Reed-Muller Codes

Reed-Muller Codes

Reed-Muller codes are among the oldest known codes and have found widespread applications. They were discovered by Muller and provided with a decoding algorithm by Reed in 1954.

These codes were initially given as binary codes, but modern generalizations to q-ary codes exist. We will restrict our investigation to the binary case.

One of the interesting things about these codes is that there are several ways to describe them and we shall look at two of these. One reason for doing this is to see how to move to the generalization even though we will not do so.

Recursive Definition

For each positive integer *m* and each integer *r* with $0 \leq r \leq m$, there is an rth order Reed-Muller Code $\mathcal{R}(r,m)$. We start our definition by considering the 1st order case (r = 1).

Definition: The (first order) ***Reed-Muller codes $\mathcal{R}(1,m)$*** are binary codes defined for all integers $m \geq 1$, recursively by:

(i) $\mathcal{R}(1,1) = \{00,01,10,11\} = \mathbb{Z}_2^2$.

(ii) for $m > 1$,

$\mathcal{R}(1,m) = \{(\mathbf{u},\mathbf{u}), (\mathbf{u},\mathbf{u}+\mathbf{1}): \mathbf{u} \in \mathcal{R}(1,m-1)$ and $\mathbf{1}$ = all 1 vector$\}$.

Example

Thus,

$\mathcal{R}(1,2)$ = {0000, 0101, 1010, 1111, 0011, 0110, 1001, 1100} and

$\mathcal{R}(1,3)$ =
{ 00000000, 00001111,
01010101, 01011010,
10101010, 10100101,
11111111, 11110000,
00110011, 00111100,
01100110, 01101001,
10011001, 10010110,
11001100, 11000011 }

(u, u+v)-construction

This recursive construction is a special case of a more general construction of linear codes.

Theorem [(u, u+v)-construction]: Let C_i be an $[n,k_i,d_i]$- linear q-ary code for i = 1,2. Then the code C defined by,

$$C = \{(\mathbf{u}, \mathbf{u}+\mathbf{v}) : \mathbf{u} \in C_1, \mathbf{v} \in C_2\}$$

is a $[2n, k_1 + k_2, \min(2d_1, d_2)]$ linear q-ary code.

Pf: It is clear that C is a q-ary [2n, k, d] linear code for some k and d, which we now determine.

Consider the map from $C_1 \times C_2 \rightarrow C$ given by $(c_1,c_2) \rightarrow (c_1, c_1+c_2)$. This is easily seen to be a bijection, so the size of C is the same as that of $C_1 \times C_2$, which is $q^{k_1}q^{k_2} = q^{k_1+k_2}$. So, the dimension of C is $k_1 + k_2$.

(u, u+v)-construction

Theorem [(u, u+v)-construction]: Let C_i be an $[n,k_i,d_i]$- linear q-ary code for i = 1,2. Then the code C defined by,

$$C = \{(\mathbf{u}, \mathbf{u+v}): \mathbf{u} \in C_1, \mathbf{v} \in C_2\}$$

is a $[2n, k_1 + k_2, \min(2d_1, d_2)]$ linear q-ary code.

Pf(cont): Consider a non-zero code word (c_1, c_1+c_2) of C. If $c_2 = 0$ then $c_1 \neq 0$ and

$$wt((c_1, c_1+c_2)) = wt((c_1,c_1)) = 2\, wt(c_1) \geq 2d_1 \geq \min(2d_1, d_2).$$

On the other hand, if $c_2 \neq 0$, then

$$wt((c_1, c_1+c_2)) = wt(c_1) + wt(c_1+c_2) \geq wt(c_1) + (wt(c_2) - wt(c_1))$$

$$\geq wt(c_2) \geq \min(2d_1,d_2).$$ Thus, $d \geq \min(2d_1,d_2)$.

If $x \in C_1$ has wt d_1, and $y \in C_2$ has wt d_2, then

$(x,x) \in C$ has wt $2d_1$ and $(0,y) \in C$ has wt d_2

so, $d \leq \min(2d_1, d_2)$, and so we have equality.

Dimension

Proposition: For $m > 0$, the Reed-Muller code $\mathcal{R}(1,m)$ is a binary $[2^m, m+1, 2^{m-1}]$ linear code, in which every codeword except $\mathbf{0}$ and $\mathbf{1}$ has weight 2^{m-1}.

Pf: Clearly, $\mathcal{R}(1,1)$ is a $[2,2,1]$ code. We will establish the result by induction on m. Assume then that $\mathcal{R}(1,m-1)$ is a $[2^{m-1}, m, 2^{m-2}]$ code. By our inductive definition, $\mathcal{R}(1,m)$ is constructed by using the $(u,u+v)$ construction with $C_1 = \mathcal{R}(1,m-1)$ and $C_2 = \{\mathbf{0},\mathbf{1}\}$. Thus, $\mathcal{R}(1,m)$ is a $[2(2^{m-1}), m + 1, \min(2(2^{m-2}), 2^{m-1})]$ code, i.e., a $[2^m, m+1, 2^{m-1}]$ code.

A codeword in $\mathcal{R}(1,m)$ is either of the form (u,u) or $(u,u+\mathbf{1})$ with u in $\mathcal{R}(1,m-1)$. If the word is of the form (u,u) then u can not be $\mathbf{0}$ or $\mathbf{1}$ in $\mathcal{R}(1,m-1)$ else the word would be $\mathbf{0}$ or $\mathbf{1}$ in $\mathcal{R}(1,m)$. So,

Dimension

Proposition: For $m > 0$, the Reed-Muller code $\mathcal{R}(1,m)$ is a binary $[2^m, m+1, 2^{m-1}]$ linear code, in which every codeword except $\mathbf{0}$ and $\mathbf{1}$ has weight 2^{m-1}.

Pf(cont): by the induction hypothesis, u has weight 2^{m-2}, and so (u,u) has weight $2(2^{m-2}) = 2^{m-1}$.

Now assume the word is of the form $(u,u+\mathbf{1})$.

If $u = \mathbf{0}$, then $u+\mathbf{1}$ is $\mathbf{1}$ and $wt(u,u+\mathbf{1}) = 2^{m-1}$.

If $u = \mathbf{1}$, then $u+\mathbf{1}$ is $\mathbf{0}$ and $wt(u,u+1) = 2^{m-1}$.

For every other u in $\mathcal{R}(1,m-1)$ $wt(u) = 2^{m-2}$, that is, exactly half of the coordinates of u are 1. So, in $u+\mathbf{1}$ we also have half of the coordinates being 1, thus $wt(u,u+\mathbf{1}) = 2(2^{m-2}) = 2^{m-1}$.

Generator Matrix

Proposition: (i) A generator matrix of $\mathcal{R}(1,1)$ is

$$\begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}$$

(ii) If G_m is a generator matrix for $\mathcal{R}(1,m)$, then a generator matrix for $\mathcal{R}(1,m+1)$ is

$$G_{m+1} = \begin{pmatrix} G_m & G_m \\ 0\cdots 0 & 1\cdots 1 \end{pmatrix}$$

Pf: Homework problem 23 (pg. 37) applied to this situation (this will be assigned).

r^th Order Reed-Muller Codes

Definition: The 0th order Reed-Muller code $\mathcal{R}(0,m)$ is defined to be the repetition code {0,1} of length 2^m.

For any r ≥ 2, the rth order Reed-Muller code $\mathcal{R}(r,m)$ is defined recursively by:

$$\mathcal{R}(r, m) = \begin{cases} Z_2^{2r} & \text{if } m = r \\ \{(u, u+v) : u \in \mathcal{R}(r, m-1), v \in \mathcal{R}(r-1, m-1)\} & \text{if } m>r \end{cases}$$

Example

We construct $\mathcal{R}(2,3)$. By the definition we need to use $\mathcal{R}(2,2)$ and $\mathcal{R}(1,2)$ in the construction.

v=0000	v=0101	v=1010	v=1111	v=0011	v=0110	v=1001	v=1100
00000000	00000101	00001010	00001111	00000011	00000110	00001001	00001100
00010001	00010100	00011011	00011110	00010010	00010111	00011000	00011101
00100010	00100111	00101000	00101101	00100001	00100100	00101011	00101110
00110011	00110110	00111001	00111100	00110000	00110101	00111010	00111111
01000100	01000001	01001110	01001011	01000111	01000010	01001101	01001000
01010101	01010000	01011111	01011010	01010110	01010011	01011100	01011001
01100110	01100011	01101100	01101001	01100101	01100000	01101111	01101010
01110111	01110010	01111101	01111000	01110100	01110001	01111110	01111011
10001000	10001101	10000010	10000111	10001011	10001110	10000001	10000100
10011001	10011100	10010011	10010110	10011010	10011111	10010000	10010101
10101010	10101111	10100000	10100101	10101001	10101100	10100011	10100110
10111011	10111110	10110001	10110100	10111000	10111101	10110001	10110111
11001100	11001001	11000110	11000011	11001111	11001010	11000101	11000000
11011101	11011000	11010111	11010010	11011110	11011011	11010100	11010001
11101110	11101011	11100100	11100001	11101101	11101000	11100111	11100010
11111111	11111010	11110101	11110000	11111100	11111001	11110110	11110011

Dimension

Theorem 22: If $G_{r,m}$ is a generator matrix of $\mathcal{R}(r,m)$ then a generator matrix for $\mathcal{R}(r+1, m+1)$ is given by:

$$G_{r+1,m+1} = \begin{pmatrix} G_{r+1,m} & G_{r+1,m} \\ 0 & G_{r,m} \end{pmatrix}$$

The dimension of $\mathcal{R}(r,m)$ is:

$$1 + \binom{m}{1} + \binom{m}{2} + \cdots + \binom{m}{r}.$$

The minimum weight (= minimum distance) of $\mathcal{R}(r,m)$ is 2^{m-r}.

$\mathcal{R}(m-1,m)$

Proposition: $\mathcal{R}(m-1,m)$ consists of all binary words of length 2^m that have even weight. Therefore, if $r < m$, $\mathcal{R}(r,m)$ contains code words of even weight only.

Pf: We prove this by induction on m. Clearly, $\mathcal{R}(0,1) = \{00,11\}$ consists of all binary words of length 2 with even weight. Assume that this is true for $\mathcal{R}(m-2, m-1)$. A typical codeword in $\mathcal{R}(m-1,m)$ looks like $(u, u+v) = (u, u) + (0,v)$ with $u \in \mathcal{R}(m-1,m-1)$ and $v \in \mathcal{R}(m-2, m-1)$. Now, (u,u) has even weight for any u, and $(0,v)$ has even weight by the induction hypothesis. So,

$$wt((u,u+v)) = wt((u,u)) + wt((0,v)) - 2\,(u,u)\bullet(0,v)$$

which is therefore even. So $\mathcal{R}(m-1,m)$ contains only even weight vectors. Since dim $\mathcal{R}(m-1,m) = 2^m - 1$, it contains all even weight vectors. Finally, if $r < m$, then $\mathcal{R}(r,m) \subset \mathcal{R}(m-1,m)$ from which the second statement follows.

Duals

Theorem 23: $\mathcal{R}$(m – r – 1, m) and $\mathcal{R}$(r, m) are dual codes.

Pf: We will establish the orthogonality of these codes by induction on m. For m = 2, both codes exist only if r = 0 or 1 (and we get the same pair). We have $\mathcal{R}$(0,2) = {0000,1111} while $\mathcal{R}$(1,2) = {0000, 0101, 1010, 1111, 0011, 0110, 1001, 1100}. Since all the vectors in $\mathcal{R}$(1,2) have even weight, every vector of $\mathcal{R}$(0,2) is orthogonal to every vector of $\mathcal{R}$(1,2). Now assume the statement is true for m-1. To prove the assertion we need only show that the generators of $\mathcal{R}$(r,m) are orthogonal to the generators of $\mathcal{R}$(m-r-1,m). Consider the generator matrices of these two codes as given in Thm 22.

$$\boldsymbol{G}_{r,m} = \begin{pmatrix} \boldsymbol{G}_{r,m-1} & \boldsymbol{G}_{r,m-1} \\ \mathbf{0} & \boldsymbol{G}_{r-1,m-1} \end{pmatrix} \qquad \boldsymbol{G}_{m-r-1,m} = \begin{pmatrix} \boldsymbol{G}_{m-r-1,m-1} & \boldsymbol{G}_{m-r-1,m-1} \\ \mathbf{0} & \boldsymbol{G}_{m-r-2,m-1} \end{pmatrix}$$

Duals

Theorem 23: $\mathcal{R}$(m – r – 1, m) and $\mathcal{R}$(r, m) are dual codes.

Pf(cont): The rows of the form (a,a) are clearly orthogonal to those of the form (b,b). Rows of the form (a,a) are orthogonal to those of the form (0,d) by the induction hypothesis. For the same reason, rows of the form (0,c) are orthogonal to those of the form (b,b). Finally, because $\mathcal{R}$(m-r-2,m-1) $\subset$ $\mathcal{R}$(m-r-1,m-1), the rows of the form (0,c) are orthogonal to those of the form (0,d).

This shows that $\mathcal{R}$(m-r-1,m) $\subset$ $\mathcal{R}(r,m)^{\perp}$. By Thm 22,

$$\begin{aligned}\dim(R(r,m)^{\perp}) &= 2^m - [1 + \binom{m}{1} + \cdots + \binom{m}{r}] \\ &= \binom{m}{r+1} + \binom{m}{r+2} + \cdots + \binom{m}{m} \\ &= \binom{m}{m-r-1} + \binom{m}{m-r-2} + \cdots + 1 \\ &= \dim(R(m-r-1,m))\end{aligned}$$

Reed Decoding

One reason that Reed-Muller codes are useful is that there is a simple decoding algorithm for them.

We illustrate the method known as Reed Decoding with an example. Consider the code $\mathcal{R}(1,3)$ with generator matrix:

$$\begin{pmatrix} \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} \\ \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{1} \\ \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{1} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} \end{pmatrix}.$$

The rows of this matrix are basis vectors for the code; label them v_0, v_1,v_2 and v_3. Any vector v of the code is a linear combination of these, i.e., $v = a_0v_0 + a_1v_1 + a_2v_2 + a_3v_3$. Written as a vector, we have $v = (a_0, a_0+a_1, a_0+a_2, a_0+a_1+a_2, a_0+a_3, a_0+a_1+a_3, a_0+a_2+a_3, a_0+a_1+a_2+a_3)$.

Reed Decoding

$v = (a_0, a_0+a_1, a_0+a_2, a_0+a_1+a_2, a_0+a_3, a_0+a_1+a_3, a_0+a_2+a_3, a_0+a_1+a_2+a_3)$

If no errors occur, a received vector $r = (y_0, y_1, y_2, y_3, y_4, y_5, y_6, y_7)$ can be used to solve for the a_i other than a_0 in several ways (4 ways for each) namely:

$$a_1 = y_0 + y_1 = y_2 + y_3 = y_4 + y_5 = y_6 + y_7$$
$$a_2 = y_0 + y_2 = y_1 + y_3 = y_4 + y_6 = y_5 + y_7$$
$$a_3 = y_0 + y_4 = y_1 + y_5 = y_2 + y_6 = y_3 + y_7$$

If one error has occurred in r, then when all the calculations above are made, 3 of the 4 values will agree for each a_i, so the correct value will be obtained by majority decoding. Finally, a_0 can be determined as the majority of the components of $r + a_1v_1 + a_2v_2 + a_3v_3$ (why?).

Example

Suppose that v = 10100101 is received as 10101101. Using,

$a_1 = y_0 + y_1 = y_2 + y_3 = y_4 + y_5 = y_6 + y_7$

$a_2 = y_0 + y_2 = y_1 + y_3 = y_4 + y_6 = y_5 + y_7$

$a_3 = y_0 + y_4 = y_1 + y_5 = y_2 + y_6 = y_3 + y_7$

we calculate:

$a_1 = 1 = 1 = 0 = 1$ so $a_1 = 1$

$a_2 = 0 = 0 = 1 = 0$ so $a_2 = 0$

$a_3 = 0 = 1 = 1 = 1$ so $a_3 = 1$

and $a_0 = 1$ since 10101101 + 01010101 + 00001111 = 11110111.

So, v = 11111111 + 01010101 + 00001111 = 10100101.

Another view

In a projective geometry PG(n,q) a subgeometry of co-dimension 1 (a subspace isomorphic to PG(n-1,q)) is called a ***hyperplane***.

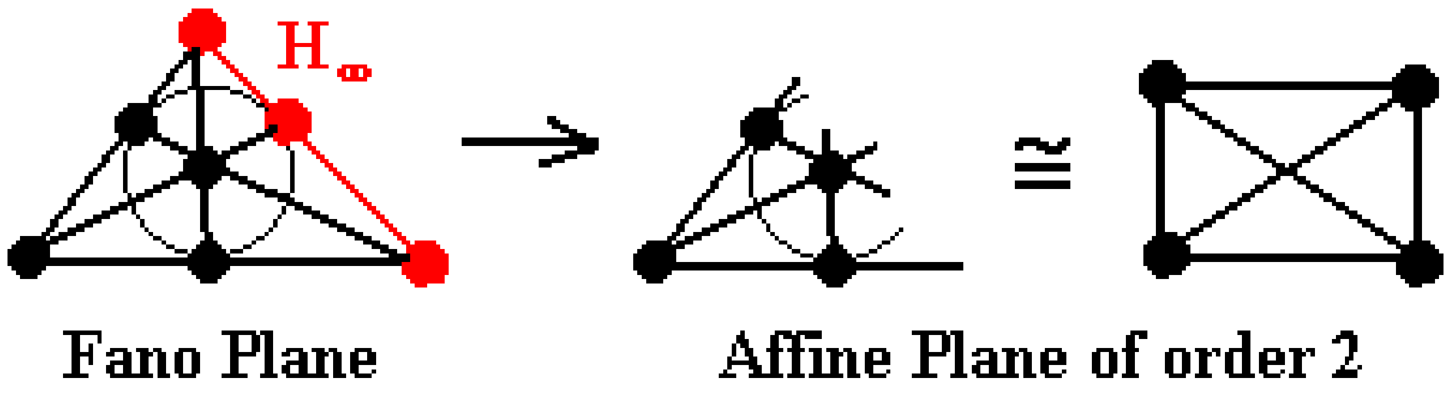
If H is a hyperplane of PG(n,q) then the structure obtained by removing the points of H from the geometry is called an ***affine geometry*** and denoted by AG(n,q). The point set of an affine geometry AG(n,q) is isomorphic to the vector space V[n,q]. Thus, the number of points in AG(n,q) is q^n.

Starting with a projective plane PG(2,q), a hyperplane is a line, so we obtain an affine plane AG(2,q) by removing a line from the projective plane.

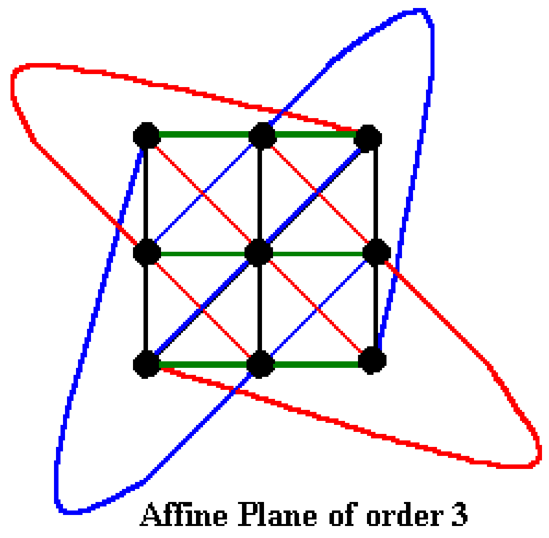
Examples of Affine Planes

Examples:

1.



2.



Flats in an Affine Geometry

Suppose that an affine geometry AG(n,q) is obtained from a projective geometry PG(n,q) by the removal of a hyperplane H. Any subspace of PG(n,q) of dimension m, when the points of H are removed is called an ***m-flat*** in AG(n,q).

Thus, in an affine plane, the 1-flats are the lines of the plane and the points are the 0-flats of the plane.

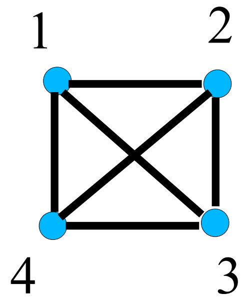
The number of points in an m-flat of AG(n,q) is q^m.

The ***point/m-flat incidence matrix*** of an affine geometry is a 0-1 matrix whose columns are the points and whose rows are the m-flats, where an entry is 1 iff the corresponding point is contained in the corresponding m-flat.

Example

The point/1-flat (line) incidence matrix of AG(2,2) is:

$$\begin{array}{c|cccc} \text{points} & 1 & 2 & 3 & 4 \\ \hline & 1 & 1 & 0 & 0 \\ & 1 & 0 & 1 & 0 \\ & 1 & 0 & 0 & 1 \\ & 0 & 1 & 1 & 0 \\ & 0 & 1 & 0 & 1 \\ & 0 & 0 & 1 & 1 \end{array}$$



Notice that each row is a vector in $\mathcal{R}(1,2)$, and if we take the span of all of these vectors we will also get 0000 and 1111, giving all 8 vectors of $\mathcal{R}(1,2)$. This is a particular case of the following theorem.

Another View

Theorem: The codewords of $\mathcal{R}$(r,m) of minimum weight are precisely the rows of the point/(m-r)-flats incidence matrix of AG(m,2). Furthermore, these codewords span $\mathcal{R}$(r,m).

Back to Mariner 9

Recall that in the Mariner 9 mission, the data consisted of binary 6-tuples (64 grayness levels) and transmission restrictions permitted coding that would lengthen the transmitted words to about 30 bits.

The 5-repeat code would satisfy this condition, but it is only 2 error correcting.

The code that was chosen however is $\mathcal{R}(1,5)$ which is a [32, 6, 16] binary code. Code words are 32 bits long and there are 64 of them. Moreover, as the minimum distance is 16, this code is 7-error correcting.

Back to Mariner 9

Encoding:

As there are 64 code words and 64 data types, any assignment of code word to data type will work, but the requirement that the encoding should require no memory meant that an arbitrary assignment would not do.

Since $\mathcal{R}(1,5)$ is a 6 dimensional code, there is a basis with 6 elements (any linear combination of which gives a code word). The data type 6-tuple is used to provide the coefficients for the linear combination of the basis vectors ... thus associating a unique code word to each data type.

This simple computation can be hard wired and requires no memory.

Back to Mariner 9

Decoding:
As we have previously mentioned, the real reason for selecting this code was that it had a very fast decoding algorithm which we now describe.

First, convert all the code words (and the received vector) to ± 1 vectors by turning the 0's into -1's. Take the dot product of the received vector with each of the code words in turn. As soon as the result is 16 or greater, decode as that code word.

Suppose no errors have been made in transmission. Then the dot product of the received vector with itself will be 32 and with any other codeword will be 0 or -32.

Back to Mariner 9

Decoding:

This follows since the distance between two code words is the weight of their difference (which is another code word) and so is either 0, 16 or 32. If 0, the code words are the same. If 32, the code words have no common component and the dot product of the ±1 form will be -32. In all remaining cases, 16 places are the same and 16 places are different, giving a dot product of 16 – 16 = 0.

For each error that occurs, the dot product will decrease by 2 (or increase by 2 from an incorrect codeword). If no more than 7 errors occur, the dot product with the correct code word decreases to at least 18 and the dot product with incorrect code words increases to at most 14 ... so correct decoding will occur. If 8 or more errors occur, there will be dot products of at least 16 but correct decoding is not possible.

Back to Mariner 9

Decoding:

Even though this is a rapid decoding algorithm, the computations involved can be speeded up by a factor of 3 by using a Fast Fourier Transform for Abelian groups. This is what was actually done by the “green machine”.