^{(j+1)}(x)$	$2e > j$	$B^{(j+1)}(x)$
0	0	0	z	1	no	z^6
1	1	1	z^3	$1 + z^2x$	yes	z^6
1	2	2	z	$1 + z^2x + x^2$	no	$z^6 + zx$
2	1	3	z	$1 + z^6x + z^6x^2$	yes	$z^6 + zx$
2	2	4	Stop			

The comparison with Example 8.7 shows that the result $C(x) = C^{(4)}(x) = 1 + z^6x + z^6x^2$ is correct. ■

In Table 8.6, the necessary addition-and-multiplication operations for the decoding of RS codes are approximately summarized. The BMA reduces the effort for solving the key equation from t^3 to t^2 and is therefore unproblematic. The error correction in the time domain is less time-consuming than in the frequency domain. The biggest effort is caused by the partial transformations for calculating the syndrome, by the Chien search and by the encoder. However, the effort can be reduced by using fast algorithms [17, 18] as well as using various symmetries which exist in some cases. Many hints and ideas for implementing the algorithms are given in [79] and especially in [145]. For binary BCH codes, there are enormous simplifications which will be discussed in Section 8.7.

The equation (8.3.9) is valid regardless of where the parity frequencies are situated. So if, instead of $S_0, \ldots, S_{2t-1}$, the values $S_l, \ldots, S_{l+2t-1}$ are known, then we only have to shift the indices into the range of $0, \ldots, 2t-1$. The key equation (8.3.11) takes on the form

$$S_i + \sum_{\mu=1}^{\tau} C_\mu S_{j-\mu} = 0 \quad \text{for} \quad i = l+\tau, \ldots, l+2\tau-1 \tag{8.5.1}$$

and the switch condition on Δ_j in the BMA is to be changed into

$$\Delta_j = S_{j+l} + \sum_{\mu=1}^{e} C_\mu^{(j)} S_{j-\mu+l} = 0 \ ? \tag{8.5.2}$$

Table 8.6. Number of GF operations for RS coding

Partial algorithm	Number of add/mult-ops in $\mathbb{F}_{p^m}$
calculation of syndrome	$2t \cdot n$
Berlekamp-Massey algorithm	t^2 to $2t^2$
direct solution of key equation	t^3
recursive extension	$t(n-2t)$
partial IDFT	$t \cdot n$
calculation of $T(x)$	$t^2/2$
Forney algorithm	$2t^2$
Chien search	$t \cdot n$
subtraction of the error magnitudes	t additions
systematic encoder	$2t(n-2t)$

Of course, we still need to modify the recursive extension in Figure 8.18 and the calculation of $T(x)$ in Figure 8.19 accordingly.

8.5.2 The First Version of the Euclidean Algorithm (EA1)

Now, we will introduce two methods for solving the key equation, both based on the Euclidean Algorithm (EA, see Theorem A.8), whose advantages and disadvantages over the BMA have to be considered for each case. The EA methods not only determine the error-locator polynomial $C(x)$ of degree τ from the syndrome $S(x)$ of degree $\leq 2t-1$, but also the error-evaluator polynomial $T(x)$ of degree $\tau-1$. Let $t = (d_{\min}-1)/2 = (n-k)/2$ be the maximum number of errors and τ be the actual number of errors with $\tau \leq t$.

Theorem 8.9 (Version EA1). *We use the polynomials*

$$a(x) = r_{-2}(x) = x^{2t} \quad , \quad b(x) = r_{-1}(x) = S(x) \tag{8.5.3}$$

for the start configuration of the EA. The recursive scheme of Theorem A.8 in $\mathbb{F}_{p^m}[x]$ *leads to a series of polynomials* $r_i(x)$ *of decreasing degree such that there exists an index* λ *with*

$$\deg\, r_{\lambda-1}(x) \geq t \quad \textit{and} \quad \deg\, r_\lambda(x) < t. \tag{8.5.4}$$

At the same time the recursion for the polynomial $t_i(x)$ *is computed. With the constant* $\gamma = 1/t_\lambda(0) \in \mathbb{F}_{p^m}$ *we have*

$$C(x) = \gamma \cdot t_\lambda(x) \quad , \quad T(x) = -\gamma \cdot r_\lambda(x) \tag{8.5.5}$$

for the error-locator and the error-evaluator polynomial.

Proof. Although the recursion for the polynomial $s_i(x)$ is used for this proof, it does not have to be calculated in practical applications of the algorithm. According to (A.7.6), the recursive polynomials are linked as

$$r_\lambda(x) = s_\lambda(x)x^{2t} + t_\lambda(x)S(x). \tag{8.5.6}$$

According to (A.7.16),

$$\deg\, t_\lambda(x) = \underbrace{\deg\, a(x)}_{=2t} - \underbrace{\deg\, r_{\lambda-1}(x)}_{\geq t} \leq t.$$

The main equation (8.4.2) can be written as

$$(x^n - 1)T(x) = C(x)E(x) = C(x)\left(S(x) + \sum_{i=2t}^{n-1} E_i x^i\right)$$

and with an appropriate polynomial $L(x)$ we then have

$$-T(x) = C(x)S(x) + x^{2t}L(x). \tag{8.5.7}$$

The multiplication of (8.5.6) by $C(x)$ together with (8.5.7) leads to

$$\begin{aligned}\underbrace{C(x)r_\lambda(x)}_{\deg<t+\tau} &= C(x)s_\lambda(x)x^{2t} + C(x)S(x)t_\lambda(x)\\ &= \underbrace{-T(x)t_\lambda(x)}_{\deg<\tau+t} + \underbrace{\Big(C(x)s_\lambda(x) - L(x)t_\lambda(x)\Big)x^{2t}}_{\deg\geq 2t}.\end{aligned}$$

Comparing the degrees gives us

$$C(x)s_\lambda(x) = L(x)t_\lambda(x), \tag{8.5.8}$$
$$C(x)r_\lambda(x) = -T(x)t_\lambda(x). \tag{8.5.9}$$

The error-locator polynomial $C(x)$ has the roots z^{-i} for $i \in I$. Since these zeros are only singular, $C'(z^{-i}) \neq 0$ must be valid for the first derivative and the Forney algorithm (8.4.7) implies that $T(z^{-i}) \neq 0$ for $i \in I$. Thus, according to (8.5.9), $C(x)$ has to be a factor of $t_\lambda(x)$, hence, there exists a polynomial $\beta(x) = t_\lambda(x)/C(x)$. Trivially, $\beta(x)$ is a factor of $t_\lambda(x)$ and, according to (8.5.8), $\beta(x)$ is also a factor of $s_\lambda(x)$. However, according to (A.7.10), $s_\lambda(x)$ and $t_\lambda(x)$ do not have a common divisor, thus $\beta(x) = \gamma^{-1}$ is a constant that is determined from $\gamma^{-1} = \beta(0) = t_\lambda(0)/C(0) = t_\lambda(0)$. Thus $C(x) = \gamma \cdot t_\lambda(x)$ is obvious and $T(x) = -\gamma \cdot r_\lambda(x)$ is implied by (8.5.9). ■

Example 8.11. As in Example 8.7, we will consider the $(7,3,5)_8$ RS code with $t=2$ and the error pattern $e(x)=z^5x+z^6x^5$ with $S(x)=z+z^3x+z^6x^2+z^3x^3$. The EA leads to the recursive scheme

$$\underbrace{x^4}_{r_{-2}(x)} = \underbrace{z^4x}_{\alpha_0(x)} \cdot \underbrace{(z^3x^3+z^6x^2+z^3x+z)}_{r_{-1}(x)} + \underbrace{(z^3x^3+x^2+z^5x)}_{r_0(x)}$$

$$\underbrace{(z^3x^3+z^6x^2+z^3x+z)}_{r_{-1}(x)} = \underbrace{1}_{\alpha_1(x)} \cdot \underbrace{(z^3x^3+x^2+z^5x)}_{r_0(x)} + \underbrace{(z^2x^2+z^2x+z)}_{r_1(x)}$$

$$\underbrace{(z^3x^3+x^2+z^5x)}_{r_0(x)} = \underbrace{(zx+z^6)}_{\alpha_2(x)} \cdot \underbrace{(z^2x^2+z^2x+z)}_{r_1(x)} + \underbrace{(x+1)}_{r_2(x)}.$$

Thus $\lambda=2$. According to (A.7.3), the $t_i(x)$ recursion is

$$\begin{array}{lllll} t_0(x) & = & t_{-2}(x)-\alpha_0(x)t_{-1}(x) & = & z^4x \\ t_1(x) & = & t_{-1}(x)-\alpha_1(x)t_0(x) & = & 1+z^4x \\ t_2(x) & = & t_0(x)-\alpha_2(x)t_1(x) & = & z^6+z^5x+z^5x^2. \end{array}$$

Thus $\gamma=1/t_\lambda(0)=z$ and, according to Theorem 8.9, $C(x)=zt_2(x)=1+z^6x+z^6x^2$ as well as $T(x)=-zr_2(x)=z+zx$, which are equal to the results of Examples 8.7 and 8.9. ■

8.5.3 The Second Version of the Euclidean Algorithm (EA2)

An alternative application of the EA for determining $C(x)$ and $T(x)$ is based on the *complementary error-locator polynomial* of degree $n-|I|=n-\tau$:

$$C^*(x)=\prod_{i\notin I}(1-xz^i). \tag{8.5.10}$$

While keeping $\{z^0,\ldots,z^{n-1}\}=\{z^{-0},\ldots,z^{-(n-1)}\}$ in mind, according to (7.2.13),

$$C(x)C^*(x)=\prod_{i=0}^{n-1}(1-xz^i)=\left(\prod_{i=0}^{n-1}(-z^i)\right)\cdot\left(\prod_{i=0}^{n-1}(x-z^{-i})\right)$$

$$=\Delta\cdot(x^n-1) \quad \text{with} \quad \Delta\in\mathbb{F}_{p^m}. \tag{8.5.11}$$

For $i\in I$, $C^*(x)$ has the roots z^{-i} and $e_i=-E(z^{-i})=0$. So $C^*(x)$ splits up into linear factors which are also linear factors of $E(x)$, thus $C^*(x)$ is a factor of x^n-1 as well as of $E(x)$. Since $C(x)$ is to have a minimum degree and $C^*(x)$ is to have a maximum degree,

$$C^*(x)=\mathrm{GCD}\Big(E(x),x^n-1\Big). \tag{8.5.12}$$

Now, we perform a cyclic shift of $n-2t$ positions to the right on $E(x)$, such that the known syndromes are in the higher frequencies and the unknown error magnitudes are in the lower frequencies:

$$E_s(x) = R_{x^n-1}[x^{n-2t}E(x)] \;\leftrightarrow\; (E_{2t},\ldots,E_{n-1},S_0,\ldots,S_{2t-1}). \tag{8.5.13}$$

According to Theorem 7.8, this implies a multiplication for $e_s(x) \circ\!\!-\!\!\bullet E_s(x)$ in the time domain, hence $e_{s,i} = e_i \cdot z^{-i(n-2t)}$ for the coefficients of $e_s(x)$. Since the error magnitudes do not move, $E(x)$ and $E_s(x)$ have the same roots, thus

$$C^*(x) = \mathrm{GCD}\Big(E_s(x), x^n - 1\Big). \tag{8.5.14}$$

Corresponding to (8.4.2), there is a representation

$$C(x)E_s(x) = T_s(x)(x^n - 1), \tag{8.5.15}$$

where $T_s(x)$ is the error-evaluator polynomial for $e_s(x)$.

Theorem 8.10 (Version EA2). *We choose the polynomials*

$$a(x) = r_{-2}(x) = x^n - 1 \quad , \quad b(x) = r_{-1}(x) = E_s(x) \tag{8.5.16}$$

for the start configuration of the EA. The recursive scheme of Theorem A.8 in $\mathbb{F}_{p^m}[x]$ *leads to a series of polynomials* $r_i(x)$ *of decreasing degree such that there exists a minimum index* λ *with*

$$\deg\, r_{\lambda+1}(x) < n - t. \tag{8.5.17}$$

At the same time, the recursions for the polynomials $s_i(x)$ *and* $t_i(x)$ *are computed. With the constant* $\gamma = 1/t_{\lambda+1}(0) \in \mathbb{F}_{p^m}$ *we have*

$$C(x) = \gamma \cdot t_{\lambda+1}(x) \quad , \quad T_s(x) = -\gamma \cdot s_{\lambda+1}(x) \tag{8.5.18}$$

for the error-locator and the error-evaluator polynomial.

Proof. According to Theorem A.8, there exists an index λ with

$$r_\lambda(x) = \mathrm{GCD}(x^n - 1, E_s(x)) = C^*(x) \quad \text{and} \quad r_{\lambda+1}(x) = 0.$$

So, for $i \le \lambda$, $\deg\, r_i(x) \ge \deg\, r_\lambda(x) = n - \tau \ge n - t$ and for $i \ge \lambda+1$, $r_i(x) = 0$. According to (A.7.5),

$$(-1)^\lambda t_{\lambda+1}(x) = \frac{x^n-1}{\mathrm{GCD}(x^n-1, E_s(x))} = \frac{x^n-1}{C^*(x)} \;=\; \Delta^{-1}C(x),$$
$$-(-1)^\lambda s_{\lambda+1}(x) = \frac{E_s(x)}{\mathrm{GCD}(x^n-1, E_s(x))} = \frac{E_s(x)}{C^*(x)} \;=\; \Delta^{-1}T_s(x),$$

where (8.5.11) implies the right hand side of the equalities. The polynomial $C(x) = \Delta(-1)^\lambda t_{\lambda+1}(x)$ with $x = 0$ implies that $\gamma = \Delta(-1)^\lambda = 1/t_{\lambda+1}(0)$. So the proof is complete. Finally, (A.7.15) and (A.7.16) can be used to verify the correct degrees of $C(x)$ and $T_s(x)$. ■

Because of the special relation of the degrees, the EA can actually be performed, although only the upper $2t$ powers of $E_s(x)$ are known. Here is an example that makes this quite clear:

Example 8.12. We make the same presumptions as for Example 8.11 with the error pattern $e(x) = z^5x + z^6x^5$. The shift in the frequency domain leads to

$$e_s(x) = z^{5-1\cdot3}x + z^{6-5\cdot3}x^5 = z^2x + z^5x^5$$

in the time domain. In fact, we do not know the polynomials $e_s(x)$ or $E_s(x)$ completely but only

$$E_s(x) = (E_4, E_5, E_6, S_0, S_1, S_2, S_3) = (, z, z^3, z^6, z^3),$$

where the coefficients E_4, E_5, E_6 are unknown. With the start configuration of $r_{-2}(x) = x^n - 1 = x^7 + 1$ and $r_{-1}(x) = E_s(x)$ the EA gives us the recursive scheme

$$\underbrace{(x^7+1)}_{r_{-2}(x)} = \underbrace{(z^4x+1)}_{\alpha_0(x)} \cdot \underbrace{(z^3x^6 + z^6x^5 + z^3x^4 + zx^3 + \ldots)}_{r_{-1}(x)} + \underbrace{(z^2x^5 + z^2x^4 + \ldots)}_{r_0(x)}$$

$$\underbrace{(z^3x^6 + z^6x^5 + z^3x^4 + zx^3 + \ldots)}_{r_{-1}(x)} = \underbrace{(zx+z^2)}_{\alpha_1(x)} \cdot \underbrace{(z^2x^5 + z^2x^4 + \ldots)}_{r_0(x)} + \underbrace{(x^4\ldots)}_{r_1(x)}.$$

With $\deg r_1(x) < n - t = 5$ we have $\lambda = 0$. So inevitably $r_1(x) = 0$. According to (A.7.3),

$$t_0(x) = t_{-2}(x) - \alpha_0(x)t_{-1}(x) = z^4x + 1$$
$$t_1(x) = t_{-1}(x) - \alpha_1(x)t_0(x) = 1 + (zx + z^2)(z^4x + 1) = z^5x^2 + z^5x + z^6$$

for the $t_i(x)$ recursion, and

$$s_0(x) = s_{-2}(x) - \alpha_0(x)s_{-1}(x) = 1$$
$$s_1(x) = s_{-1}(x) - \alpha_1(x)s_0(x) = zx + z^2$$

for the the $s_i(x)$-recursion. The polynomials of (8.5.18) with $\gamma = 1/t_1(0) = z$ give us our usual results $C(x) = zt_1(x) = 1 + z^6x + z^6x^2$ as well as $T_s(x) = zs_1(x) = z^3 + zx$. The Forney algorithm with

$$e_{s,i} = \frac{T_s(z^{-i})}{C'(z^{-i})} = \frac{z^3 + z^{2-i}}{z^6} z^i = \left\{ \begin{array}{ll} z^2 & i = 1 \\ z^5 & i = 5 \end{array} \right\}$$

implies the correct result for $e_s(x)$. ■

8.6 Error-and-Erasure Decoding with RS Codes

8.6.1 Overview and Decoding Capability

We presuppose a q-ary symmetric channel with erasures as a generalization of the BSEC of Figure 1.4:

$$\mathcal{A}_{\text{in}} = \mathbb{F}_q \quad , \quad \mathcal{A}_{\text{out}} = \mathbb{F}_q \cup \{?\}. \tag{8.6.1}$$

The demodulator decides on $y = ?$ (erasure), if the decision on a specific $y \in \mathbb{F}_q$ was too unreliable. The advantage of this method is that the correction of such an erasure (position known, transmitted symbol unknown) only requires one parity-check symbol, whereas to correct an error (position unknown, transmitted symbol and error symbol unknown) two parity-check symbols are required. Formally, the received word as an extension to (8.3.1) is described by

$$\boldsymbol{y} = \boldsymbol{a} + \boldsymbol{e} + \boldsymbol{v}, \tag{8.6.2}$$

where $\boldsymbol{a} \in \mathbb{F}_q^n$ is the codeword, $\boldsymbol{e} \in \mathbb{F}_q^n$ is the error word and $\boldsymbol{v} \in \{0, ?\}^n$ is the erasure word. As the components y_i and v_i can take on the value ?, the arithmetic operations between ? and $b \in \mathbb{F}_q$ are formally defined as

$$b{+}? = ? \quad , \quad b{\cdot}? = ? \text{ für } b \neq 0 \quad , \quad 0{\cdot}? = 0. \tag{8.6.3}$$

In addition to the set of unknown error locations I and the error-locator polynomial $C(x)$, we will now introduce the *set of known erasure locations* I_v and the *erasure-locator polynomial* $C_v(x)$:

$$I = \{i \mid e_i \neq 0\} \quad , \quad C(x) = \prod_{i \in I} (1 - xz^i), \tag{8.6.4}$$

$$I_v = \{i \mid v_i = ?\} \quad , \quad C_v(x) = \prod_{i \in I_v} (1 - xz^i). \tag{8.6.5}$$

Per definition, an erasure is not considered to be an error, thus I and I_v are disjoint. Of course, I and $C(x)$ are unknown, whereas I_v and $C_v(x)$ are known. Let $\tau = |I| = \deg C(x)$ be the unknown number of errors and $\tau_v = |I_v| = \deg C_v(x)$ be the known number of erasures. The decoding method described below proves the following theorem.

Theorem 8.11 (Error-and-Erasure Decoding). *An $(n, n-2t, 2t+1)_q$ RS code corrects τ errors and τ_v erasures, if*

$$2\tau + \tau_v \;\leq\; 2t = d_{\min} - 1 = n - k. \tag{8.6.6}$$

For $\tau = 0$ and $\tau_v = n - k$, this corresponds to the statement of Theorem 4.8, because k arbitrary correct positions in the codeword uniquely determine the whole codeword. The calculation of the error probability for the error-and-erasure correction will be covered in Theorem 11.1?.

The decoding is performed in two major steps. The first step completely eliminates the influence of erasures by using a trick, so that the usual methods of error correction can be applied. Then the second step corrects the erasures.

8.6.2 First Step: Error Correction

For $i \in I_v$, $c_{v,i} = -C_v(z^{-i}) = 0$ and for $i \notin I_v$, $v_i = 0$. Thus $v_i c_{v,i} = 0$ for $0 \le i \le n-1$. The representation (8.6.2) implies that

$$\underbrace{y_i c_{v,i}}_{\tilde{y}_i} = \underbrace{a_i c_{v,i}}_{\tilde{a}_i} + \underbrace{e_i c_{v,i}}_{\tilde{e}_i} + \underbrace{v_i c_{v,i}}_{0}. \tag{8.6.7}$$

Since I and I_v are disjoint, the combination $e_i \neq 0$ and $c_{v,i} = 0$ is impossible, thus $e_i = 0$ is equivalent to $\tilde{e}_i = 0$. Therefore the influence of the erasures is completely eliminated in $\tilde{\boldsymbol{y}} = \tilde{\boldsymbol{a}} + \tilde{\boldsymbol{e}}$, and the error locations remain unchanged. However, the error magnitudes have been modified and the syndrome is reduced by τ_v components, as we will see shortly. According to Theorem 7.8, the transformation of (8.6.7) into the frequency domain leads to

$$\underbrace{R_{x^n-1}[-Y(x)C_v(x)]}_{\tilde{Y}(x)} = \underbrace{R_{x^n-1}[-A(x)C_v(x)]}_{\tilde{A}(x)} + \underbrace{R_{x^n-1}[-E(x)C_v(x)]}_{\tilde{E}(x)}. \tag{8.6.8}$$

The product $A(x)C_v(x)$ has the degree $\le (n-1) + \tau_v$, so that the upper τ_v powers are added to the lower powers by the modulo $(x^n - 1)$ operation:

$$\begin{aligned} \boldsymbol{A} &= (\underbrace{0, \ldots, 0}_{2t \text{ zeros}}, \underbrace{A_{2t}, \ldots, A_{n-1}}_{n-2t \text{ positions}}), \\ \tilde{\boldsymbol{A}} &= (\underbrace{\tilde{A}_0, \ldots, \tilde{A}_{\tau_v-1}}_{\tau_v \text{ positions}}, \underbrace{0, \ldots, 0}_{2t-\tau_v \text{ zeros}}, \underbrace{\tilde{A}_{2t}, \ldots, \tilde{A}_{n-1}}_{n-2t \text{ positions}}). \end{aligned} \tag{8.6.9}$$

Thus, for the syndrome, we can only use the $2t - \tau_v$ positions of the error word, which are left unchanged by the codeword, thus $\tilde{\boldsymbol{S}} = (\tilde{E}_{\tau_v}, \ldots, \tilde{E}_{2t-1})$ is the new syndrome. According to the presupposition (8.6.6), $2\tau \le 2t - \tau_v$, and therefore τ errors are correctable. For solving the key equation, we now have to note that, according to (8.3.10) or (8.5.2),

$$\tilde{E}_i + \sum_{\mu=1}^{\tau} C_\mu \tilde{E}_{i-\mu} = 0 \quad \text{for} \quad i = \tau_v + \tau, \ldots, \tau_v + 2\tau - 1. \tag{8.6.10}$$

Since all indices of $\tilde{E}_i$ are within the range of $\tau_v, \ldots, \tau_v + 2t - 1$, $\tilde{E}_i$ corresponds to the known syndrome $\tilde{S}_i$. So, the τ equations of (8.6.10) completely determine the τ unknowns $C_1, \ldots, C_\tau$.

So now the error-locator polynomial $C(x)$ is given. The error correction can be performed similarly to Subsection 8.4.1 or 8.4.2 or the errors can be interpreted as erasures for the decoding to come, since their locations are now known. Therefore, from now on we will only need to consider the correction of erasures.

8.6.3 Second Step: Erasure Correction

After the first step, we can now assume that the received word is superimposed by erasures only, so for $\boldsymbol{y} = \boldsymbol{a} + \boldsymbol{v}$,

$$\begin{aligned} &I_v = \{i \mid v_i = ?\} \quad , \quad C_v(x) = \prod_{i \in I_v} (1 - xz^i), \\ &\tau_v = |I_v| = \deg\, C_v(x) \leq 2t, \end{aligned} \tag{8.6.11}$$

where I_v, $C_v(x)$ and τ_v known. In the frequency domain,

$$\boldsymbol{Y} = \boldsymbol{A} + \boldsymbol{V} = (V_0, \ldots, V_{2t-1}, A_{2t} + V_{2t}, \ldots, A_{n-1} + V_{n-1}). \tag{8.6.12}$$

For the actual calculation, y_i is set to zero at the erasure locations. According to the first step, $v_i c_{v,i} = 0$ for $0 \leq i \leq n-1$, and according to Theorem 7.8, we have our usual implication of

$$R_{x^n-1}[C_v(x)V(x)] = 0, \tag{8.6.13}$$

or equivalently expressed by the cyclic convolution,

$$\sum_{\mu=0}^{\tau_v} C_{v,\mu} V_{(i-\mu) \bmod n} = 0 \quad \text{für} \quad i = 0, \ldots, n-1. \tag{8.6.14}$$

So a recursive extension of the components $V_{2t}, \ldots, V_{n-1}$ is possible from the known values $V_0, \ldots, V_{2t-1}$ with

$$\begin{aligned} V_i &= -\sum_{\mu=1}^{\tau_v} C_{v,\mu} V_{i-\mu} \quad \text{for} \quad i = n-2t, \ldots, n-1 \\ &= \text{function of } (V_{i-1}, \ldots, V_{i-\tau_v}). \end{aligned} \tag{8.6.15}$$

With the inverse transform $v_i = -V(z^{-i})$, erasure correction can be performed in the frequency domain. An alternative is the erasure correction in the time domain: according to (8.6.13), there exists an error-evaluator polynomial $T_v(x)$ of degree $\leq \tau_v - 1$ with

$$C_v(x)V(x) = T_v(x)(x^n - 1). \tag{8.6.16}$$

Then, corresponding to (8.4.4),

$$T_{v,j} = -\sum_{\mu=0}^{j} C_{v,\mu} V_{j-\mu} \quad \text{for} \quad j = 0, \ldots, \tau_v - 1 \tag{8.6.17}$$

and the Forney algorithm is

$$v_i = \frac{T_v(z^{-i})}{C_v'(z^{-i})} \cdot z^i \quad \text{for} \quad i \in I_v. \tag{8.6.18}$$

Example 8.13. As in Examples 8.7 to 8.12, we again consider the $(7, 3, 5)_8$ RS code with $t = 2$ and

$$\begin{array}{lll} \boldsymbol{a} = (z^5, z^3, z^2, z, z^4, z^6, 1) \circ\!\!-\!\!\bullet (0, 0, 0, 0, 1, z, z^2) & \text{unknown,} \\ \boldsymbol{e} = (0, 0, 0, 0, 0, z^5, 0) \quad \text{d.h.} \quad \tau = 1, I = \{5\} & \text{unknown,} \\ \boldsymbol{v} = (0, ?, 0, 0, ?, 0, 0) \quad \text{d.h.} \quad \tau_v = 2, I_v = \{1, 4\} & \text{known.} \end{array}$$

For the received word $\boldsymbol{y} = (z^5, ?, z^2, z, ?, z^6 + z^5 = z, 1)$, the receiver knows the erasure-locator polynomial $C_v(x) = (1 - xz^1)(1 - xz^4) = 1 + z^2 x + z^5 x^2$.

Step 1: error correction or location of errors. The equation $c_{v,i} = -C_v(z^{-i}) = 1 + z^{2-i} + z^{5-2i}$ implies that $\boldsymbol{c}_v = (z, 0, z, 1, 0, z^3, z^3)$. Thus the receiver can create $\tilde{\boldsymbol{y}} = (z^6, 0, z^3, z, 0, z^4, z^3)$, according to (7.10.7). Since

$$\tilde{\boldsymbol{Y}} = (\tilde{Y}_0, \tilde{Y}_1, \tilde{E}_2, \tilde{E}_3, \tilde{Y}_4, \tilde{Y}_5, \tilde{Y}_6)$$

we only need to determine $\tilde{Y}_i = \tilde{y}(z^i) = \tilde{E}_i$ for $i = 2, 3$:

$$\begin{aligned} \tilde{E}_2 = \tilde{Y}_2 = \tilde{y}(z^2) &= z^6 + z^3(z^2)^2 + z(z^2)^3 + z^4(z^2)^5 + z^3(z^2)^6 = z^4 \\ \tilde{E}_3 = \tilde{Y}_3 = \tilde{y}(z^3) &= z^6 + z^3(z^3)^2 + z(z^3)^3 + z^4(z^3)^5 + z^3(z^3)^6 = z^2. \end{aligned}$$

The key equation (8.6.10) is $\tilde{E}_3 + C_1 \tilde{E}_2 = 0$ implying that $C_1 = -\tilde{E}_3 / \tilde{E}_2 = z^5$. Thus $C(x) = C_0 + C_1 x = 1 - xz^5$ which implies the correct result for $I = \{5\}$. From now on we will treat this located error as an erasure.

Step 2: erasure correction for $I_v = \{1, 4, 5\}$ with $\tau_v = 3$. The received word is $\boldsymbol{y} = (z^5, ?, z^2, z, ?, ?, 1)$ and the error-locator polynomial is

$$C_v(x) = (1 - xz^1)(1 - xz^4)(1 - xz^5) = 1 + z^3 x + z^4 x^2 + z^3 x^3.$$

With 0 for ?, $Y_i = y(z^i) = z^5 + z^{2+2i} + z^{1+3i} + z^{6i}$ implies that

$$\boldsymbol{Y} = (0, z, z^2, 0, z, 0, z^3) = (V_0, V_1, V_2, V_3, Y_4, Y_5, Y_6).$$

The recursive extension with $V_i = C_{v,1} V_{i-1} + C_{v,2} V_{i-2} + C_{v,3} V_{i-3}$ as in (8.6.15) leads to

$$\begin{array}{lllllll} V_4 & = & z^3 V_3 + z^4 V_2 + z^3 V_1 & = & 0 + z^6 + z^4 & = & z^3 \\ V_5 & = & z^3 V_4 + z^4 V_3 + z^3 V_2 & = & z^6 + 0 + z^5 & = & z \\ V_6 & = & z^3 V_5 + z^4 V_4 + z^3 V_3 & = & z^4 + 1 + 0 & = & z^5. \end{array}$$

Thus, the correct result is

$$\boldsymbol{A} = \boldsymbol{Y} - \boldsymbol{V} = (0, 0, 0, 0, z - z^3 = 1, 0 - z = z, z^3 - z^5 = z^2).$$

Inverse discrete Fourier transform (IDFT) leads to $a_i = -A(z^{-i})$ for $i \in I_v$. For the alternative erasure correction in the time domain, we first determine $T_v(x)$ according to (8.6.17):

$$\begin{array}{lcll} T_{v,0} & = & C_{v,0}V_0 & = & 0 \\ T_{v,1} & = & C_{v,0}V_1 + C_{v,1}V_0 & = & z \\ T_{v,2} & = & C_{v,0}V_2 + C_{v,1}V_1 + C_{v,2}V_0 & = & z. \end{array}$$

Thus $T_v(x) = zx + zx^2$. The first derivative of $C_v(x)$ is

$$C'_v(x) = z^3 + 2z^4x + 3z^3x^2 = z^3 + z^3x^2.$$

Then, according to (8.6.18), the Forney algorithm is

$$v_i = \frac{T_v(z^{-i})}{C'_v(z^{-i})} z^i = \frac{z^{1-i} + z^{1-2i}}{z^3 + z^{3-2i}} z^i.$$

For $i \in I_v$ we get the correct results $v_1 = z^3$, $v_4 = z^4$ and $v_5 = z^6$. ■

A modified Berlekamp-Massey algorithm (BMA) for error-and-erasure decoding can be found in [64]. A correspondingly modified Euclidean algorithm (EA) and its implementation are introduced in [224].

8.6.4 Encoding per Erasure Decoder

An erasure-location decoder can also be used for the systematic encoding of RS codes as follows. According to Theorem 3.8, the k information symbols uniquely determine the whole codeword or the $n - k$ parity-check symbols. For the encoding-by-decoding method a word is created containing the information symbols but the unknown parity-check symbols are replaced by erasures. Then the erasure-locator decoder determines the desired parity-check symbols at the erasure locations without any further manipulations of the other positions. The advantage of this method is that encoder and decoder are identical.

8.7 Decoding of Binary BCH Codes

In contrast to the general RS codes, the decoding algorithm for binary BCH codes is so much simpler that we will discuss the corresponding methods separately in this section. We presuppose the $(2^m - 1, k, d_{\min})_2$ narrow-sense BCH

code as in (8.2.2) with $2t = d - 1 \leq d_{\min} - 1 \leq n - k$ and $n = 2^m - 1$. So we need to determine the syndromes

$$S_i = y(z^i) = e(z^i) \quad \text{für} \quad i = 1, \ldots, 2t. \tag{8.7.1}$$

Unfortunately, it is of no use for decoding purposes that there are further frequencies in the received word which are not affected by the transmitted codeword, in other words, decoding can only take place with regards to the designed distance d although the actual minimum distance $d_{\min}$ might be larger.

The calculation of $\boldsymbol{S} = (S_1, \ldots, S_{2t})$ in $\mathbb{F}_{2^m}$ can be simplified as follows. According to Theorem A.4, for the received word $y(x) \in \mathbb{F}_2[x]$ and the minimal polynomials $f_{[z^i]}(x) \in \mathbb{F}_2[x]$, there exist polynomials $\alpha_i(x), r_i(x) \in \mathbb{F}_2[x]$ with

$$y(x) = \alpha_i(x) f_{[z^i]}(x) + r_i(x) \quad \text{with} \quad \deg\, r_i(x) < \deg\, f_{[z^i]}(x) \leq m. \tag{8.7.2}$$

For $i = 1, \ldots, 2t$, many of the minimal polynomials are, of course, identical, so many of the polynomials $r_i(x)$ are also identical. For $x = z^i$, (8.7.2) implies that

$$S_i = y(z^i) = r_i(z^i). \tag{8.7.3}$$

Since $y(x)$ can have a degree up to $2^m - 2$, but $r_i(x)$ can only have a much smaller degree of a maximum of m, calculating the syndrome over $r_i(x)$ is not as time-consuming as using (8.7.1). Furthermore, according to Theorem 7.9, we only need to determine the syndromes with an odd index because $S_{2i} = S_i^2$. Similar to (8.3.13), the syndrome can also be represented by

$$S_r = E_r = \sum_{\mu=0}^{n-1} e_\mu z^{r\mu} = \sum_{i \in I} z^{ri}. \tag{8.7.4}$$

In the binary case, $e_i \neq 0$ is equivalent to $e_i = 1$, so determining the error locations is the last step of decoding. As in Definiton 8.4, each error pattern $e(x)$ is mapped to the set of unknown error locations

$$\begin{aligned} I &= \{i \mid 0 \leq i \leq n-1 \ \wedge \ e_i \neq 0\} \\ &= \{i \mid 0 \leq i \leq n-1 \ \wedge \ e_i = 1\} \end{aligned} \tag{8.7.5}$$

with $|I| = \tau \leq t$ and $e(x)$ is also mapped to the error-locator polynomial

$$C(x) = \prod_{i \in I} (1 - xz^i) = 1 + C_1 x + \cdots + C_\tau x^\tau. \tag{8.7.6}$$

The key equation changes from (8.3.12) to

$$\begin{pmatrix} -S_{\tau+1} \\ -S_{\tau+2} \\ \\ \vdots \\ \\ -S_{2\tau-1} \\ -S_{2\tau} \end{pmatrix} = \begin{pmatrix} S_1 & S_2 & \cdots & S_{\tau-1} & S_\tau \\ S_2 & S_3 & \cdots & S_\tau & S_{\tau+1} \\ \\ \vdots & \vdots & & \vdots & \vdots \\ \\ S_{\tau-1} & S_\tau & \cdots & S_{2\tau-3} & S_{2\tau-2} \\ S_\tau & S_{\tau+1} & \cdots & S_{2\tau-2} & S_{2\tau-1} \end{pmatrix} \cdot \begin{pmatrix} C_\tau \\ C_{\tau-1} \\ \\ \vdots \\ \\ C_2 \\ C_1 \end{pmatrix} \tag{8.7.7}$$

because of the different positions of the parity frequencies. The solution of this linear equation system can be simplified quite a bit (in contrast to the general case with RS codes), for which we will again have to derive the relation between $C(x)$ and $E(x)$ or $\boldsymbol{S}$. Similar to the Forney algorithm in Subsection 8.4.2, the first derivative of the error-locator polynomial is

$$\begin{aligned} C'(x) &= \left(\prod_{i\in I}(1-xz^i)\right)' = \sum_{i\in I}\left((-z^i)\prod_{j\in I\setminus\{i\}}(1-xz^j)\right) \\ &= -\sum_{i\in I} z^i \frac{C(x)}{1-xz^i}\,. \end{aligned}$$

The multiplication by x and using the sum of the geometric series leads to

$$xC'(x) = -C(x)\sum_{i\in I}\frac{xz^i}{1-xz^i} = -C(x)\sum_{r=1}^{\infty}\left(\sum_{i\in I} z^{ir}\right)x^r.$$

An appropriate series expansion gives us the representation

$$\begin{aligned} 0 &= xC'(x) + C(x)\sum_{r=1}^{\infty} E_{r \bmod n} z^i \\ &= (C_1x + 2C_2x^2 + 3C_3x^3 + \cdots + \tau C_\tau x^\tau) \\ &\quad + (1 + C_1x + C_2x^2 + \cdots + C_\tau x^\tau) \\ &\quad\quad \cdot (S_1x + S_2x^2 + \cdots + S_{2\tau}x^{2\tau} + E_{2\tau+1}x^{2\tau+1} + \cdots) \\ &= x(C_1 + S_1) + \\ &\quad x^2(2C_2 + S_2 + C_1S_1) + \\ &\quad x^3(3C_3 + S_3 + C_1S_2 + C_2S_1) + \cdots + \\ &\quad x^\tau(\tau C_\tau + S_\tau + C_1S_{\tau-1} + \cdots + C_{\tau-1}S_1) + \\ &\quad x^{\tau+1}(S_{\tau+1} + C_1S_\tau + \cdots + C_\tau S_1) + \cdots + \\ &\quad x^{2\tau}(S_{2\tau} + C_1S_{2\tau-1} + \cdots + C_\tau S_\tau) + \cdots \end{aligned}$$

The result is the following system of equations (note that $-1 = 1$ in $\mathbb{F}_2$):

$$\begin{pmatrix} S_1 \\ S_2 \\ S_3 \\ \vdots \\ S_\tau \\ S_{\tau+1} \\ \vdots \\ S_{2\tau} \end{pmatrix} = \begin{pmatrix} 1 & & & & & \\ S_1 & 2 & & & & \\ S_2 & S_1 & 3 & & & \\ \vdots & \vdots & \vdots & & & \\ S_{\tau-1} & S_{\tau-2} & S_{\tau-3} & \cdots & S_1 & \tau \\ S_\tau & S_{\tau-1} & S_{\tau-2} & \cdots & S_2 & S_1 \\ \vdots & \vdots & \vdots & & \vdots & \vdots \\ S_{2\tau-1} & S_{2\tau-2} & S_{2\tau-3} & \cdots & S_{\tau+1} & S_\tau \end{pmatrix} \cdot \begin{pmatrix} C_1 \\ \\ \vdots \\ \\ C_\tau \end{pmatrix}. \tag{8.7.8}$$

This system has 2τ equations to determine the τ unknowns $C_1, \ldots, C_\tau$ and thus to determine the error locations. Each component of the syndrome is

known, even if, in (8.7.8), τ is replaced by t. Since $S_{2i} = S_i^2$, we can drop every second equation and all syndromes can be described as powers of syndromes with odd indices: $S_2 = S_1^2$, $S_4 = S_1^4$, $S_6 = S_3^2$, $S_8 = S_1^8$ etc. Furthermore, $1 = 3 = 5 = 7 = \cdots = 1$ in $\mathbb{F}_2$, so for odd τ, (8.7.8) is reduced to

$$\begin{pmatrix} S_1 \\ S_3 \\ \vdots \\ S_\tau \\ S_{\tau+2} \\ \vdots \\ S_{2\tau-1} \end{pmatrix} = \begin{pmatrix} 1 & & & & & \\ S_2 & S_1 & 1 & & & \\ \vdots & \vdots & \vdots & & & \\ S_{\tau-1} & S_{\tau-2} & S_{\tau-3} & \cdots & S_1 & 1 \\ S_{\tau+1} & S_\tau & S_{\tau-1} & \cdots & S_3 & S_2 \\ \vdots & \vdots & \vdots & & \vdots & \vdots \\ S_{2\tau-2} & S_{2\tau-3} & S_{2\tau-4} & \cdots & S_\tau & S_{\tau-1} \end{pmatrix} \cdot \begin{pmatrix} C_1 \\ \\ \vdots \\ \\ C_\tau \end{pmatrix} \tag{8.7.9}$$

and for even τ it is reduced to

$$\begin{pmatrix} S_1 \\ S_3 \\ \vdots \\ S_{\tau-1} \\ S_{\tau+1} \\ \vdots \\ S_{2\tau-1} \end{pmatrix} = \begin{pmatrix} 1 & & & & & \\ S_2 & S_1 & 1 & & & \\ \vdots & \vdots & \vdots & & & \\ S_{\tau-2} & S_{\tau-3} & S_{\tau-4} & \cdots & 1 & \\ S_\tau & S_{\tau-1} & S_{\tau-2} & \cdots & S_2 & S_1 \\ \vdots & \vdots & \vdots & & \vdots & \vdots \\ S_{2\tau-2} & S_{2\tau-3} & S_{2\tau-4} & \cdots & S_\tau & S_{\tau-1} \end{pmatrix} \cdot \begin{pmatrix} C_1 \\ \\ \vdots \\ \\ C_\tau \end{pmatrix}. \tag{8.7.10}$$

In both cases the matrix is (τ, τ)-dimensional. The direct solution of this equation system is also known as *Peterson's direct-solution decoder*. We will now determine the direct solution for $\tau = 1, 2, 3$. For $\tau = 1$,

$$S_1 = 1 \cdot C_1 \;\Rightarrow\; C_1 = S_1.$$

For $\tau = 2$,

$$\begin{pmatrix} S_1 \\ S_3 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ S_1^2 & S_1 \end{pmatrix} \cdot \begin{pmatrix} C_1 \\ C_2 \end{pmatrix} \;\Rightarrow\; \begin{cases} C_1 = S_1 \\ C_2 = (S_1^3 + S_3)/S_1. \end{cases}$$

For $\tau = 3$,

$$\begin{pmatrix} S_1 \\ S_3 \\ S_5 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ S_1^2 & S_1 & 1 \\ S_1^4 & S_3 & S_1^2 \end{pmatrix} \cdot \begin{pmatrix} C_1 \\ C_2 \\ C_3 \end{pmatrix} \;\Rightarrow\; \begin{cases} C_1 = S_1 \\ C_2 = (S_1^2 S_3 + S_5)/(S_1^3 + S_3) \\ C_3 = S_1^3 + S_3 + S_1 C_2. \end{cases}$$

Further explicit formulas for $\tau = 4, 5, 6$ can be found, for example, in [95]. We can easily verify that these solutions satisfy the key equation in the standard form of (8.7.7).

The number τ of errors in the received word is, of course, unknown, but we can use a similar method as in Section 7.5 to determine τ. The system of equations (8.7.9) or (8.7.10) is stated with t instead of τ. If $\tau = t$ or $\tau = t - 1$,

then the matrix is non-singular [83, 105] and the equation system is solvable. For $\tau = t - 1$ we then have $C_t = 0$. However, if $\tau \leq t - 2$, then the matrix is singular, in which case the two bottom rows and the two right-hand columns are deleted and the resulting matrix is checked for singularity. This procedure is repeated until a non-singular matrix is found.

Example 8.14. Consider the $(15, 5, 7)_2$ BCH code of Example 8.5(3) with $t = 3$ and the generator polynomial

$$\begin{aligned} g(x) &= f_{[z]}(x) \cdot f_{[z^3]}(x) \cdot f_{[z^5]}(x) \\ &= (x^4 + x + 1)(x^4 + x^3 + x^2 + x + 1)(x^2 + x + 1) \\ &= x^{10} + x^8 + x^5 + x^4 + x^2 + x + 1. \end{aligned}$$

(1) Assume $e(x) = 1 + x^4 + x^{10}$ to be an error pattern, then the syndrome is $S_i = y(z^i) = 1 + z^{4i} + z^{10i}$, thus

$$\begin{array}{lcllcl} S_1 & = & 1 + z^4 + z^{10} & = & z^8 & \\ S_2 & = & 1 + z^8 + z^5 & = & z & = & S_1^2 \\ S_3 & = & 1 + z^{12} + 1 & = & z^{12} & \\ S_4 & = & 1 + z + z^{10} & = & z^2 & = & S_1^4 \\ S_5 & = & 1 + z^5 + z^5 & = & 1 & \\ S_6 & = & 1 + z^9 + 1 & = & z^9 & = & S_3^2. \end{array}$$

For the alternative calculation of the syndrome according to (8.7.3),

$$\begin{array}{lcll} r_1(x) & = & R_{x^4+x+1}[e(x)] & = & x^2 + 1 \\ r_3(x) & = & R_{x^4+x^3+x^2+x+1}[e(x)] & = & x^3 + x^2 + x + 1 \\ r_5(x) & = & R_{x^2+x+1}[e(x)] & = & 1 \end{array}$$

with the same result of

$$\begin{array}{lcllcllcl} S_1 & = & r_1(z) & = & z^2 + 1 & & & = & z^8 \\ S_2 & = & r_1(z^2) & = & z^4 + 1 & & & = & z \\ S_3 & = & r_3(z^3) & = & z^9 + z^6 + z^3 + 1 & & & = & z^{12} \\ S_4 & = & r_1(z^4) & = & z^8 + 1 & & & = & z^2 \\ S_5 & = & r_5(z^5) & = & 1 & & & = & 1 \\ S_6 & = & r_3(z^6) & = & z^3 + z^{12} + z^6 + 1 & & & = & z^9. \end{array}$$

The equation system (8.7.9) is

$$\begin{pmatrix} z^8 \\ z^{12} \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ z & z^8 & 1 \\ z^2 & z^{12} & z \end{pmatrix} \cdot \begin{pmatrix} C_1 \\ C_2 \\ C_3 \end{pmatrix} \quad \Rightarrow \quad \begin{cases} C_1 = z^8 \\ C_2 = z^{13} \\ C_3 = z^{14} \end{cases}$$

and the result $C(x) = 1 + z^8 x + z^{13} x^2 + z^{14} x^3 = (1 + xz^0)(1 + xz^4)(1 + xz^{10})$ is obviously the correct error-locator polynomial.

(2) Consider the error pattern $e(x) = 1 + x^4$ with the syndrome

$$(S_1, S_2, S_3, S_4, S_5, S_6) = (z, z^2, z^{11}, z^4, z^{10}, z^7).$$

The equation system (8.7.9) is

$$\begin{pmatrix} z \\ z^{11} \\ z^{10} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ z^2 & z & 1 \\ z^4 & z^{11} & z^2 \end{pmatrix} \cdot \begin{pmatrix} C_1 \\ C_2 \\ C_3 \end{pmatrix} \quad \Rightarrow \quad \begin{cases} C_1 = z \\ C_2 = z^4 \\ C_3 = 0 \end{cases}$$

und the result $C(x) = 1 + zx + z^4x^2 = (1 + xz^0)(1 + xz^4)$ is the correct error-locator polynomial.

(3) Now take a look at the error pattern $e(x) = x^4$ with the syndrome

$$(S_1, S_2, S_3, S_4, S_5, S_6) = (z^4, z^8, z^{12}, z, z^5, z^9).$$

The equation system (8.7.9) is

$$\begin{pmatrix} z^4 \\ z^{12} \\ z^5 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ z^8 & z^4 & 1 \\ z & z^{12} & z^8 \end{pmatrix} \cdot \begin{pmatrix} C_1 \\ C_2 \\ C_3 \end{pmatrix}.$$

The matrix is singular since its determinant is zero. Reducing it by one or two dimensions leads to

$$\begin{pmatrix} z^4 \\ z^{12} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ z^8 & z^4 \end{pmatrix} \cdot \begin{pmatrix} C_1 \\ C_2 \end{pmatrix} \quad \text{or to} \quad z^4 = 1 \cdot C_1.$$

In both cases the solution is $C_1 = z^4$ and $C_2 = 0$, so again the error-locator polynomial $C(x) = 1 + xz^4$ is correct. ■

The condition in the Berlekamp-Massey algorithm (BMA) takes on the form (8.5.2) with $l = 1$ for binary narrow-sense BCH codes. Elementary operations lead to $\Delta_0 = S_1$, $\Delta_1 = S_2$, $\Delta_2 = S_3 + S_1S_2$. Furthermore, according to [17],

$$\Delta_3 = \Delta_5 = \Delta_7 = \cdots = 0, \tag{8.7.11}$$

so every second iteration in the BMA can be omitted.

Example 8.15. Continuation of Example 8.14(1) with the syndrome

$$(S_1, S_2, S_3, S_4, S_5, S_6) = (z^8, z, z^{12}, z^2, 1, z^9).$$

The BMA works as follows:

e	k	j	Δ_j	$C^{(j+1)}(x)$	$2e > j$	$B^{(j+1)}(x)$
0	0	0	z^8	1	no	z^7
1	1	1	z	$1 + z^8x$	yes	z^7
1	2	2	z^8	$1 + z^8x + x^2$	no	$z^7 + x$
2	1	3	0	$1 + z^8x + x^2$	yes	$z^7 + x$
2	2	4	z^{14}	$1 + z^8x + z^{13}x^2 + z^{14}x^3$	no	$z + z^9x + zx^2$
3	1	5	0	$1 + z^8x + z^{13}x^2 + z^{14}x^3$	yes	$z + z^9x + zx^2$
3	2	6	stop			

The comparison with Example 8.14(1) reveals that the result $C(x) = C^{(6)}(x) = 1 + z^8x + z^{13}x^2 + z^{14}x^3$ is correct. ■

8.8 Modifications to RS und BCH Codes

8.8.1 Overview

We will now apply the code modifications of Section 5.5 to RS and BCH codes. The methods of

- expanding (add parity-check symbols, keep k)
- puncturing (remove parity-check symbols, keep k)
- lengthening (add information symbols, keep $n-k$)
- shortening (remove information symbols, keep $n-k$)

applied to an $(n, k, d_{\min})_q$ code $\mathcal{C}$ create an $(n', k', d'_{\min})_q$ code $\mathcal{C}'$ with a different block length but the same symbol size q. Starting with the primitive block length $n = p^m - 1$ we can construct almost all other block lengths n', which is extremely important for many applications. However, the modification of cyclic codes often results in non-cyclic codes. Other methods such as transitions to

- subcodes ($\mathcal{C}' \subset \mathcal{C}$, also called expurgating)
- extension codes ($\mathcal{C} \subset \mathcal{C}'$, also called augmenting)

are often also considered to be code modifications where the block length remains the same. These relatively simple modifications are represented by divisibility relationships between the generator polynomials (or parity-check polynomials) and will not be re-discussed here (see Problems 6.6 and 8.10).

A further possibility for non-primitive block lengths is provided by generalized BCH codes with block lengths of the form $n = m(2^{mr} - 1)$, mentioned at the end of Subsection 8.2.1. However, they require a more time-consuming mathematical discussion and will also not be covered here.

8.8.2 Expansion

Expansion adds additional parity-check bits to the codeword, the objective being an increased minimum distance, of course:

$$n' > n, \quad k' = k, \quad n' - k' > n - k, \quad d'_{\min} \geq d_{\min}, \quad R' < R. \tag{8.8.1}$$

The following method for expanding RS codes generalizes the principle of Theorem 5.11, however, without presuming an odd minimum distance:

Theorem 8.12. *A $(q-1, k, d)_q$ RS code $\mathcal{C}$ with $q = p^m$ can be easily expanded to a $(q, k, d+1)_q$ MDS code $\mathcal{C}'$ as follows. From $\boldsymbol{a} = (a_0, \ldots, a_{q-2}) \in \mathcal{C}$, $\boldsymbol{a}' = (a_0, \ldots, a_{q-2}, a_{q-1}) \in \mathcal{C}'$ is created, where the additional parity-check symbol of $a(x)$ is calculated at $x = z^{l-1}$:*

$$a_{q-1} = -a(z^{l-1}) = -A_{l-1} \tag{8.8.2}$$

For narrow-sense RS codes with $l = 1$, $a_{q-1} = -\sum_{i=0}^{q-2} a_i$ or equivalently $\sum_{i=0}^{q-1} a'_i = 0$.

Proof. The linearity of $\mathcal{C}'$ is apparent. The RS code $\mathcal{C}$ is generated by $g(x) = (x - z^l) \cdots (x - z^{l+d-2})$. Let $\boldsymbol{a} \in \mathcal{C}$ with $w_H(\boldsymbol{a}) = d$. If $a_{q-1} \neq 0$, then $w_H(\boldsymbol{a}') = w_H(\boldsymbol{a}) + 1 = d + 1$. However, if $a_{q-1} = -a(z^{l-1}) = 0$, then $x - z^{l-1}$ has to be a factor of $a(x)$, but not contained in $g(x)$. Thus, there exists a representation $a(x) = u(x)(x - z^{l-1})g(x)$ with suitable $u(x)$ and therefore $a(x) \in \mathcal{C}''$, where $\mathcal{C}''$ is created by the generator polynomial $g(x) = (z - z^{l-1})(x - z^l) \cdots (x - z^{l+d-2})$ and the designed distance is $d'' = d''_{\min} = d + 1$. Thus $w_H(\boldsymbol{a}') = w_H(\boldsymbol{a}) \geq d + 1$. Therefore $d'_{\min} \geq d + 1$ is proved.

The result above and $d'_{\min} \leq n' - k' + 1 = n - k + 2 = d + 1$ lead to $d'_{\min} = d + 1 = n' - k' + 1$, hence, $\mathcal{C}'$ is an MDS code. ■

The parity-check matrix of the expanded RS code is created by adding an extra row of ones and a column of zeros to the parity-check matrix of (8.1.7) as done in (5.5.7).

A further (double) expansion gives us a $(q + 1, k, d + 2)_q$ MDS code, which is less important for practice, by using [17, 144]

$$a_q = -a(z^{l+d-1}). \tag{8.8.3}$$

Furthermore, there are some more q-ary triple expanded MDS codes with $n = q + 2$.

The decoding of expanded codes and an accordingly modified Berlekamp-Massey algorithm (BMA) are described in [17, 64, 95]. Of course, BCH codes can also be expanded: starting from a $(2^m - 1, k, 2t + 1)_2$ code the method of Theorem 5.11 creates a $(2^m, k, 2t+2)_2$ code. Expanded BCH codes are discussed extensively in [17].

8.8.3 Puncturing

For puncturing, parity-check symbols are deleted from the codeword:

$$n' < n, \quad k' = k, \quad n' - k' < n - k, \quad d'_{\min} \leq d_{\min}, \quad R' > R. \tag{8.8.4}$$

In contrast to the expansion, where the MDS property is only preserved for simple or double expansion, it is always preserved for puncturing:

Theorem 8.13. *A punctured MDS code remains an MDS code.*

Proof. After deleting r parity-check symbols, the minimum distance is decreased by r at the most and the Singleton bound leads to

$$n' - k + 1 = n - r - k + 1 = d_{\min} - r \leq d'_{\min} \leq n' - k' + 1 = n' - k + 1.$$

The statement of the theorem is directly implied by Theorem 4.8. ■

In particular, punctured RS codes are still MDS codes where any arbitrary selection of components can be punctured. So, together with expansion there are $(n, k, d)_q$ MDS codes for all block lengths in the range of $k \leq n \leq q + 1$.

The decoding of punctured RS codes can be implemented by an error- or erasure-locator decoder without restrictions or losses: expand the received word to a word of primitive block length by interpreting the r punctured positions as erasures. According to Theorem 8.11, the maximum number of $\tau = (d'_{\min} - 1)/2$ errors can be corrected for $\tau_v = r$ erasures, since $2\tau + \tau_v = d'_{\min} + r - 1 = d_{\min} - 1$.

8.8.4 Shortening

Shortening codes means deleting information bits, for instance, information symbols are set to zero for encoding and are not transmitted:

$$n' < n, \quad k' < k, \quad n' - k' = n - k, \quad d'_{\min} \geq d_{\min}, \quad R' < R. \tag{8.8.5}$$

The information polynomials are restricted to deg $u(x) \leq k' - 1$ such that $a(x) = u(x)g(x)$ has a degree $\leq (k' - 1) + (n' - k') = n' - 1$. For shortened BCH codes, the minimum distance increases in a unforseeable way, but for MDS codes the minimum distance remains constant and a statement similar to that of Theorem 8.13 can be made:

Theorem 8.14. *A shortened MDS code remains an MDS code.*

Proof. The relation $d'_{\min} \geq d_{\min}$ is apparent and the Singleton bound implies that $n - k + 1 = d_{\min} \leq d'_{\min} \leq n' - k' + 1 = n - k + 1$. ■

So, since shortened RS codes are still MDS codes, their weight distribution can be exactly calculated according to Theorem 8.3.

8.9 Problems

8.1. Name the parameters of a 2-error-correcting RS code with $q = 2^m$. How many errors can be corrected for binary interpretation?

8.2. For RS codes with $q = 2^m$ and $R = k/n \approx 1/2$, prove that $d_{\min}/n \approx 1/2$ for the distance rate.

8.3. For a 1-error-correcting RS code in the form of (8.1.3) with $q = 5$, determine the generator polynomial $g(x)$ and the parity-check polynomial $h(x)$, where $z = 3$ is a primitive element. Give a formula to directly calculate the frequency components of a codeword from the time components of the information word.

8.4. For the $(7, 3, 5)_8$ RS code in the form of (8.1.3) determine the generator polynomial $g(x)$ and the parity-check polynomial $h(x)$ as well as both forms of the parity-check matrix according to (6.3.3) and (8.1.7).

8.5. Consider a binary BCH code where the parity-check frequencies start at z^0. Prove that the designed distance d is even.

8.6. For the 1-error-correcting BCH codes over $\mathbb{F}_{16}$, prove that the degree of the generator polynomial

$$g(x) = \mathrm{LCM}\Big(f_{[z^l]}(x), f_{[z^{l+1}]}(x)\Big)$$

is at its minimum for $l = 1$. Interpret your result.

8.7. Derive the parameters of the $(63, k, d)_2$ BCH codes of Table 8.2 without explicitly calculating the generator polynomial. Use Example 7.6.

8.8. Construct a ternary BCH code in the form of (8.2.2) to correct 2 errors of length $n = 26$. Use $p(x) = x^3 + 2x + 1$ as a primitive polynomial for $\mathbb{F}_{27}$ and write the generator polynomial as a formula without an explicit calculation.

8.9. For the $(7, 5, 3)_8$ RS code with $l = 1$, determine the generator polynomial in the time and in the frequency domain. Which weight does the generator polynomial have for binary interpretation?

Consider the $(7, k, d)_2$ BCH code which is supposed to be a maximum subcode of the $(7, 5, 3)_8$ RS code. Determine k and $d_{\min}$ as well as the generator polynomial. Find a $u(x) \in \mathbb{F}_8[x]$ with $g_{\mathrm{BCH}}(x) = u(x)g_{\mathrm{RS}}(x)$. Is the BCH code still a subcode of the RS code for binary interpretation?

8.10. Consider the $(6, 2, 5)_7$ RS code with $l = 0$ over the prime field $\mathbb{F}_7$, where $z = 3$ is a primitive element. Create the transformation matrices according to (7.5.5) and (7.5.6). Determine $g(x)$ and $h(x)$. Decode the received words

$$\begin{aligned}
\boldsymbol{y}_1 &= (4, 6, 0, 1, 4, 3)\\
\boldsymbol{y}_2 &= (4, 6, 0, 1, 4, 0)\\
\boldsymbol{y}_3 &= (3, 1, 6, 1, 2, 0)\\
\boldsymbol{y}_4 &= (2, 1, 6, 1, 3, 2)\\
\boldsymbol{y}_5 &= (5, 0, 5, 0, 5, 0)\\
\boldsymbol{y}_6 &= (2, 1, 3, 5, 5, 6)\\
\boldsymbol{y}_7 &= (4, 0, 0, 1, 0, 6)
\end{aligned}$$

in the time as well as in the frequency domain. Solve the key equation with the Peterson method as well as with the Berlekamp-Massey algorithm. In all 7 cases, decoding is basically different. Interpret the occuring error scenarios.

8.11. As in Problem 8.10, consider the $(6, 2, 5)_7$ RS code with $l = 0$, however, this time with error-and-erasure correction. Decode the received words

$$\begin{aligned} \boldsymbol{y}_8 &= (?, ?, ?, 1, 4, ?) \quad \text{with} \quad I_v = \{0, 1, 2, 5\} \\ \boldsymbol{y}_9 &= (?, 6, 0, 1, 4, ?) \quad \text{with} \quad I_v = \{0, 5\} \end{aligned}$$

in the time as well as in the frequency domain.

8.12. For the general case with RS and BCH codes, prove the following representation of the error-evaluator polynomial:

$$T(x) = -\sum_{i \in I} \left(e_i \prod_{j \in I \setminus \{i\}} (1 - xz^j) \right). \tag{8.9.1}$$

In textbooks, we often find the polynomial

$$W(x) = -\sum_{i \in I} \left(e_i x z^i \prod_{j \in I \setminus \{i\}} (1 - xz^j) \right). \tag{8.9.2}$$

Prove the relation

$$W(x) = T(x) + E_0 C(x) \quad \text{mit} \quad E_0 = e(z^0). \tag{8.9.3}$$

Prove that the Forney algorithm is also valid for $W(x)$ instead of $T(x)$. For the special case of binary BCH codes, prove the relation

$$T(x) = xC'(x) - E_0 C(x) \quad , \quad W(x) = xC'(x) \tag{8.9.4}$$

and interpret this result.

8.13. Verify the equations of Problem 8.12 for the $(15, 5, 7)_2$ BCH code, discussed in Example 8.14 (however, now with $l = 0$), with the set of unknown error locations $I = \{0, 4, 10\}$ as well as for the $(6, 2, 5)_7$ RS code with $l = 0$ and $z = 3$ with the error pattern $e(x) = 3x + x^4$.