

11. BCH Codes: Constructing them and finding the Syndrome of a Message

We claim that a two error correcting encoding polynomial can be created by multiplying a primitive polynomial, $p(x)$, by another polynomial $p_3(x)$ defined by the condition

$$\text{rem } p_3(x^3) = 0, \text{ on dividing by } p(x),$$

and

codes allowing correction of 3 errors can be obtained by having a similar factor with 3 replaced by 5 as well as this one, and so on.

We will show that this is true by showing how we can find and correct errors. First we consider the question:

How do we find polynomials p_3 , p_5 and so on, to create such codes? We illustrate the procedure described for p_3

We want to find an equation of lowest degree obeyed by the remainders of the powers of x that are divisible by three. This equation will be a linear dependence among the remainders of these powers.

It is a fundamental fact, true by definition, that there is a linear dependence among any $d+1$ vectors in a d dimensional real space. There is, furthermore, a standard way to find a linear dependence among any set of vectors. It is called row reduction.

To do it, you write down the vectors and add an appropriate multiple (0 or 1) of (say) the first vector to each of the others so as to eliminate some one component from them all. You then add an appropriate multiple of the second vector to eliminate another component from the rest. And so on.

If your vectors are linearly dependent, eventually one of your vectors will become the (0) vector. Your equation will then be that this vector originally was the sum of the others that you added to it to get the (0) vector.

We illustrate this procedure for the primitive polynomial $1 + x^2 + x^5$.

The first step is to write down the remainders of the first few powers that are divisible by 3. These can be read off from the remainder table for that polynomial, or we can construct a similar table whose successive rows represent powers that increase by 3.

Let us notice how to construct such a table. Similar tables for this and for other powers will be very useful to us soon,

We will start from the 0^{th} power whose remainder is 1. But suppose in general that the present power has remainder $a + bx + cx^2 + dx^3 + ex^4$.

What happens when this remainder is multiplied by x^3 ?

We get as new remainder $ax^3 + bx^4 + c(x^2 + 1) + d(x^3 + x) + e(x^4 + x^2)$, or

$$c + dx + (c + e)x^2 + (a + d)x^3 + (b + e)x^4.$$

We can use this fact to write a table whose successive entries are x^0, x^3, x^6, \dots ,

However we get them we find that the remainders of the powers divisible by 3 upon dividing by our primitive polynomial are

power	1	x	x ²	x ³	x ⁴
0	1	0	0	0	0
3	0	0	0	1	0
6	0	1	0	1	0
9	0	1	0	1	1
12	0	1	1	1	0
15	1	1	1	1	1

Remainders of powers dividing by $(1+x^2+x^5)$

Since these rows form 6 vectors in a five dimensional space, there must be a linear dependence among them. It is easy to see here that the sum of all but the power 3 is the 0 vector.

Doing it by row reduction, we can use the 0 power remainder to remove the 0th power from the rest, the 3rd power remainder to eliminate power 3, the power (6+3) row to eliminate the first power column, the 9+6+3+3th to eliminate the 4th power column the 12th +6th to eliminate the 2nd, and we find that the 15th is the 0th plus 3rd + 3rd +6th +9th + 6th + 12th +6th, which boils down to everything except the 3rd power.)

We deduce then that, on dividing by $1 + x^2 + x^5$, the remainder of

$$1 + x^6 + x^9 + x^{12} + x^{15}$$

is 0, from which we conclude that

$$p_3(x) = 1 + x^2 + x^3 + x^4 + x^5$$

is the second factor we want to create in our encoding polynomial for a two error correcting code:

$$(1 + x^2 + x^5)(1 + x^2 + x^3 + x^4 + x^5).$$

Exactly the same approach can be used to obtain similar factors starting from any primitive polynomial, and powers that are multiples of any odd numbers.

We know that each even power obeys the same equation as does the power that is the largest odd factor of its power. Thus x^{10} obeys the same equation as x^5 does, and x^{28} obeys the same equation as x^7 does.

How come?

If the remainder of $(x^5 + x^2 + 1)$ is 0, then on squaring both sides of this equation we deduce that the remainder of $(x^{10} + x^4 + 1)$ is also 0, which says that x^2 obeys the same equation as x .

Similarly, x^6 obeys the same equation as x^3 .

11.2 Finding Two or More Errors: Step 1: Finding the Error Syndrome

Suppose we encode using an encoding polynomial $p(x)$ $p_3(x)$. And suppose further that the received word $R(x)$ was garbled and has at most 2 errors. Then we will have

$$R(x) = m(x)p(x)p_3(x) + ax^{e1} + bx^{e2}$$

where a and b are each either 0 or 1 and $e1$ and $e2$ are the two error powers if there are two errors.

If we take the remainder of $R(x)$ on dividing by $p(x)$ the first or message term here will give remainder 0, so we will find the remainder of the error terms.

Similarly, if we take each power that appears in $R(x)$ and replace it by its cube, we will create the polynomial $R(x^3)$, and as a result of the factor $p_3(x)$ in the encoding polynomial, the first term above, $m(x^3)p(x^3)p_3(x^3)$ will also have 0 remainder on dividing by p .

This means that the remainder of $R(x^3)$ will be that of $ax^{3e1} + bx^{3e2}$ on dividing by p .

To summarize,

with a two error correcting code that has the structure described here, we can find not only the sum of the remainders of the error monomials, but also the sum of the third powers of the error monomials.

In the identical way,
we can find the sum of the fifth powers of the error monomials in a three error correcting code whose encoding polynomial has the additional factor $p_5(x)$, and so on.

We have to address two questions at this point.

First, how can we actually find the remainder of $R(x^3)$ conveniently?

Second, how do we use the remainders of sums of odd powers of the errors to find the errors themselves?

The answer to the first question is exactly like the way we find the remainder of the sum of the errors in a single error correcting code.

To find the sum of the errors we took the dot product of R with the remainder table starting

26	1	1	1	0	1	23	0	0	0	0	0	0	0
27	1	1	0	1	0	11	0	0	0	0	0	0	0
28	0	1	1	0	1	22	0	0	0	0	0	0	0
29	1	0	0	1	0	9	0	0	0	0	0	0	0
30	0	1	0	0	1	18	0	0	0	0	0	0	0
31	1	0	0	0	0	1							
error		pwr					2	3	1	3	3		24
Error rem		and	id				0	1	1	1	1		30

To do all this requires entering the information on the power 1 row needed to construct the remainder table,

making one entry to get the rows to be summed in making the dot product

Summing and computing mod 2;

then making one more entry in the next column (or perhaps two)

The rest is all copying.

Here are the entries used to get the tables above, with the received message r starting in A18. The columns shown are columns A B C D E F and G. A contains the received word, B the power identifier for the row C the constant term in the remainder of the corresponding power, D the x coefficient of that remainder, etc.

R	Power	0	1	2	3	4
1	0	1	0			
1	=B18+1	=G18	=C18	=MOD(D18+G18,2)	=E18	=F18

This allows construction of the remainder table for this code.

It is convenient to add an identifier for the actual remainder in column H and in subsequent set of 5 column you can put the contribution from the current row to the dot product

Identifier	taking dot product rT	
=C18*1+D18*2+E18*4+F18*8+G18*16	=A18*C18	=A18*D18
=C19*1+D19*2+E19*4+F19*8+G19*16	=A19*C19	=A19*D19

You then have the remainder identifier and the start of taking the dot product. If the remainder identifier of the dot product is located in box O51 then you can obtain the corrected coded message in a column by entering in row 18 of that column =mod(A18+(\$O\$51=H18),2), where we make use of the fact that what is in the inner parenthesis here is, (in Excel) a function that is 1 when the equality holds and 0 otherwise.

If you want to see what power had the error you can find it with the first instruction in the next table, suitably copied down, and a less slick way to make the correction just described is by use of the instruction shown in the second column in this next table.

Error	mC
Power	

=IF(\$O\$51=H18,B18,0) =MOD(A18+IF(\$O\$51=H18,1,0),2)
 =IF(\$O\$51=H19,B19,0) =MOD(A19+IF(\$O\$51=H19,1,0),2)

We compute the dot product as started above, (copied from all 5 columns of the remainder table) by using the lower two instructions given in the table below. Here we assume that the first entry of the dot product is in column J, and we give only the first two columns of the dot product here.

=A47*C47 =A47*D47
 =A48*C48 =A48*D48

 =SUM(J18:J49) =SUM(K18:K49)
 =MOD(J50,2) =MOD(K50,2)

O51 is the numerical identifier of the dot product which is the last row here. The instruction for it can be gotten by copying any numerical row identifier from the remainder table into box O51.

Notice that you can change codes (of the same length) by changing the entries of line 19 of the remainder table and copying them down.

Now,
if you have all this, you can find the identifier and power of the sum of the third powers of the message monomials by creating a table like the remainder table whose powers go up by 3 each time to the right of everything you have, and then doing one clever copying.

If you copy everything that you have done to the right of your ordinary (up by one power from row to row) remainder table except for the last error correcting column (that is, copying the remainder identifier dot product computation and the power locator) to the right of the original remainder table in the same relative position to the right of this new table, you will be able to product the dot product of your message with the third power remainders as remainders, by and power identifier and by its numerical identifier, whichever you want: it is the sum of the third power of the errors.

If you wanted to correct three errors, you could add a table in which the powers go up by 5 each time and copy once more to its right, and you can find the remainders of the fifth powers of the error monomials, if your code is such that your encoded messages have the 0 for such remainders. And so on.

Essentially all the work required lies in constructing the appropriate power remainder table, and there is one such task for each additional error you want to correct.

Next we will describe how you can use the information provided by this syndrome: namely the remainder of the sum of the error monomials, and the remainder of the sum of their third powers.

Exercise: Construct a spreadsheet that for a two error correcting code of the kind we are considering, given some primitive polynomial of degree 5 other than $1+x^2+x^5$, and a received word r, produces the sum of the error monomials and the sum of their third powers, both as remainders, but also by their identifiers and by their powers.