# Error Correcting Codes with Mathematica

Igor Gashkov

Karlstad University, Department of Engineering Sciences, Physics and Mathematics 65188 Karlstad Sweden

Igor.Gachkov@kau.se

**Abstract.** The author (with Kenneth Hulth) got the idea to develop a non-standard, methodical-oriented course where hands-on sessions could add substantial understanding in the introduction of mentioned mathematical concepts. The package in MATHEMATICA in the field "Coding Theory" was developed for course " Error-Correcting codes with MATHEMATICA ", for students on advanced undergraduate level.

## 1 Introduction

Applications of the theories of Error-Correcting Codes have increased tremendously in recent years. Thus it is hardly possible today to imagine engineers working with data transmission and related fields without basic knowledge of coding/decoding of information. The possibilities of quantifying information with electronic equipment are developing rapidly, supplying the specialists working in communication theory with more sophisticated methods for circuit realization of concrete algorithms in Coding Theory. During preceding years courses in Coding Theory have been considered only for students on postgraduate level. This is due to the complexity of the mathematical methods used in most codes, such as results from abstract algebra including linear spaces over Galois Fields. With the introduction of computers and computer algebra the methods can be fairly well illustrated. The author has developed a course, 'Coding Theory in MATHEMATICA', using the wide range of capabilities of MATHEMATICA. The course was given at Jönköping University and Karlstad University, Sweden, on undergraduate level with a minimum of prerequisites. The hands on sessions were based on a package of application programs/algorithms, developed to illustrate the mathematical constructions, used in coding theory to encode and decode information. We will present some of our hands-on materials, which are used to construct Block Codes with means of algebraic methods. We first present the basic concepts of coding theory, such as, hamming distance, generator and parity check matrices, binary linear codes and group codes. We will then use some basic results from matrix algebra and linear spaces to construct code words from the information we want to send. Due to noise in the channel of transmission the received words may differ from the code words which were sent, and the very aim of coding theory is to detect and correct possible errors. In the linear codes (the code words having group structure) the encoding is accomplished by multiplying the

P.M.A. Sloot et al. (Eds.): ICCS 2003, LNCS 2658, pp. 1042–1051, 2003.

information word (seen as a vector) with a generator matrix, whereas the decoding process starts with multiplication of the received word with the parity check matrix. Within the cyclic codes it is preferable to work with generator- and parity check polynomials instead of matrices, and the code words here form a polynomial ring. With algebraic extensions of the field $Z_p$ by irreducible (over $Z_p$) polynomials, the final step is taken into the BCH-codes, which have rich algebraic, structure. The application programs in the package support the learning processes by illustrating the algorithms in the mathematical constructions. The rich variety of manipulations in the algebraic structures, the possibility to vary the parameters in specific cases and the simplicity to construct concrete codes with MATHEMATICA should strengthen the understanding of the mathematical ideas used in coding theory.

## 2 Introduction to Coding Theory

As an introduction to the theory of Error-Correcting Codes we study the well-known Venn-Diagram with the three overlapping circles (Fig. 1)

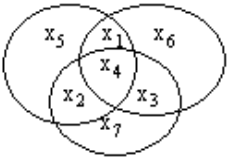



**Fig. 1.** Venn-Diagram with the three overlapping circles

We place 7 pieces $x_i$, where $x_i$ is marked with the number (1) on one side and (0) on the other, on the respective i:th area of the figure. This could be performed in such a way, that the sum of the numbers in each circle is even, i.e. for each of the circles applies $\Sigma x_i = 0$ (mod 2). We will see that there are a total of 16 such possibilities, and if we write these $x = x_1x_2x_3...x_7$ we have created a code with 16 codewords. If we transmit such a codeword through a communication channel, it may happen, due to noise in the channel, that one (or several) of the $x_i$ changes from 0 to 1 or vise versa. In our figure the error is easily found by checking the sum (mod 2) of the numbers in each of the circles. If we change $x_i$ for exactly one i the conditions $\Sigma x_i = 0$ (mod 2) will be fulfilled. We thus see, that there do exist possibilities to correct errors in the transmission of information. We now turn to the figure again. The condition $\sum x_i = 0$ for each of the tree circles immediately gives (if we perform the operations modulo 2, i.e. within the field $Z_2$)

$$\begin{cases} x_1 + x_2 + x_4 + x_5 = 0 \\ x_1 + x_3 + x_4 + x_6 = 0 \\ x_2 + x_3 + x_4 + x_7 = 0 \end{cases} \Leftrightarrow \begin{cases} x_1 + x_2 + x_4 = x_5 \\ x_1 + x_3 + x_4 = x_6 \\ x_2 + x_3 + x_4 = x_7 \end{cases} \quad (1)$$

It follows that in order to place the 7 pieces correctly, we simply place $x_1$ ... $x_4$ freely and then compute $x_5$, $x_6$ and $x_7$ in accordance with the equations. This means, that out of a total of $2^7$ combinations of $x_1$.... $x_7$ there will be $2^4$ combinations satisfying our condition, i.e. from $2^7$ *words* there will be $2^4$ *codewords* . In order to find the codewords we could also proceed as follows: The word v= ($v_1 v_2 ... v_7$ ) would be a codeword iff (1) is satisfied, which in matrix form means iff v satisfy

$$\begin{pmatrix} 1 & 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 \end{pmatrix} v^{tr} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \Leftrightarrow H\, v^{tr} = \mathbf{0} \quad (2)$$

In order to check whether a word w received from the communication channel is a code word, we only compute H $w^{tr} = s^{tr}$ , where s is called the syndrome, if $s^{tr} = \mathbf{0}$, then v is a code word. If exactly one error occurs in the transmission, then exactly one $x_i$ would be false, and we write (as vectors) w = v + $e_i$ , where v is a code word and $e_i$ has a single nonzero bit, in the position i .

We compute H $v^{tr}$ = H ($v^{tr}$ +$e_i^{tr}$) = (linearity!) = H $v^{tr}$ + H $e_i^{tr}$ = 0 + H $e_i^{tr}$ = the i:th column in H. One of these columns has to be in accordance with $s^{tr}$ and so we correct the corresponding $v_i$. It can easily be proved that we would always find the error, provided we have exactly one false bit.

In general, if H is a binary matrix, the linear code with the *parity check matrix* H consists of all vectors v, satisfying H $v^{tr} = \mathbf{0}$. Usually H is an (n-k) x n matrix H = [A $I_{n-k}$], with $I_{n-k}$ the unit matrix (n-k) x (n-k) . With the information word u = $u_1$... $u_k$ we write the codeword v = $v_1$...$v_k v_{k+1}$...$v_n$ , where $v_i = u_i$ $1 \le i \le k$ and where $v_{k+1}$.... $v_n$ are the check symbols. We then have H $v^{tr} = \mathbf{0} \Leftrightarrow \mathbf{x} = \mathbf{u}G$, where G = [$I_k$ $A^{tr}$]. G is called the generator matrix of the code. We have in our introductory example ended demonstrated the *Hamming Code* K[7,4,3], where the parameter 7 indicates the length and 4 the dimension (=number of information bits) of the code. The parameter 3 gives the *hamming distance* d of the code: the hamming distance d (x, y) between the words x and y equals the number of positions i, where $x_i \neq y_i$ ; d = min d (x, y) = minimum distance between any two codewords. d is easily found to be equal to the minimum *hamming weight* wt (v) of any codeword v, where wt(v) is the number of nonzero $v_i$.

The Hamming Code K [n, k, d] is characterized by (let m be the number of check bits, $m \geq 2$) $n = 2^m-1$, $k = 2^m-1-m$, d=3 and is a perfect single-error-correcting code, meaning that the code has exactly the information needed for correcting one error.

## 3 The Package " Coding Theory"

The package "Coding Theory" is a file written in MATHEMATICA and will be read into MATHEMATICA with the commands.

```
In[1] :=<<CodingTheory.m
```

The package consists of two parts: one part with illustrative explanations, and one for scientific purposes. The illustrative part (commands starting with Show...) is considered to visualize the theoretical aspects of encoding / decoding, construct shift-register circuits etc. The command `?Show*` gives the following list of commands

```
In[2] := ?Show*

Out[2]=Show
ShowHammingCode
ShowBinaryGaloisField
ShowMeggittDecoder
ShowErrorTrappingDecoderBCHCode
ShowSystematicEncoderCyclicCode
…
```

The complete information about a command is received by using the command ? *Name*.

```
In[3] : = ? ShowHammingCode

Out[3]= ShowHammingCode[m,inf] shows the method of
encoding an information  word inf  into a hamming code
word of length 2^m - 1 .
```

We now use MATHEMATICA to construct Hamming codes. We first chose m = 3 which gives $n=2^3 - 1 =7$, i.e. the code K [7,4,3].

```
In[4]:=  inf={1,1,1,0}; ShowHammingCode[3,inf]

Out[4]=
PARITY CHECK                GENERATOR
```

$$\text{MATRIX}=\begin{pmatrix}0&0&0&1&1&1&1\\0&1&1&0&0&1&1\\1&0&1&0&1&0&1\end{pmatrix}\quad\text{MATRIX}=\begin{pmatrix}1&1&0&1&0&0&1\\0&1&0&1&0&1&0\\1&0&0&1&1&0&0\\1&1&1&0&0&0&0\end{pmatrix}$$

```
inf = {1,1,1,0}
v = inf * G  HAMMING CODE VECTOR v = {0,0,0,1,1,1,1}
```

We proceed by sending the code word v and let an error appear in the 5:th position. We thus decode the received word w = (0001011).

```
In[5]:= v={0,0,0,1,1,1,1};e={0,0,0,0,1,0,0};
w=Mod[v + e,2]; ShowDecHammingCode[3,w]

Out[5]=
```

$$\text{PARITY CHECK MATRIX}=\begin{pmatrix}0&0&0&1&1&1&1\\0&1&1&0&0&1&1\\1&0&1&0&1&0&1\end{pmatrix}\quad S=\begin{pmatrix}1\\0\\1\end{pmatrix}$$

```
POSITION OF ERROR = 5 RECEIVED WORD = {0,0,0,1,0,1,1}
DECODED WORD =  {0,0,0,1,1,1,1} inf ={1,1,1,0}
```

We now compute the parameters of the code.

```
In[6]:=
H = HammingMatrix[3];Length[H[[1]]] ;DimensionCode[H];
DistanceCode[H]
Out[6]=
7
4
3
```

## 4 Fundamental from Algebra

We need for the construction of the linear block-codes some prerequisites from algebra, such as structures of groups, cosets, rings and finite fields. The linear block codes are seen as subspaces of linear spaces over a finite field. Of fundamental importance are the following algebraic concepts:

- Irreducible polynomial : f (x) is irreducible over the field F, if f(x) cannot be factored as a product of two polynomials of degrees smaller than that of f(x).
- Primitive element: $\mapsto$ is a primitive element in the field F provided that every nonzero element in F is equal to some power of $\mapsto$.
- Minimal polynomial: If $\updownarrow$ is an element of some extension of the field F, the minimal polynomial of $\updownarrow$ (with respect to F) is the lowest-degree monic polynomial M (x) over F with M ($\updownarrow$) = 0. The minimal polynomial M(x) of $\updownarrow$ $\in$ GF($p^m$) can be computed as
- M(x)= $(x-\beta)(x-\beta^{p})(x-\beta^{p^2})\ldots(x-\beta^{p^i})$, where $\beta=\beta^{p^{i+1}}$
- The Galois field GF($p^m$) is defined as an algebraic extension of the field $Z_p$ (p prime number) by an irreducible polynomial of degree m. If g(x) is a polynomial of degree m and irreducible over $Z_p$ we construct GF($p^m$) = $Z_p$ / g(x) with $p^m$ elements $\sum_{i=0}^{m-1} a_i\alpha^i$, $a_i \in Z_p$. The element $\alpha$ satisfies $g(\alpha)=0$.

```
In[7]:=
BIP = BinaryIrreduciblePolynomials[3,x]
Out[7] = {1 + x^2 + x^3 ,  1 + x + x^3 }

In[8]:=irr=BIP[[2]];ShowBinaryGaloisField[irr, x, b, b]
```

GF(8) is received by extending the field $Z_2$ by the irreducible polynomial $g(x) = 1 + x + x^3$ and observe that b is a primitive element in this extension field.

```
Out[8] =
```

| Log | Vector | Pr. el. | Polynomial | Min. polynomial |
|---|---|---|---|---|
| $-\infty$ | (0, 0, 0) | 0 | 0 | x |
| 0 | (1, 0, 0) | 1 | 1 | $1 + x$ |
| 1 | (0, 1, 0) | b | b | $1 + x + x^3$ |
| 2 | (0, 0, 1) | $b^2$ | $b^2$ | $1 + x + x^3$ |
| 3 | (1, 1, 0) | $b^3$ | $1 + b$ | $1 + x^2 + x^3$ |
| 4 | (0, 1, 1) | $b^4$ | $b + b^2$ | $1 + x + x^3$ |
| 5 | (1, 1, 1) | $b^5$ | $1 + b + b^2$ | $1 + x^2 + x^3$ |
| 6 | (1, 0, 1) | $b^6$ | $1 + b^2$ | $1 + x^2 + x^3$ |

We illustrate the multiplication of the elements $1 + b + b^2$ and $1 + b^2$ in GF(8):

```
In[9] := PolynomialMod[PolynomialMod[(1+b+b^2)*(1+b^2),
irrpol /. x - > b], 2]

Out[9] =       b + b^2
```

## 5 Cyclic Codes

The linear code K is called cyclic, if $v = (v_0, v_1, ..., v_{n-1}) \in$ K $\Leftrightarrow$ $w = (v_{n-1}, v_0, ..., v_{n-2}) \in$ K. If we then represent the vector $v = (v_0, v_1, ..., v_{n-1})$ with the polynomial $v(x) = \sum_{i=0}^{n-1} v_i x^i$, the cyclical shift of $v$ to $w$ corresponds to a multiplication of $v(x)$ with $x$. If $g(x)$ is a polynomial with lowest degree in K, it can be shown, that $g(x)$ is uniquely determined and generates K and

is thus called the *generator polynomial* of K $g(x)$ is a divisor of $x^n - 1$ and we have the following connection between generator matrix and generator polynomial:

$$G = \begin{pmatrix} g(x) \\ xg(x) \\ . \\ x^{k-1} g(x) \end{pmatrix} \tag{3}$$

In a cyclical code we now construct the code word $v(x)$ from the information word u in following steps:

- $u = (u_0,...,u_k) \Rightarrow u(x) = u_k x^{n-1} + u_{k-1} x^{n-2} + ... + u_0 x^{n-k}$
- construct $r(x)$ as the remainder in $u(x) = q(x) \cdot g(x) + r(x)$
- let the code word be $v(x) = u(x) \cdot r(x)$

These algebraic operations could be performed by the *shift register circuit* (Fig. 2):

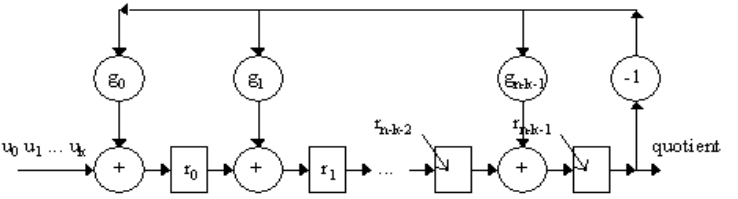



**Fig. 2.** Shift register circuit.

In MATHEMATICA we illustrate the procedure with the Hamming code K [7,4,3]: We first receive the generator polynomials to the code by the command:

```
In[10] : = gp=GeneratorPolynomials[7,x]
```

(This command gives all possible $g(x)$ of length 7)

```
Out[10] ={1, 1+x, 1+x+x^3, 1+x^2 +x^3, 1+x+x^2+x^4,
1+x^2+x^3+x^4, 1+x+x^2+x^3+x^4+x^5+x^6, 1+x^7}
In[11] : =      g=gp[[3]]
```

(We choose the third polynomial as $g(x)$ i.e. $g(x) = 1 + x + x^3$ )

```
Out[11] = 1 + x + x^3

In[12] : = ShowCyclicCode[g,7,x]

Out[12] =

GENERATOR    POLYNOMIAL      g = 1 + x + x^3
```

$$\text{GENERATOR MATRIX} \quad GM = \begin{pmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 \end{pmatrix}$$

$$\text{PARITY CHECK POLYNOMIAL } h = 1 + x + x^2 + x^4$$

$$\text{PARITY CHECK MATRIX} \quad HM = \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{pmatrix}$$

From MATHEMATICA we then get the shift-register (systematic encoding)

```
In[13] : = ShowSystematicEncoderCyclicCode[g,7,x]

Out[13] =
```

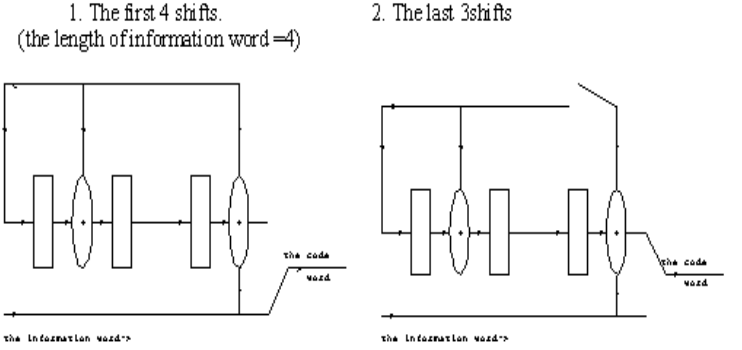



**Fig. 3.** Shift-register (systematic encoding) for cyclic code with generator polynomial $g(x) = 1 + x + x^3$

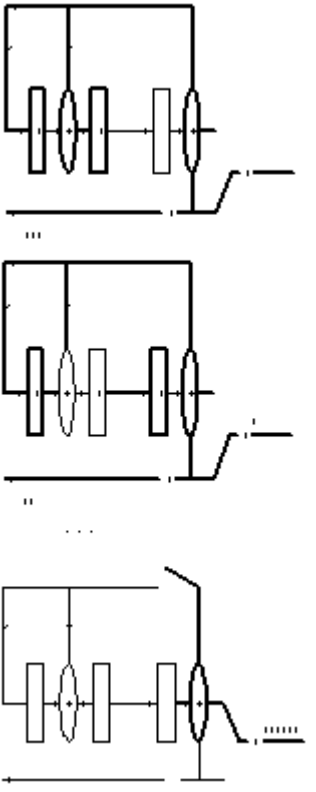
With the information word u = (1, 0, 1, 1) we get the code word as follows:

```
In[14] : =  u={1,0,1,1}; U=x^6+x^5+x^3;
r=PolynomialMod[PolynomialMod[U,g],2];
V=PolynomialMod[U-r,2] v=CoefficientList[V,x]
Out[14] = 1 + x^3  + x^5  + x^6 {1, 0, 0, 1, 0, 1, 1}
```

Using the package we get:

```
In[15] : = SystematicEncodeCyclicCode[g,u,7,x]
Out[15] =
THE INFORMATION POLYNOMIAL = x^3  + x^5  + x^6
```

THE CODE POLYNOMIAL = 1 + $x^3$ + $x^5$ + $x^6$



THE CODE WORD = (1, 0, 0, 1, 0, 1, 1)

From the information word u = (1, 0, 1, 1) we have thus got the code word $v(x)$ = 1 + $x^3$ + $x^5$ + $x^6$ = (1, 0, 0, 1, 0, 1, 1) which will be sent through the communication channel. Due to noise in the channel, the receiver might receive $w(x) = v(x) + e(x)$, where $e(x)$ is an error vector. We illustrate this in our example by letting $e(x)$ = $x^4$, i. e $w(x)$ = 1 + $x^3$ + $\mathbf{x^4}$ + $x^5$ + $x^6$. Our next aim is to decode the received word w = (1, 0, 0, 1, 1, 1, 1).

In general, when we receive a word w from the communication channel we use the concept of syndromes to reconstruct the information word. For a general linear code this requires the following steps:

- compute the syndrome $s^{tr}$ = H $w^{tr}$
- divide the received words into cosets and fix a coset leader e with smallest hamming weight to each coset
- decode as v = w - e

Within the cyclic codes we construct the syndrome polynomial $s(x)$ from $w(x) = q(x) \cdot g(x) + s(x)$. For every $s(x)$ we chose an error vector $e(x)$ with smallest hamming weight and decode as $v(x) = w(x) - e(x)$. For the Hamming code K[7,4,3] with generator polynomial $g(x) = 1 + x + x^3$ we construct the syndromes and we get for error $e(x) = x^4$, syndrome $s(x)$ is $x^2 + x$.

We use MATHEMATICA to describe the methods of decoding. We consider (as in the example above) the Hamming code K [7, 4, 3], were we received the word $w(x) = 1 + x^3 + \mathbf{x^4} + x^5 + x^6$. The following steps illustrate how the information word could be reconstructed:

```
In[16] : =V=1+x^3+x^5+x^6; e=x^4; W=V+e;
s=PolynomialMod[PolynomialMod[W,g],2]
Out[16] = x + x^2
```

We have got the syndrome polynomial and from the syndrome list above we recognize the error polynomial $e(x) = x^4$. An alternative way to decode the received word $w(x) = 1 + x^3 + \mathbf{x^4} + x^5 + x^6$ is given in next steps:

```
In[17] : = w =CoefficientList[W,x]
Out[17] =   {1, 0, 0, 1, 1, 1, 1}
In[18] : =H = CyclicCode[g,7,x][[1]] Syndrome[H,w]
```

$$\text{Out[18]} = \begin{pmatrix} 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 & 1 & 0 & 0 \end{pmatrix} \qquad s = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$$

We see that this column is identical with the 5:th column in H, indicating that we have an error in the 5:th position.

## References

1. MacWilliams, F. J., and Sloane, N. J. A. (1977) The Theory of Error - Correcting Codes. North - Holland, Amsterdam.
2. Adamek, J., Foundations of Coding , John Wiley & Sons, Inc 1991