Chapter 11
Analogue BCH Codes and Direct Reduced Echelon Parity Check Matrix Construction

11.1 Introduction

Analogue error-correcting codes having real and complex number coefficients were first discussed by Marshall [2]. Later on Jack Wolf [3] introduced Discrete Fourier Transform (DFT) codes having complex number coefficients and showed that an (n, k) DFT code can often correct up to $n - k - 1$ errors using a majority voting type of decoder. The codes are first defined and it is shown that (n, k) DFT codes have coordinate coefficients having complex values. These codes have a minimum Hamming distance of $n - k + 1$ and are Maximum Distance Separable (MDS) codes. The link between the Discrete Fourier Transform and the Mattson–Solomon polynomial is discussed and it is shown that the parity check algorithm used to generate DFT codes can be applied to all BCH codes including Reed–Solomon codes simply by switching from complex number arithmetic to Galois Field arithmetic. It is shown that it is straightforward to mix together quantised and non-quantised codeword coefficients which can be useful in certain applications. Several worked examples are described including that of analogue error-correction encoding and decoding being applied to stereo audio waveforms (music).

In common with standard BCH or Reed–Solomon (RS) codes, it is shown that parity check symbols may be calculated for any $n - k$ arbitrary positions in each codeword and an efficient method is described for doing this. A proof of the validity of the method is given.

11.2 Analogue BCH Codes and DFT Codes

In a similar manner to conventional BCH codes, a codeword of an analogue (n, k) BCH code is defined as

$$c(x) = c_0 + c_1x + c_2x^2 + c_3x^3 + c_4x^4 + c_5x^5 + \cdots + c_{n-1}x^{n-1}$$

M. Tomlinson et al., *Error-Correction Coding and Decoding*,
Signals and Communication Technology,
DOI 10.1007/978-3-319-51103-0_11

where

$$c(x) = g(x)d(x)$$

$g(x)$ is the generator polynomial of the code with degree $n-k$ and $d(x)$ is any data polynomial of degree less than k. Correspondingly,

$$g(x) = g_0 + g_1x + g_2x^2 + \cdots + g_{n-k}x^{n-k}$$

and

$$d(x) = d_0 + d_1x + d_2x^2 + \cdots + d_{k-1}x^{k-1}$$

The coefficients of $c(x)$ are complex numbers from the field of complex numbers. A parity check polynomial $h(x)$ is defined, where

$$h(x) = h_0 + h_1x + h_2x^2 + h_3x^3 + h_4x^4 + h_5x^5 + \cdots + h_{n-1}x^{n-1}$$

where

$$h(x)g(x) \bmod (x^n - 1) = 0$$

and accordingly,

$$h(x)c(x) \bmod (x^n - 1) = 0$$

The generator polynomial and the parity check polynomial may be defined in terms of the Discrete Fourier Transform or equivalently by the Mattson–Solomon polynomial.

Definition 11.1 (*Definition of Mattson–Solomon polynomial*) The Mattson–Solomon polynomial of any polynomial $a(x)$ is the linear transformation of $a(x)$ to $A(z)$ and is defined by [1],

$$A(z) = \text{MS}(a(x)) = \sum_{i=0}^{n-1} a(\alpha^{-i})\, z^i \tag{11.1}$$

The inverse Mattson–Solomon polynomial or inverse Fourier transform is:

$$a(x) = \text{MS}^{-1}(A(z)) = \frac{1}{n}\sum_{i=0}^{n-1} A(\alpha^i)\, x^i \tag{11.2}$$

α is a primitive root of unity with order n and for analogue BCH codes

$$\alpha = e^{j\frac{2\pi}{n}} \tag{11.3}$$

where $j = (-1)^{\frac{1}{2}}$.

In terms of a narrow sense, primitive BCH code with a generator polynomial of $g(x)$, the coefficients of G(z) are all zero from z^0 through z^{n-k-1} and the coefficients of H(z) are all zero from z^{n-k} through z^{n-1}. Consequently, it follows that the coefficient by coefficient product of $G(z)$ and $H(z)$ represented by $\odot$

$$G(z) \odot H(z) = \sum_{j=0}^{n-1} (G_j \odot H_j)\, z^j = 0 \tag{11.4}$$

The nonzero terms of $H(z)$ extend from z^0 through to z^{n-k-1} and a valid parity check matrix in the well known form is:

$$\mathbf{H} = \begin{bmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & \alpha^1 & \alpha^2 & \dots & \alpha^{n-1} \\ 1 & \alpha^2 & \alpha^4 & \dots & \alpha^{2(n-1)} \\ 1 & \alpha^3 & \alpha^6 & \dots & \alpha^{3(n-1)} \\ \dots & \dots & \dots & \dots & \dots \\ 1 & \alpha^{n-k-1} & \alpha^{2(n-k-1)} & \dots & \alpha^{n-k-1(n-1)} \end{bmatrix}$$

It will be noticed that each row of this matrix is simply given by the inverse Mattson–Solomon polynomial of $H(z)$, where

$$\begin{aligned} H(z) &= \quad 1 \\ H(z) &= \quad z \\ H(z) &= \quad z^2 \\ H(z) &= \quad \dots \\ H(z) &= z^{n-k-1} \end{aligned} \tag{11.5}$$

Consider $H(z) = z - \alpha^i$, the inverse Mattson–Solomon polynomial produces a parity check equation defined by

$$1 - \alpha^i \;\; \alpha^1 - \alpha^i \;\; \alpha^2 - \alpha^i \; \dots 0 \dots \; \alpha^{n-1} - \alpha^i$$

Notice that this parity check equation may be derived from linear combinations of the first two rows of **H** by multiplying the first row by α^i before subtracting it from the second row of **H**. The resulting row may be conveniently represented by

$$\alpha^{a_0} \; \alpha^{a_1} \; \alpha^{a_2} \; \alpha^{a_3} \dots 0 \dots \alpha^{a_{n-2}} \; \alpha^{a_{n-1}}$$

It will be noticed that the i^{th} coordinate of the codeword is multiplied by zero, and hence the parity symbol obtained by this parity check equation is independent of the value of the i^{th} coordinate. Each one of the other coordinates is multiplied by a nonzero value. Hence any one of these $n-1$ coordinates may be solved using this parity check equation in terms of the other $n-2$ coordinates involved in the equation.

Similarly, considering $H(z) = z - \alpha^j$, the inverse Mattson–Solomon polynomial produces a parity check equation defined by

$$1-\alpha^j \;\; \alpha^1-\alpha^j \;\; \alpha^2-\alpha^j \;\ldots 0 \ldots\; \alpha^{n-1}-\alpha^j$$

and this may be conveniently represented by

$$\alpha^{b_0} \; \alpha^{b_1} \; \alpha^{b_2} \; \alpha^{b_3} \;\ldots 0 \ldots\; \alpha^{b_{n-2}} \; \alpha^{b_{n-1}}$$

Now the j^{th} coordinate is multiplied by zero and hence the parity symbol obtained by this parity check equation is independent of the value of the j^{th} coordinate.

Developing the argument, if we consider $H(z) = (z-\alpha^i)(z-\alpha^j)$, the inverse Mattson–Solomon polynomial produces a parity check equation defined by

$$\alpha^{a_0}\alpha^{b_0} \; \alpha^{a_1}\alpha^{b_1} \;\ldots 0 \ldots 0 \ldots\; \alpha^{a_{n-1}}\alpha^{b_{n-1}}$$

This parity check equation has zeros in the i^{th} and j^{th} coordinate positions and as each one of the other coordinates is multiplied by a nonzero value, any one of these $n-2$ coordinates may be solved using this parity check equation in terms of the other $n-3$ coordinates involved in the equation.

Proceeding in this way, for $H(z) = (z-\alpha^i)(z-\alpha^j)(z-\alpha^k)$, the inverse Mattson–Solomon polynomial produces a parity check equation which is independent of the i^{th}, j^{th} and k^{th} coordinates and these coordinate positions may be arbitrarily chosen. The parity check matrix is

$$\mathbf{H_m} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 \ldots & 1 \\ \alpha^{u_0} & \alpha^{u_1} & \alpha^{u_2} & \alpha^{u_3} & \alpha^{u_4} & 0 \ldots & \alpha^{u_{n-1}} \\ \alpha^{v_0} & 0 & \alpha^{v_2} & \alpha^{v_3} & \alpha^{v_4} & 0 \ldots & \alpha^{v_{n-1}} \\ \alpha^{w_0} & 0 & \alpha^{w_2} & 0 & \alpha^{w_4} & 0 \ldots & \alpha^{w_{n-1}} \end{bmatrix}$$

The point here is that this parity check matrix $\mathbf{H_m}$ has been obtained from linear combinations of the original parity check matrix $\mathbf{H}$ and all parity check equations from either $\mathbf{H}$ or $\mathbf{H_m}$ are satisfied by codewords of the code.

The parity check matrix $\mathbf{H_m}$ may be used to solve for 4 parity check symbols in 4 arbitrary coordinate positions defined by the i^{th}, j^{th} and k^{th} coordinate positions plus any one of the other coordinate positions which will be denoted as the l^{th} position. The coordinate value in the l^{th} position is solved first using the last equation. Parity symbols in the i^{th}, j^{th} and k^{th} positions are unknown but this does not matter as these are multiplied by zero. The third parity check equation is used next to solve for the parity symbol in the k^{th} position. Then, the second parity check equation is used to solve for the parity symbol in the j^{th} position and lastly, the first parity check equation is used to solve for the parity symbol in the i^{th} position. The parity check matrix values, for $s = 0$ through to $n-1$, are given by:

$$\begin{aligned}\alpha^{u_s} &= \alpha^s-\alpha^i\\ \alpha^{v_s} &= (\alpha^s-\alpha^i)(\alpha^s-\alpha^j) = \alpha^{u_s}(\alpha^s-\alpha^j)\\ \alpha^{w_s} &= (\alpha^s-\alpha^i)(\alpha^s-\alpha^j)(\alpha^s-\alpha^k) = \alpha^{v_s}(\alpha^s-\alpha^k)\end{aligned}$$

Codewords of the code may be produced by first deciding on the number of parity check symbols and their positions and then constructing the corresponding parity check matrix $\mathbf{H_m}$. From the information symbols, the parity check symbols are calculated by using each row of $\mathbf{H_m}$ starting with the last row as described above.

In the above, there are 4 parity check rows and hence 4 parity check symbols which can be in any positions of the code. Clearly, the method can be extended to any number of parity check symbols. Any length of code may be produced by simply assuming coordinates are always zero, eliminating these columns from the parity check matrix. The columns of the parity check matrix may also be permuted to any order but the resulting code will not be cyclic.

It follows that with the $n-k$ parity check equations constructed using the method above, codeword coordinates may be solved in any of $n-k$ arbitrary positions. In the construction of each parity check equation there is exactly one additional zero compared to the previously constructed parity check equation. Hence there are $n-k$ independent parity check equations in any of $n-k$ arbitrary positions.

Since these equations are all from the same code the minimum Hamming distance of the code is $n-k+1$ and the code is MDS. A system for the calculation of parity check symbols in arbitrary positions may be used for encoding or for the correction of erasures. A block diagram of such an encoder/erasures decoder is shown in Fig. 11.1.

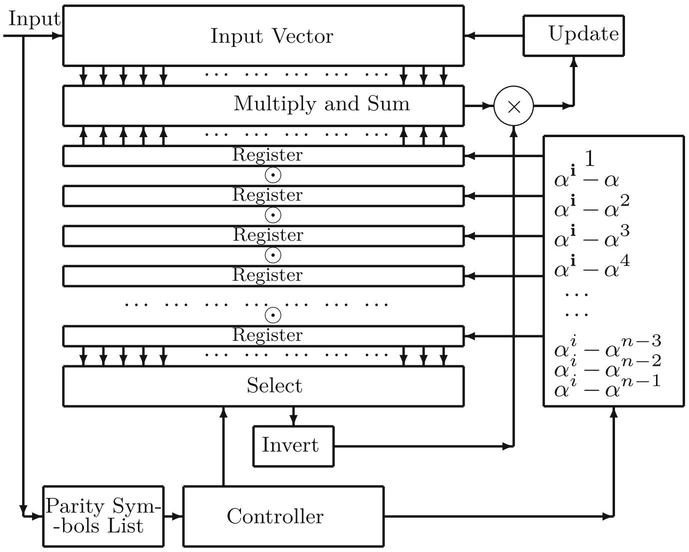

Fig. 11.1 The efficient encoder/erasures decoder realisation for BCH codes

When operating as an erasures decoder in Fig. 11.1, the List of Parity Symbols is replaced with a list of the erasures positions.

11.3 Error-Correction of Bandlimited Data

In many cases, the sampled data to be encoded with the analogue BCH code is already bandlimited or near bandlimited in which case, the higher frequency coefficients of the Mattson–Solomon polynomial, $D(z)$ of the data polynomial $d(x)$, consisting of successive PAM samples, will be zero or near zero. An important point here is that there is no need to add additional redundancy with additional parity check samples. In a sense the data, as PAM samples, already contains the parity check samples. Commonly, it is only necessary to modify a small number of samples to turn the sampled data into codewords of the analogue BCH code as illustrated in the example below. The broad sense BCH codes are used with the following parity check matrix, with $\alpha = e^{-j\frac{2\pi}{n}}$.

$$\mathbf{H_f} = \begin{bmatrix} 1 & \alpha^{\beta} & \alpha^{2\beta} & \dots & \alpha^{(n-1)\beta} \\ 1 & \alpha^{\beta+1} & \alpha^{2(\beta+1)} & \dots & \alpha^{((n-1)(\beta+1)} \\ 1 & \alpha^{\beta+2} & \alpha^{2(\beta+2)} & \dots & \alpha^{(n-1)(\beta+2)} \\ 1 & \alpha^{\beta+3} & \alpha^{2(\beta+3)} & \dots & \alpha^{(n-1)(\beta+3)} \\ \dots & \dots & \dots & \dots & \dots \\ 1 & -1 & 1 & \dots & 1 \end{bmatrix} \tag{11.6}$$

Using this parity check matrix will ensure that the highest $n-k$ Fourier coefficients will be zero. Several alternative procedures may be used. $n-k$ samples in each sequence of n samples may be designated as parity symbols and solved using this parity check matrix following the procedure above for constructing the reduced echelon matrix so that the values of the designated parity samples may be calculated. An alternative, more complicated procedure, is for each constructed codeword, to allow the $n-k$ parity samples to be in any of the $\frac{n!}{k!(n-k)!}$ combinations of positions and choose the combination which produces the minimum mean squared differences compared to the original $n-k$ complex samples.

11.4 Analogue BCH Codes Based on Arbitrary Field Elements

It is not necessary that the parity check matrix be based on increasing powers of α with parity check equations corresponding to the forcing of Fourier coefficients to be zero. An arbitrary ordering of complex field elements corresponding to permuted powers of α may be used. With $\alpha = e^{-j\frac{2\pi}{N}}$ where $N \geq n$, consider the parity check matrix

$$\mathbf{H_a} = \begin{bmatrix} \alpha_0 & \alpha_0 & \alpha_0 & \alpha_0 & \ldots & \alpha_0 \\ \alpha_0 & \alpha_1 & \alpha_2 & \alpha_3 & \ldots & \alpha_{n-1} \\ \alpha_0 & \alpha_1^2 & \alpha_2^2 & \alpha_3^2 & \ldots & \alpha_{n-1}^2 \\ \alpha_0 & \alpha_1^3 & \alpha_2^3 & \alpha_3^3 & \ldots & \alpha_{n-1}^3 \\ \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\ \alpha_0 & \alpha_1^{n-k-1} & \alpha_2^{n-k-1} & \alpha_3^{n-k-1} & \ldots & \alpha_{n-1}^{n-k-1} \end{bmatrix}$$

The $\{\alpha_0, \alpha_1, \alpha_2, \alpha_3, \ldots, \alpha_{n-1}\}$ complex number field elements are all distinct and arbitrary powers of α. Any combination of any $n-k$ columns, or less, of this parity check matrix are independent because the matrix transpose is a Vandermonde matrix [1]. Consequently, the code is a $(n, k, n-k+1)$ MDS code.

Following the same procedure as outlined above to produce directly a reduced echelon parity check matrix $\mathbf{H_b}$ with zeros in arbitrary columns, for example in three columns headed by, α_a, α_b and α_c.

$$\mathbf{H_b} = \begin{bmatrix} \alpha_0 & \alpha_0 & \alpha_0 & \alpha_0 & \ldots & \alpha_0 \\ (\alpha_0-\alpha_a) & 0 & (\alpha_2-\alpha_a) & (\alpha_3-\alpha_a) & \ldots & (\alpha_{n-1}-\alpha_a) \\ (\alpha_0-\alpha_a)(\alpha_0-\alpha_b) & 0 & 0 & (\alpha_3-\alpha_a)(\alpha_3-\alpha_b) & \ldots & (\alpha_{n-1}-\alpha_a)(\alpha_{n-1}-\alpha_b) \\ (\alpha_0-\alpha_a)(\alpha_0-\alpha_b)(\alpha_0-\alpha_c) & 0 & 0 & (\alpha_3-\alpha_a)(\alpha_3-\alpha_b)(\alpha_3-\alpha_c) & \ldots & 0 \end{bmatrix}$$

The parity check equation corresponding to the fourth row of this parity check matrix is

$$\sum_{i=0}^{n-1} (\alpha_i-\alpha_a)(\alpha_i-\alpha_b)(\alpha_i-\alpha_c)c_i = 0 \tag{11.7}$$

where the analogue BCH codeword consists of n complex numbers

$$\{c_0, c_1, c_2, c_3, \ldots, c_{n-1}\}$$

k of these complex numbers may be arbitrary, determined by the information source and $n-k$ complex numbers are calculated from the parity check equations:

Defining

$$(\alpha_i-\alpha_a)(\alpha_i-\alpha_b)(\alpha_i-\alpha_c) = \alpha_i^3 + \beta_2\alpha_i^2 + \beta_1\alpha_i^1 + \beta_0$$

Parity check Eq. (11.7) becomes

$$\sum_{i=0}^{n-1} \alpha_i^3 c_i + \beta_2 \sum_{i=0}^{n-1} \alpha_i^2 c_i + \beta_1 \sum_{i=0}^{n-1} \alpha_i c_i + \beta_0 \sum_{i=0}^{n-1} c_i = 0 \tag{11.8}$$

This codeword is from the same code as defined by the parity check matrix $\mathbf{H_a}$ because using parity check matrix $\mathbf{H_a}$, codewords satisfy the equations

$$\sum_{i=0}^{n-1} \alpha_i^3 c_i = 0 \quad \sum_{i=0}^{n-1} \alpha_i^2 c_i = 0 \quad \sum_{i=0}^{n-1} \alpha_i c_i = 0 \quad \sum_{i=0}^{n-1} c_i = 0$$

and consequently the codewords defined by $\mathbf{H_a}$ satisfy (11.8) as

$$0 + \beta_2 0 + \beta_1 0 + \beta_0 0 = 0$$

It is apparent that the reduced echelon matrix $\mathbf{H_b}$ consists of linear combinations of parity check matrix $\mathbf{H_a}$ and either matrix may be used to produce the same MDS, analogue BCH code.

11.5 Examples

11.5.1 Example of Simple $(5, 3, 3)$ *Analogue Code*

This simple code is the extended analogue BCH code having complex sample values with $\alpha = e^{\frac{j2\pi}{4}}$ and uses the parity check matrix:

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

This parity check matrix is used to encode 3 complex data values in the last 3 positions, viz $(0.11 + 0.98j, \quad -0.22 - 0.88j, \quad 0.33 + 0.78j)$. This produces the codeword:

$$(-0.2 - 0.22j, \quad -0.02 - 0.66j, \quad 0.11 + 0.98j, \quad -0.22 - 0.88j, \quad 0.33 + 0.78j)$$

Suppose the received vector has the last digit in error

$$(-0.2 - 0.22j, \quad -0.02 - 0.66j, \quad 0.11 + 0.98j, \quad -0.22 - 0.88j, \quad 0.4 + 0.9j)$$

Applying the first parity check equation produces $0.07 + 0.12j$. This result tells us that there is an error of $0.07 + 0.12j$ in one of the received coordinates. Applying the second parity check equation produces $0.12 - 0.07j$. Since this is the error multiplied by $-j$, this tells us that the error is in the last coordinate. Subtracting the error from the last coordinate of the received vector produces $(0.4 + 0.9j) - (0.07 + 0.12j) = 0.33 + 0.78j$ and the error has been corrected.

11.5.2 Example of Erasures Correction Using **(15, 10, 4)** *Binary BCH code*

This is an example demonstrating that the erasures decoder shown in Fig. 11.1 may be used to correct erasures in a binary BCH code as well as being able to correct erasures using an analogue BCH code.

The code is a binary BCH code of length $n = 15$, with binary codewords generated by the generator polynomial $g(x) = (1+x^3+x^4)(1+x)$. The Galois field is GF(2^4) generated by the primitive root α, which is a root of the primitive polynomial $1+x+x^4$ so that $1+\alpha+\alpha^4 = 0$, and the Galois field consists of the following table of 15 field elements, plus the element, 0.

One example of a codeword from the code is

$$c(x) = x + x^3 + x^4 + x^6 + x^8 + x^9 + x^{10} + x^{11} \tag{11.9}$$

and consider that in a communication system, the codeword is received with erasures in positions in $\lambda_0 = 5$, $\lambda_1 = 0$ and $\lambda_2 = 8$, so that the received codeword is

$$\hat{c}(x) = \hat{c}_0 + x + x^3 + x^4 + \hat{c}_5 x^5 + x^6 + \hat{c}_8 x^8 + x^9 + x^{10} + x^{11} \tag{11.10}$$

To find the party check equations to solve for the erasures, referring to Fig. 11.1, the first parity check equation, $h_0(x)$, the all 1's vector is stored in the Register. The second parity check equation $h_1(x)$ has zero for the coefficient of $x^{n-\lambda_0=n-5}$ and is given by

$$h_1(x) = \sum_{j=0}^{n-1} (\alpha^j - \alpha^{10})\, x^j \tag{11.11}$$

Note that $h_i(x).\hat{c}(x) = 0$ and these polynomials are derived with the intention that the coefficient of x^0 will be evaluated. Referring to Fig. 11.1, $h_1(x)$ is stored in the corresponding Register. After substitution using Table 11.1, it is found that

$$\begin{aligned} h_1(x) = {} & \alpha^5 + \alpha^8 x + \alpha^4 x^2 + \alpha^{12} x^3 + \alpha^2 x^4 + x^5 \\ & + \alpha^7 x^6 + \alpha^6 x^7 + \alpha x^8 + \alpha^{13} x^9 + \alpha^{14} x^{11} \\ & + \alpha^3 x^{12} + \alpha^9 x^{13} + \alpha^{11} x^{14} \end{aligned} \tag{11.12}$$

Notice that although the codeword is binary, the coefficients of this equation are from the full extension field of GF(16). The third parity check equation $h_2(x)$ has zero in position $n - \lambda_1 = n - 0$ and is given by

$$h_2(x) = \sum_{j=0}^{n-1} (\alpha^j - 1)\, x^j \tag{11.13}$$

after evaluation

$$\begin{aligned}h_2(x) &= \alpha^4 x + \alpha^8 x^2 + \alpha^{14} x^3 + \alpha x^4 + \alpha^{10} x^5\\ &+ \alpha^{13} x^6 + \alpha^9 x^7 + \alpha^2 x^8 + \alpha^7 x^9 + \alpha^5 x^{10}\\ &+ \alpha^{12} x^{11} + \alpha^{11} x^{12} + \alpha^6 x^{13} + \alpha^3 x^{14}\end{aligned} \tag{11.14}$$

Referring to Fig. 11.1, this polynomial is stored in the corresponding Register.

The parity check equation which gives the solution for coefficient $\hat{c}_8$ is $h_3(x) = h_0(x) \odot h_1(x) \odot h_2(x)$. Multiplying each of the corresponding coefficients together of the polynomials $h_0(x)$, $h_1(x)$ and $h_2(x)$ produces

$$\begin{aligned}h_3(x) &= \alpha^{12} x + \alpha^{12} x^2 + \alpha^{11} x^3 + \alpha^3 x^4 + \alpha^{10} x^5\\ &+ \alpha^5 x^6 + x^7 + \alpha^{10} x^8 + \alpha^5 x^9 + \alpha^{11} x^{11}\\ &+ \alpha^{14} x^{12} + x^{13} + \alpha^{14} x^{14}\end{aligned} \tag{11.15}$$

Referring to Fig. 11.1, $h_3(x)$ will be input to Multiply and Sum. It should be noted that the parity check equation $h_3(x)$ has non-binary coefficients, even though the codeword is binary and the solution to the parity check equation has to be binary.

Evaluating the coefficient of x^0 of $h_3(x)\hat{c}(x)$ gives $\alpha^{14} + \alpha^{14} + \alpha^{11} + \alpha^5 + \hat{c}_8 + \alpha^5 + \alpha^{10} + \alpha^3 = 0$, which simplifies to $\alpha^{11} + \hat{c}_8 + \alpha^{10} + \alpha^3 = 0$. Using Table 11.1 gives

$$(\alpha + \alpha^2 + \alpha^3) + \hat{c}_8 + (1 + \alpha + \alpha^2) + \alpha^3 = 0$$

Table 11.1 All 15 Nonzero Galois Field elements of GF(16)

Symbol $\alpha^{\mathbf{i}}$	modulo $1 + \alpha + \alpha^4$			
$\alpha^0 =$	1			
$\alpha^1 =$		α^1		
$\alpha^2 =$			α^2	
$\alpha^3 =$				α^3
$\alpha^4 =$	1+	α		
$\alpha^5 =$		$\alpha+$	α^2	
$\alpha^6 =$			α^2+	α^3
$\alpha^7 =$	1+	$\alpha+$		α^3
$\alpha^8 =$	1+		α^2	
$\alpha^9 =$		$\alpha+$		α^3
$\alpha^{10} =$	1+	$\alpha+$	α^2	
$\alpha^{11} =$		$\alpha+$	α^2+	α^3
$\alpha^{12} =$	1+	$\alpha+$	α^2+	α^3
$\alpha^{13} =$	1+		α^2+	α^3
$\alpha^{14} =$	1+			α^3

and $\hat{c}_8 = 1$. Referring to Fig. 11.1, Select produces from $h_3(x)$ the value of the coefficient of x^7 which is 1 and when inverted this is also equal to 1. The output of the Multiply and Add is 1, producing a product of 1, which is used by Update to update $\hat{c}_8 = 1$ in the Input Vector $\hat{c}(x)$.

The parity check equation $h_2(x)$ gives the solution for coefficient $\hat{c}_0$. Evaluating the coefficient of x^0 of $h_2(x)\hat{c}(x)$ gives

$$\begin{aligned} 0 = &\alpha^3 + \alpha^{11} + \alpha^{12} + \hat{c}_5\alpha^5 + \alpha^7 \\ &+ \alpha^9 + \alpha^{13} + \alpha^{10} + \alpha \end{aligned}$$

Substituting using Table 11.1 gives $\hat{c}_5\alpha^5 = 0$ and $\hat{c}_5 = 0$.

Lastly the parity check equation $h_0(x)$ gives the solution for coefficient $\hat{c}_0$. Evaluating the coefficient of x^0 of $h_0(x)\hat{c}(x)$ gives

$$0 = \hat{c}_0 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 \tag{11.16}$$

and it is found that $\hat{c}_0 = 0$, and the updated $\hat{c}(x)$ with all three erasures solved is

$$\hat{c}(x) = x + x^3 + x^4 + x^6 + x^8 + x^9 + x^{10} + x^{11} \tag{11.17}$$

equal to the original codeword.

11.5.3 *Example of (128, 112, 17) Analogue BCH Code and Error-Correction of Audio Data (Music) Subjected to Impulsive Noise*

In this example, a stereo music file sampled at 44.1 kHz in complex Pulse Amplitude Modulation (PAM) format is split into sequences of 128 complex samples and encoded using an analogue (128, 112, 17) BCH code with $\alpha = e^{\frac{j2\pi}{128}}$, and reassembled into a single PAM stream. A short section of the stereo left channel waveform before encoding is shown plotted in Fig. 11.2.

The encoding parity check matrix is the $\mathbf{H_f}$ matrix for bandlimited signals given above in matrix (11.6). There are 16 parity symbols and to make these obvious they are located at the beginning of each codeword. The same section of the stereo left channel waveform as before but after encoding is shown plotted in Fig. 11.3. The parity symbols are obvious as the newly introduced spikes in the waveform.

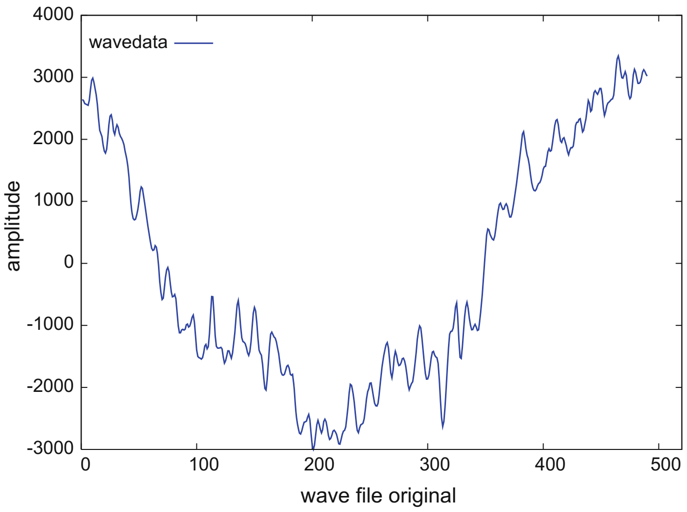

Fig. 11.2 Section of music waveform prior to encoding

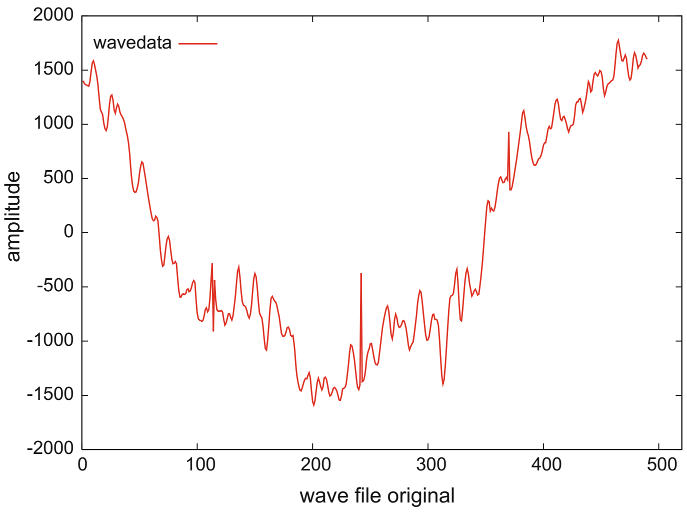

Fig. 11.3 Section of music waveform after encoding

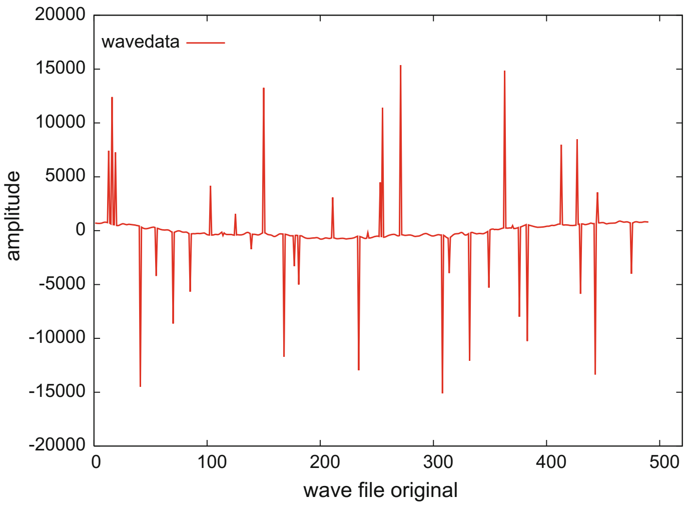

Fig. 11.4 Section of music waveform after encoding and subjected to impulse noise

The parity symbols may be calculated for any combination of 16 coordinate positions and in a more complicated encoding arrangement the positions could be selected as those that produce the minimum mean square error. However, the frequency components affected extend from 19.47 to 22.1 kHz (these components are equal to zero after encoding) and are beyond the hearing range of most people.

The encoded music waveform is subjected to randomly distributed impulse noise with a uniformly distributed amplitude in the range ±16000. The result is shown plotted in Fig. 11.4 for the same section of the waveform as before, although this is not obvious in the plot.

The decoder strategy used is that in each received codeword the 16 received PAM samples with the greatest magnitudes exceeding a dynamic threshold or with largest change relative to neighbouring samples are erased. The erasures are then solved using the parity check equations as outlined above. In several cases, correctly received PAM samples are erased, but this does not matter provided the 112 non-erased samples in each received codeword are correct. The decoded music waveform is shown in Fig. 11.5, and is apparent that waveform after decoding is the same as the encoded waveform and the impulse noise errors have been corrected.

Usually, impulse noise effects are handled by noise suppressors which produce short, zero-valued waveform sections. These audible gaps are irritating to the listener. By using analogue BCH, error-correcting codes, there are no waveform gaps following decoding.

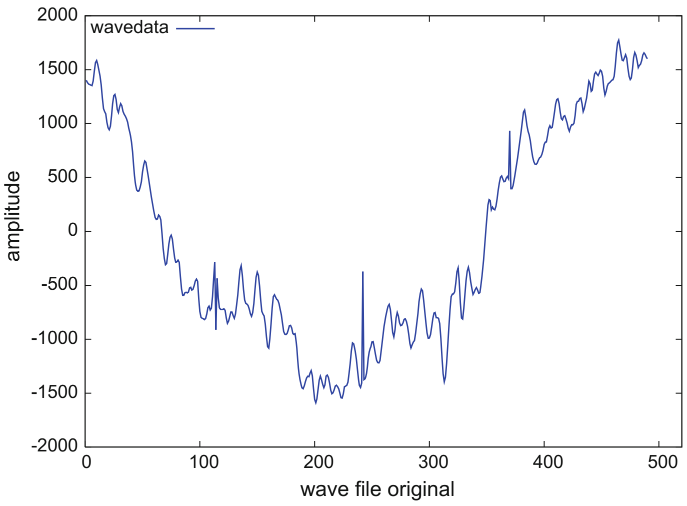

Fig. 11.5 Section of music waveform after decoding

11.6 Conclusions and Future Research

It has been demonstrated that for analogue $(n, k, n-k+1)$ BCH codes, parity check symbols having complex values may be calculated for any $n-k$ arbitrary positions in each codeword and an efficient method of calculating erased symbols for any BCH code including binary codes has been presented. Bandlimited data naturally occurs in many sources of information. In effect the source data has already been encoded with an analogue BCH code. In practice the parity check equations of the BCH code will only approximately equal zero for the PAM samples of the bandlimited source. There is scope for determining those samples which require the minimum of changes in order to satisfy the parity check equations. Similarly in decoding codewords corrupted by a noisy channel there is the opportunity to use the statistics of the noise source to design a maximum likelihood decoder for analogue BCH codes. It appears likely that the extended Dorsch decoder described in Chap. 15 may be adapted for analogue BCH codes.

There are many ad hoc noise suppression algorithms used on analogue video and audio waveforms which cause artefacts in the signal processed outputs. There appears to be an opportunity to improve on these by using analogue BCH coding since the output of the decoder is always a codeword. For high quality systems this will predominantly be the transmitted codeword and therefore the decoder output will be free of artefacts.

Whilst most communications these days is digitally based, analogue communications is usually far more bandwidth efficient, particularly in wireless applications. By using analogue BCH codes, analogue communications may be attractive once more, particularly for niche applications.

Steganography is another area in which analogue BCH codes may be utilised. Errors in parity check equations may be used to communicate data in a side channel. By virtue of the parity check equations these errors may be distributed over multiple PAM samples or pixels. Secrecy may be assured by using a combination of secret permutations of the parity check matrix columns and a secret linear matrix transformation so that the parity check equations are unknown by anyone other than the originator.

11.7 Summary

Many information sources are naturally analogue and must be digitised if they are to be transmitted digitally. The process of digitisation introduces quantisation errors and increases the bandwidth required. The use of analogue error-correcting codes eliminates the need for digitisation. It been shown that analogue BCH codes may be constructed in the same way as finite field BCH codes, including Reed–Solomon codes. The difference is that the field of complex numbers is used instead of a prime field or prime power field. It has been shown how the Mattson–Solomon polynomial or equivalently the Discrete Fourier transform may be used as the basis for the construction of analogue codes. It has also been shown that a permuted parity check matrix produces an equivalent code using a primitive root of unity to construct the code as in discrete BCH codes.

A new algorithm was presented which uses symbolwise multiplication of rows of a parity check matrix to produce directly the parity check matrix in reduced echelon form. The algorithm may be used for constructing reduced echelon parity check matrices for standard BCH and RS codes as well as analogue BCH codes. Gaussian elimination or other means of solving parallel, simultaneous equations are completely avoided by the method. It was also proven that analogue BCH codes are Maximum Distance Separable (MDS) codes. Examples have been presented of using the analogue BCH codes in providing error-correction for analogue, band-limited data including the correction of impulse noise errors in BCH encoded, analogue stereo music waveforms. It is shown that since the data is bandlimited it is already redundant and the parity check symbols replace existing values so that there is no need for bandwidth expansion as in traditional error-correcting codes. Future research areas have been outlined including an analogue, maximum likelihood, error-correcting decoder based on the extended Dorsch decoder of Chap. 15. Steganography is another future application area for analogue BCH codes.

References

1. MacWilliams, F.J., Sloane, N.J.A.: The Theory of Error-Correcting Codes. North-Holland, Amsterdam (1977)
2. Marshall, J.T.: Coding of real-number sequences for error correction: a digital signal processing problem. IEEE J. Sel. Areas Commun. **2**(2), 381–392 (1984)
3. Wolf, J.: Redundancy, the discrete fourier transform, and impulse noise cancellation. IEEE Trans. Commun. **31**(3), 458–461 (1983)