Classifying polynomials of linear codes

Jurrius, R.P.M.J.

R.P.M.J. Jurrius

Classifying polynomials of linear codes

Master thesis, defended on June 27, 2008

Thesis advisors: G.R. Pellikaan (TU/e)
R. de Jong (UL)

Mathematisch Instituut, Universiteit Leiden

Contents

1 Introduction and definitions 5

1.1 Introduction . . . 5

1.2 Definitions . . . 5

1.2.1 Linear codes . . . 5

1.2.2 Weight distributions . . . 6

1.2.3 More about codes . . . 8

1.2.4 Matroids . . . 9

1.2.5 Gaussian binomials and other products . . . 10

1.3 Notes . . . 10

2 Weight enumerator 11

2.1 Generalized weight enumerator . . . 11

2.2 Extended weight enumerator . . . 15

2.3 Another algorithm . . . 18

2.4 Generalized extended weight enumerator . . . 21

2.5 Notes . . . 23

3 Connections 25

3.1 Extended in terms of generalized . . . 25

3.2 Generalized in terms of extended . . . 27

3.3 Weight enumerator and Tutte polynomial . . . 28

3.4 Overview . . . 30

3.5 Notes . . . 31

4 MacWilliams identities 33

4.1 Using the Tutte polynomial . . . 33

4.2 Generalized MacWilliams identities . . . 34

4.3 An alternative approach . . . 35

4.4 Notes . . . 38

5 Examples **39**
5.1 MDS codes 39
5.2 Two $[6, 3]$ codes 41
5.3 Notes 45

6 Implementation and complexity **47**
6.1 About the computer implementation 47
6.2 The implementation 48
6.3 Example: the ternary Golay code 51
6.4 Complexity calculations 53
6.5 Notes 54

7 Conclusions **55**

Chapter 1

Introduction and definitions

1.1 Introduction

The weight enumerator of a linear code is a classifying polynomial associated with the code. Besides its intrinsic importance as a mathematical object, it is used in the probability theory around codes. For example, the weight enumerator of a binary code is very useful if we want to study the probability that a received message is closer to a different codeword than to the codeword sent. (Or, rephrased: the probability that a maximum likelihood decoder makes a decoding error.)

We will generalize the weight enumerator in two ways, which lead to polynomials which are better invariants for a code. A procedure for the determination of these polynomials is given. We will show that the two generalisations determine each other, and that they connect to the Tutte polynomial of a matroid, thus linking coding theory and matroid theory. Most of the generalisations and connections have been studied before, but mostly only one-way, and a complete overview was never given.

We will use the established connections to derive MacWilliams relations for our generalisations. Also other examples of the developed machinery will be given, as well as a computer implementation.

1.2 Definitions

1.2.1 Linear codes

Definition 1.2.1 *Let q be a prime power, and let $\mathbb{F}_q$ be the finite field with q elements. A linear subspace of $\mathbb{F}_q^n$ of dimension k is called a* linear $[n, k]$ code *and is usually denoted by C. The elements of the code are called* (code)words.

A code can be given by writing down all elements, but because the code is a linear subspace, it has a basis.

Definition 1.2.2 *The* generator matrix *of a linear $[n,k]$ code C is a $k \times n$ matrix of full rank over $\mathbb{F}_q$ whose rows form a basis of C.*

Note that this matrix is not unique. We can rewrite the definition of a code in terms of the generator matrix:

$$C = \{\mathbf{x}G : \mathbf{x} \in \mathbb{F}_q^k\}.$$

A measurement for how much information is added by using a code, is the information rate:

Definition 1.2.3 *The* information rate *of a linear $[n,k]$ code is defined by $R = \frac{k}{n}$.*

We will assume all our codes to be non-degenerate: there are no coordinates which are zero for al codewords, i.e. the generator matrix does not contain any zero columns.

1.2.2 Weight distributions

For a word of a linear $[n,k]$ code C, the *(Hamming) weight* is the number of non-zero coordinates of the word. So the zero word has weight 0, and the maximum possible weight is n, the length of the code. The *(Hamming) distance* between two codewords of C is the number of coordinates where the words differ. The minimum of all non-zero distances between codewords is called the *minimum distance*. Because C is assumed to be linear, this is equal to the minimum non-zero weight of the code.
We summarize all this in the following definition:

Definition 1.2.4 *Let C be a linear $[n,k]$ code and $\mathbf{x},\mathbf{y} \in \mathbb{F}_q^n$. Then we define*

$$\mathrm{wt}(\mathbf{x}) = |\{i : x_i \neq 0\}|,$$

$$d(\mathbf{x},\mathbf{y}) = |\{i : x_i \neq y_i\}|,$$

$$d = \min\{d(\mathbf{x},\mathbf{y}) : \mathbf{x},\mathbf{y} \in C, \mathbf{x} \neq \mathbf{y}\} = \min\{\mathrm{wt}(\mathbf{x}) : \mathbf{x} \in C, \mathbf{x} \neq \mathbf{0}\}.$$

The number of codewords $\mathbf{c} \in C$ with $\mathrm{wt}(\mathbf{c}) = w$ is denoted by A_w. Note that $A_0 = 1$ and that d is the smallest $w > 0$ for which $A_w > 0$. The numbers A_w for all $0 \leq w \leq n$ form the *weight distribution* of the code. They also form the coefficients of the *weight enumerator*:

Definition 1.2.5 *The weight enumerator of a linear $[n,k]$ code C is the polynomial*

$$W_C(X,Y) = \sum_{w=0}^{n} A_w X^{n-w} Y^w,$$

where $A_w = |\{\mathbf{c} \in C : \mathrm{wt}(\mathbf{c}) = w\}|$.

Another way to define the weight enumerator is

$$W_C(X,Y) = \sum_{\mathbf{c} \in C} X^{n-\mathrm{wt}(\mathbf{c})} Y^{\mathrm{wt}(\mathbf{c})}.$$

We will always use this homogeneous form of the weight enumerator. There is also the one-variable form, $W_C(Z)$, which is connected to the homogeneous form via $W_C(X,Y) = X^n W_C(YX^{-1})$ and $W_C(Z) = W_C(1,Z)$.

We can generalize the weight distribution in the following way. Instead of looking at words of C, we consider all the subcodes of C of a certain dimension r. We say that the *weight of a subcode* is equal to n minus the number of coordinates which are zero for every word in the subcode. The smallest weight for which a subcode of dimension r exists, is called the *r-th generalized Hamming weight* of C. To summarize:

Definition 1.2.6 *Let D be an r-dimensional subcode of the $[n,k]$ code C. Then we define*

$$\mathrm{wt}(D) = |\{i \in [n] : \exists \mathbf{x} \in D : x_i \neq 0\}|,$$

$$d_r = \min\{\mathrm{wt}(D) : D \subseteq C \textit{ subcode}, \dim D = r\}.$$

Note that $d_0 = 0$ and $d_1 = d$, the minimum distance of the code. The number of subcodes with a given weight w and dimension r is denoted by A_w^r. Together they form the *r-th generalized weight distribution* of the code. Just as with the ordinary weight distribution, we can make a polynomial with the distribution as coefficients: the *generalized weight enumerator.*

Definition 1.2.7 *The generalized weight enumerator is given by*

$$W_C^r(X,Y) = \sum_{w=0}^{n} A_w^r X^{n-w} Y^w,$$

where $A_w^r = |\{D \subseteq C : \dim D = r, \mathrm{wt}(D) = w\}|$.

We can see from this definition that $A_0^0 = 1$ and $A_0^r = 0$ for all $0 < r \leq k$. Furthermore, every 1-dimensional subspace of C contains $q-1$ non-zero codewords, so $(q-1)A_w^1 = A_w$ for $0 < w \leq n$.

If confusion about the underlying code is possible, we denote the code between brackets behind the invariant: for example $d_r(C)$ or $A_w(C^\perp)$.

1.2.3 More about codes

Let $< , >$ be the inner product on $\mathbb{F}_q^n$ given by the symmetric linear form $< \mathbf{x}, \mathbf{y} >= \sum_{i=1}^n x_i y_i$. Then the *dual code* of a linear $[n, k]$ code C over $\mathbb{F}_q$ is the subspace of $\mathbb{F}_q^n$ orthogonal to C with respect to $< , >$.

Definition 1.2.8 *Let C be a linear $[n, k]$ code over $\mathbb{F}_q$. Then the dual code is*

$$C^\perp = \{\mathbf{x} \in \mathbb{F}_q^n : < \mathbf{x}, \mathbf{c} >= 0 \text{ for all } \mathbf{c} \in C\}.$$

It is clear that the dual code is again a linear code of length n over $\mathbb{F}_q$ and has dimension $n-k$. Furthermore, $(C^\perp)^\perp = C$.

There are two important bounds on the parameters of a code which we are going to use. For the proofs we refer to [8], [10] or any other course in basic coding theory.

Theorem 1.2.9 (Singleton bound) *Let C be a linear $[n, k]$ code over $\mathbb{F}_q$. Then $d \leq n - k + 1$.* □

Definition 1.2.10 (MDS code) *A linear $[n, k]$ code C is called* maximum distance separable *if it achieves the Singleton bound, so if $d = n - k + 1$.*

Theorem 1.2.11 (Gilbert–Varshamov bound) *Fix integers q, n, d. Let k be the smallest integer satisfying*

$$q^k \geq \frac{q^n}{\sum_{i=0}^{d-1} \binom{n}{i} (q-1)^i}.$$

Then there exists a linear $[n, k]$ code over $\mathbb{F}_q$ with minimum distance d. □

We will now define what it means for two codes to be equivalent. There are several ways to do this. The most easy one is to call two linear $[n, k]$ codes over $\mathbb{F}_q$ equivalent if they are equal, i.e. if their row-space in $\mathbb{F}_q^n$ is the same. We are giving a more general definition, in order to let equivalent codes coincide with equivalent matroids and geometries.

Definition 1.2.12 *Two linear $[n, k]$ codes over $\mathbb{F}_q$ are called equivalent if their generator matrices are the same up to*

- *left multiplication with an invertible $k \times k$ matrix over $\mathbb{F}_q$;*
- *permutation of the columns;*
- *multiplication of columns with an element of $\mathbb{F}_q^*$.*

Note that two equivalent codes have the same (generalized) weight distribution.

1.2.4 Matroids

Matroid theory generalizes the notion of 'independence'. We will use it for matrices over finite fields, but it also has important applications in other branches of combinatorics such as graph theory. There are many ways to define a matroid: we will use a definition that makes it easy to see that the generator matrix of a linear code gives rise to a matroid.

Definition 1.2.13 *A matroid G is a pair (S, I) where S is a finite set and I is a collection of subsets of S called the independent sets, satisfying the following properties:*

(i) The empty set is independent.

(ii) Every subset of an independent set is independent.

(iii) Let A and B be two independent sets with $|A| > |B|$, then there exists an $a \in A$ with $a \notin B$ and $B \cup \{a\}$ an independent set.

A subset of S which is not independent, is called *dependent*. An independent subset of S for which adding an extra element of S always gives a dependent subset, is a maximal independent set or a *basis*. Just as in linear algebra, it can be showed that every basis has the same number of elements. This is called the *rank* of the matroid. We define the rank of a subset of S to be the size of the largest independent set contained in it. For notation:

Definition 1.2.14 *The rank $r(A)$ of a matroid (S, I) and its set $\mathcal{B}$ of bases are defined by*

$$r(A) = \max\{|A'| : A' \subseteq A, A' \in I\}$$

$$\mathcal{B} = \{B \subseteq S : r(B) = |B| = r(S)\}$$

The rank function and the set of bases can each be used to determine a matroid completely. Therefore we can define the *dual* of a matroid in the following way:

Definition 1.2.15 *Let $G = (S, \mathcal{B})$ be a matroid defined by its set of bases. Then its dual is the matroid $G^* = (S, \mathcal{B}^*)$ with the same underlying set and set of bases*

$$\mathcal{B}^* = \{S - B : B \in \mathcal{B}\}.$$

Note the similarity with the definition of a dual code. For a matroid we also have $(G^*)^* = G$.

1.2.5 Gaussian binomials and other products

Definition 1.2.16 *We introduce the following notations:*

$$[m, r]_q = \prod_{i=0}^{r-1}(q^m - q^i)$$

$$\langle r \rangle_q = [r, r]_q$$

$$\begin{bmatrix} k \\ r \end{bmatrix}_q = \frac{[k, r]_q}{\langle r \rangle_q}.$$

The first number is equal to the number of $m \times r$ matrices of rank r over $\mathbb{F}_q$. The second is the number of bases of $\mathbb{F}_q^r$. The third number is the Gaussian binomial, and it represents the number of r-dimensional subspaces of $\mathbb{F}_q^k$. The following useful relation can easily be verified from the definitions:

$$[m, r]_q = \frac{q^{-r(m-r)}\langle m \rangle_q}{\langle m - r \rangle_q}.$$

1.3 Notes

In the last section of a chapter, we will recall whom the material in the chapter was due to. Blahut [1] gives more applications of the weight enumerator. An introduction to coding theory can be found in [10] and [8]. More about matroid theory can be found in [9]. Kløve [6] and Wei [15] were the first to introduce the generalized Hamming weights. The notations for products and their relation to the Gaussian binomial comes from Kløve [7].

Chapter 2

Weight enumerator

2.1 Generalized weight enumerator

We will give a way to determine the generalized weight enumerator of a linear $[n,k]$ code C over $\mathbb{F}_q$.

Definition 2.1.1 *For $J \subseteq [n]$ and we define:*

$$\begin{aligned} t &= |J| \\ C(J) &= \{\mathbf{c} \in C : c_j = 0 \text{ for all } j \in J\} \\ l(J) &= \dim C(J) \end{aligned}$$

We first give two lemmas about the determination of $l(J)$, which will become useful later.

Lemma 2.1.2 *Let C be a linear code with generator matrix G. Let G' be the $k \times t$ submatrix of G existing of the columns of G indexed by J, and let $r(J)$ be the rank of G'. Then the dimension $l(J)$ is equal to $k - r(J)$.*

Proof: Let C' be the code generated by G'. Consider C' as a subcode of C, so a word of C' has zeros on the coordinates not indexed by J. Then we have $C' \cong C/C(J)$. It follows that $\dim C' = \dim C - \dim C(J)$ so $l(J) = k - r(J)$. □

Lemma 2.1.3 *Let d and $d^\perp$ be the minimum distance of C and $C^\perp$ respectively, and let $J \subseteq [n]$. Then we have*

$$l(J) = \begin{cases} k - t & \text{for all } t < d^\perp \\ 0 & \text{for all } t > n - d \end{cases}$$

Proof: Let $|J| = t$, $t > n - d$ and let $\mathbf{c} \in C(J)$. Then J is contained in the complement of supp($\mathbf{c}$), so $t \leq n - \mathrm{wt}(\mathbf{c})$. It follows that $\mathrm{wt}(\mathbf{c}) \leq n - t < d$, so $\mathbf{c}$ is the zero word and therefore $l(J)$=0.
Let G be a generator matrix for C, then G is also a parity check matrix for $C^{\perp}$. We saw in lemma 2.1.2 that $l(J) = k - r(J)$, where $r(J)$ is the rank of the matrix formed by the columns of G indexed by J. Let $t < d^{\perp}$, then every t-tuple of columns of G is linearly independent, so $r(J) = t$ and $l(J) = k - t$. □

Note that by the Singleton bound, we have $d^{\perp} \leq n - (n-k) + 1 = k+1$ and $n - d \geq k - 1$, so for $t = k$ both of the above cases apply. This is no problem, because if $t = k$ then $k - t = 0$.

Definition 2.1.4 *For $J \subseteq [n]$ and $r \geq 0$ an integer we define:*

$$\begin{aligned} B_J^r &= |\{D \subseteq C(J) : D \textit{ subspace of dimension } r\}| \\ B_t^r &= \sum_{|J|=t} B_J^r \end{aligned}$$

Note that $B_J^r = \begin{bmatrix} l(J) \\ r \end{bmatrix}_q$, the Gaussian binomial. For $r = 0$ this gives $B_t^0 = \binom{n}{t}$. So we see that in general $l(J) = 0$ does not imply $B_J^r = 0$, because $\begin{bmatrix} 0 \\ 0 \end{bmatrix}_q = 1$. But if $r \neq 0$, we do have that $l(J) = 0$ implies $B_J^r = 0$ and $B_t^r = 0$.

Proposition 2.1.5 *Let d_r be the r-th generalized Hamming weight of C, and $d^{\perp}$ the minimum distance of the dual code $C^{\perp}$. Then we have*

$$B_t^r = \begin{cases} \binom{n}{t} \begin{bmatrix} k-t \\ r \end{bmatrix}_q & \textit{for all } t < d^{\perp} \\ 0 & \textit{for all } t > n - d_r \end{cases}$$

Proof: The first case is is a direct corollary of lemma 2.1.3, since there are $\binom{n}{t}$ subsets $J \subseteq [n]$ with $|J| = t$. The proof of the second case goes analogous to the proof of the same lemma: let $|J| = t$, $t > n - d_r$ and suppose there is a subspace $D \subseteq C(J)$ of dimension r. Then J is contained in the complement of supp(D), so $t \leq n - \mathrm{wt}(D)$. It follows that $\mathrm{wt}(D) \leq n - t < d_r$, which is impossible, so such a D does not exist. So $B_J^r = 0$ for all J with $|J| = t$ and $t > n - d_r$, and therefore $B_t^r = 0$ for $t > n - d_r$. □

We can check that the formula is well-defined: if $t < d^{\perp}$ then $l(J) = k - t$. If also $t > n - d_r$, we have $t > n - d_r \geq k - r$ by the generalized Singleton bound. This implies $r > k - t = l(J)$, so $\begin{bmatrix} k-t \\ r \end{bmatrix}_q = 0$.
The relation between B_t^r and A_w^r becomes clear in the next proposition.

Proposition 2.1.6 *The following formula holds:*

$$B_t^r = \sum_{w=0}^{n} \binom{n-w}{t} A_w^r.$$

Proof: We will count the elements of the set

$$\mathcal{B}_t^r = \{(D,J) : J \subseteq [n], |J| = t, D \subseteq C(J) \text{ subspace of dimension } r\}$$

in two different ways. For each J with $|J| = t$ there are B_J^r pairs (D,J) in $\mathcal{B}_t^r$, so the total number of elements in this set is $\sum_{|J|=t} B_J^r = B_t^r$. On the other hand, let D be an r-dimensional subcode of C with $\mathrm{wt}(D) = w$. There are A_w^r possibilities for such a D. If we want to find a J such that $D \subseteq C(J)$, we have to pick t coordinates from the $n-w$ all-zero coordinates of D. Summation over all w proves the given formula. □

Note that because $A_w^r = 0$ for all $w < d_r$, we can start summation at $w = d_r$. We can end summation at $w = n-t$ because for $t > n-w$ we have $\binom{n-w}{t} = 0$. So the formula can be rewritten as

$$B_t^r = \sum_{w=d_r}^{n-t} \binom{n-w}{t} A_w^r.$$

In practice, we will often prefer the summation given in the proposition.

Theorem 2.1.7 *The generalized weight enumerator is given by the following formula:*

$$W_C^r(X,Y) = \sum_{t=0}^{n} B_t^r (X-Y)^t Y^{n-t}.$$

Proof: By using the previous proposition, changing the order of summation and using the binomial expansion of $X^{n-w} = ((X-Y)+Y)^{n-w}$ we have

$$\begin{aligned}
\sum_{t=0}^{n} B_t^r (X-Y)^t Y^{n-t} &= \sum_{t=0}^{n} \sum_{w=0}^{n} \binom{n-w}{t} A_w^r (X-Y)^t Y^{n-t} \\
&= \sum_{w=0}^{n} A_w^r \left(\sum_{t=0}^{n-w} \binom{n-w}{t} (X-Y)^t Y^{n-w-t} \right) Y^w \\
&= \sum_{w=0}^{n} A_w^r X^{n-w} Y^w \\
&= W_C^r(X,Y).
\end{aligned}$$

In the second step, we can let the summation over t run to $n-w$ instead of n because $\binom{n-w}{t}=0$ for $t>n-w$. □

It is possible to determine the A_w^r directly from the B_t^r, by using the next proposition.

Proposition 2.1.8 *The following formula holds:*

$$A_w^r = \sum_{t=n-w}^{n} (-1)^{n+w+t} \binom{t}{n-w} B_t^r.$$

There are several ways to prove this proposition. One is to reverse the argument from Theorem 2.1.7, which we will not use here. Instead, we first prove the following general lemma:

Lemma 2.1.9 *Let V be a vector space of dimension $n+1$ and let $\mathbf{a}=(a_0,\ldots,a_n)$ and $\mathbf{b}=(b_0,\ldots,b_n)$ be vectors in V. Then the following formulas are equivalent:*

$$a_j = \sum_{i=0}^{n} \binom{i}{j} b_i, \qquad b_j = \sum_{i=j}^{n} (-1)^{i+j} \binom{i}{j} a_i.$$

Proof: We can view the relations between $\mathbf{a}$ and $\mathbf{b}$ as linear transformations, given by the matrices $\left(\binom{i}{j}\right)_{i,j=0,\ldots,n}$ and $\left((-1)^{i+j}\binom{i}{j}\right)_{i,j=0,\ldots,n}$. So it is sufficient to prove that this matrices are each other's inverse. We calculate the entry on the i-th row and j-th column. Note that we can start the summation at $l=j$, because for $l<j$ we have $\binom{l}{j}=0$.

$$\begin{aligned}
\sum_{l=j}^{i} (-1)^{j+l} \binom{i}{l}\binom{l}{j} &= \sum_{l=j}^{i} (-1)^{l-j} \binom{i}{j}\binom{i-j}{l-j} \\
&= \sum_{l=0}^{i-j} (-1)^{l} \binom{i}{j}\binom{i-j}{l} \\
&= \binom{i}{j}(1-1)^{i-j} \\
&= \delta_{ij}.
\end{aligned}$$

Here δ_{ij} is the Kronecker-delta. So the product matrix is exactly the $(n+1)\times(n+1)$ identity matrix, and therefore the matrices are each other's inverse. □

Proof of Proposition 2.1.8: The Proposition is now a direct consequence of Proposition 2.1.6 and Lemma 2.1.9. □

As already noticed in the definitions, we can find back the original weight enumerator by using $W_C(X,Y) = W_C^0(X,Y) + (q-1)W_C^1(X,Y)$.

2.2 Extended weight enumerator

Let C be an $[n,k]$ code over $\mathbb{F}_q$ with generator matrix G. Then we can form the $[n,k]$ code $C \otimes \mathbb{F}_{q^m}$ over $\mathbb{F}_{q^m}$ with the same generator matrix G. We call this the *extension code* of C over $\mathbb{F}_{q^m}$ and denote its weight distribution by $A_w(C \otimes \mathbb{F}_{q^m})$. (In fact, we may do this for every field, not necessarily finite.) We can determine the weight enumerator of such an extension code by using only the code C, which makes the determination much easier and faster.
In Lemma 2.1.2 we saw that $l(J) = k - r(J)$, where $r(J)$ is the rank of the $k \times t$ matrix G', which consists of the columns of G indexed by J. Because $r(J)$ is just the number of pivots in G', it is independent of the extension field $\mathbb{F}_{q^m}$. Hence $\dim_{\mathbb{F}_q} C(J) = \dim_{\mathbb{F}_{q^m}} (C \otimes \mathbb{F}_{q^m})(J)$. This motivates the usage of T as a variable for q^m in the next definition.

Definition 2.2.1 *Let C be a linear code over $\mathbb{F}_q$. Then we define*

$$B_J(T) = T^{l(J)} - 1$$

$$B_t(T) = \sum_{|J|=t} B_J(T)$$

The extended weight enumerator *is given by*

$$W_C(X,Y,T) = X^n + \sum_{t=0}^{n} B_t(T)(X-Y)^t Y^{n-t}.$$

Note that $B_J(q^m)$ is the number of nonzero codewords in $(C \otimes \mathbb{F}_{q^m})(J)$.

Proposition 2.2.2 *Let d and $d^\perp$ be the minimum distance of C and $C^\perp$ respectively. Then we have*

$$B_t(T) = \begin{cases} \binom{n}{t}(T^{k-t} - 1) & \text{for all } t < d^\perp \\ 0 & \text{for all } t > n - d \end{cases}$$

Proof: This is a direct consequence of Lemma 2.1.3. For $t < d^\perp$ we have $l(J) = k - t$, so $B_J(T) = T^{k-t} - 1$ and $B_t(T) = \binom{n}{t}(T^{k-t} - 1)$. For $t > n - d$ we have $l(J) = 0$, so $B_J(T) = 0$ and $B_t(T) = 0$. □

Theorem 2.2.3 *The following holds:*

$$W_C(X,Y,T) = \sum_{w=0}^{n} A_w(T) X^{n-w} Y^w$$

with $A_w(T) \in \mathbb{Z}[T]$ *given by* $A_0(T) = 1$ *and*

$$A_w(T) = \sum_{t=n-w}^{n} (-1)^{n+w+t} \binom{t}{n-w} B_t(T)$$

for $0 < w \leq n$.

Proof: Note that $A_w(T) = 0$ for $0 < w < d$ because the summation is empty. By substituting $w = n - t + j$ and reversing the order of summation, we have

$$\begin{aligned}
W_C(X,Y,T) &= X^n + \sum_{t=0}^{n} B_t(T)(X-Y)^t Y^{n-t} \\
&= X^n + \sum_{t=0}^{n} B_t(T) \left(\sum_{j=0}^{t} \binom{t}{j} (-1)^j X^{t-j} Y^j \right) Y^{n-t} \\
&= X^n + \sum_{t=0}^{n} \sum_{j=0}^{t} (-1)^j \binom{t}{j} B_t(T) X^{t-j} Y^{n-t+j} \\
&= X^n + \sum_{t=0}^{n} \sum_{w=n-t}^{n} (-1)^{t-n+w} \binom{t}{t-n+w} B_t(T) X^{n-w} Y^w \\
&= X^n + \sum_{w=0}^{n} \sum_{t=n-w}^{n} (-1)^{n+w+t} \binom{t}{n-w} B_t(T) X^{n-w} Y^w
\end{aligned}$$

Hence $W_C(X,Y,T)$ is of the form $\sum_{w=0}^{n} A_w(T) X^{n-w} Y^w$ with $A_w(T)$ of the form given in the theorem. □

Note that in the definition of $A_w(T)$ we can let the summation over t run to $n-d$ instead of n, because $B_t(T) = 0$ for $t > n-d$.

Proposition 2.2.4 *The following formula holds:*

$$B_t(T) = \sum_{w=d}^{n-t} \binom{n-w}{t} A_w(T).$$

Proof: The statement is a direct consequence of Lemma 2.1.9 and Theorem 2.2.3. □

As we said before, the motivation for looking at the extended weight enumerator comes from the extensioncodes. In the next proposition we show that the extended weight enumerator for $T = q^m$ is indeed the weight enumerator of the extensioncode $C \otimes \mathbb{F}_{q^m}$.

Proposition 2.2.5 *Let C be a linear $[n, k]$ code over $\mathbb{F}_q$. Then $W_C(X, Y, q^m) = W_{C\otimes\mathbb{F}_{q^m}}(X, Y)$.*

Proof: For $w = 0$ it is clear that $A_0(q^m) = A_0(C \otimes \mathbb{F}_{q^m}) = 1$, so assume $w \neq 0$. It is enough to show that $A_w(q^m) = (q^m - 1)A^1_w(C \otimes \mathbb{F}_{q^m})$. First we have

$$\begin{aligned} B_t(q^m) &= \sum_{|J|=t} B_J(q^m) \\ &= \sum_{|J|=t} |\{\mathbf{c} \in (C \otimes \mathbb{F}_{q^m})(J) : \mathbf{c} \neq 0\}| \\ &= (q^m - 1) \sum_{|J|=t} |\{D \subseteq (C \otimes \mathbb{F}_{q^m})(J) : \dim D = 1\} \\ &= (q^m - 1)B^1_t(C \otimes \mathbb{F}_{q^m}). \end{aligned}$$

We also know that $A_w(T)$ and $B_t(T)$ are related the same way as A^1_w and B^1_t. Combining this proves the statement. □

For further applications, the next way of writing the extended weight enumerator will be useful:

Proposition 2.2.6 *The extended weight enumerator of a linear code C can be written as*

$$W_C(X, Y, T) = \sum_{t=0}^{n} \sum_{|J|=t} T^{l(J)}(X - Y)^t Y^{n-t}.$$

Proof: By rewriting and using the binomial expansion of $((X - Y) + Y)^n$, we get

$$\sum_{t=0}^{n} \sum_{|J|=t} T^{l(J)}(X - Y)^t Y^{n-t}$$

$$\begin{aligned}
&= \sum_{t=0}^{n}(X-Y)^t Y^{n-t} \sum_{|J|=t} \left((T^{l(J)}-1)+1\right) \\
&= \sum_{t=0}^{n}(X-Y)^t Y^{n-t} \left(\sum_{|J|=t}(T^{l(J)}-1) + \binom{n}{t} \right) \\
&= \sum_{t=0}^{n} B_t(T)(X-Y)^t Y^{n-t} + \sum_{t=0}^{n} \binom{n}{t}(X-Y)^t Y^{n-t} \\
&= \sum_{t=0}^{n} B_t(T)(X-Y)^t Y^{n-t} + X^n \\
&= W_C(X,Y,T)
\end{aligned}$$

□

2.3 Another algorithm

We can determine the extended weight enumerator of a $[n,k]$ code C with the use of a $k \times n$ generator matrix of C. This concept can be generalized for arbitrarily matrices, not necessarily of full rank. With the help of the following definition, we will give another way to determine the extended weight enumerator.

Definition 2.3.1 *Let G be an $k \times n$ matrix over $\mathbb{F}_q$, not necessarily of full rank and without zero columns. Then for each $J \subseteq [n]$ we define $l(J) = k - r(J)$ as in Lemma 2.1.2, and the extended weight enumerator $W_G(X,Y,T)$ as in Definition 2.2.1.*

We can now make the following remarks about $W_G(X,Y,T)$.

Proposition 2.3.2 *Let G be a $k \times n$ matrix over $\mathbb{F}_q$ and $W_G(X,Y,T)$ the associated extended weight enumerator. Then the following statements hold:*

(i) $W_G(X,Y,T)$ is invariant under row-equivalence of matrices.

(ii) Let G' be a $l \times n$ matrix with the same row-space as G, then we have $W_G(X,Y,T) = T^{k-l}W_{G'}(X,Y,T)$. In particular, if G is a generator matrix of a $[n,k]$ code C, we have $W_G(X,Y,T) = W_C(X,Y,T)$.

(iii) $W_G(X,Y,T)$ is invariant under permutation of the columns of G.

(iv) $W_G(X,Y,T)$ is invariant under multiplying a column of G with an element of $\mathbb{F}_q^*$.

(v) If G is the direct sum of G_1 and G_2, i.e. of the form

$$\begin{pmatrix} G_1 & 0 \\ 0 & G_2 \end{pmatrix},$$

then $W_G(X,Y,T) = W_{G_1}(X,Y,T) \cdot W_{G_2}(X,Y,T)$.

Proof: If we multiply G from the left with an invertible $k \times k$ matrix, the $r(J)$ do not change, and therefore (i) holds. For (ii), we may assume without loss of generality that $k \geq l$. Because G and G' have the same row-space, the $r(J)$ are the same. Using Proposition 2.2.6 we have for G

$$\begin{aligned} W_G(X,Y,T) &= \sum_{t=0}^{n} \sum_{|J|=t} T^{l(J)}(X-Y)^t Y^{n-t} \\ &= \sum_{t=0}^{n} \sum_{|J|=t} T^{k-r(J)}(X-Y)^t Y^{n-t} \\ &= T^{k-l} \sum_{t=0}^{n} \sum_{|J|=t} T^{l-r(J)}(X-Y)^t Y^{n-t} \\ &= T^{k-l} W_{G'}(X,Y,T). \end{aligned}$$

The last part of (ii) and (iii)–(v) follow directly from the definitions. □

With the use of the extended weight enumerator for general matrices, we can derive a recursive algorithm to determine the extended weight enumerator of a code. If G is a $k \times n$ matrix, we denote by G^* a matrix which is row-equivalent to G and has a column a of the form $(1,0,\ldots,0)^T$. In general, this reduction G^* is not unique.
The matrix $G^* - a$ is the $k \times (n-1)$ matrix G^* with the column a removed, and G^*/a is the $(k-1) \times (n-1)$ matrix G^* with the column a and the first row removed. For the extended weight enumerators of these matrices, we have the following connection (we omit the (X,Y,T) part for clarity):

Proposition 2.3.3 *For the extended weight enumerator of a reduced matrix G^* holds*

$$W_{G^*} = (X-Y)W_{G^*/a} + YW_{G^*-a}$$

Proof: We distinguish between two cases here. First, assume that $G^* - a$ and $G * /a$ have the same rank. Then we can choose a G^* with all zeros in the first row, except for the 1 in the column a. So G^* is the direct sum of 1 and G^*/a. By Proposition 2.3.2 parts (v) and (ii) we have

$$W_{G^*} = (X+(T-1)Y)W_{G^*/a} \qquad \text{and} \qquad W_{G^*-a} = TW_{G^*/a}.$$

Combining the two gives

$$\begin{aligned} W_{G^*} &= (X+(T-1)Y)W_{G^*/a} \\ &= (X-Y)W_{G^*/a} + YTW_{G^*/a} \\ &= (X-Y)W_{G^*/a} + YW_{G^*-a}. \end{aligned}$$

For the second case, assume that G^*-a and G^*/a do not have the same rank. This implies G^* and G^*-a do have the same rank. The vectors in $\{\mathbf{x}G^* : \mathbf{x} \in \mathbb{F}_{q^m}\}$ now fall into two cases: those which have a zero on position a, and those which do not. The first have extended weight distribution equal to $XW_{G*/a}(X,Y,q^m)$. The second are the vectors in $\{\mathbf{x}(G^*-a) : \mathbf{x} \in \mathbb{F}_{q^m}\}$ but not in $\{\mathbf{x}(G^*/a) : \mathbf{x} \in \mathbb{F}_{q^m}\}$, with a single nonzero coordinate added. So we have

$$W_{G^*} = XW_{G^*/a}(X,Y,q^m) + Y(W_{G^*-a}(X,Y,q^m) - W_{G^*/a}(X,Y,q^m)).$$

Changing to T by Lagrange interpolation proves the given formula. □

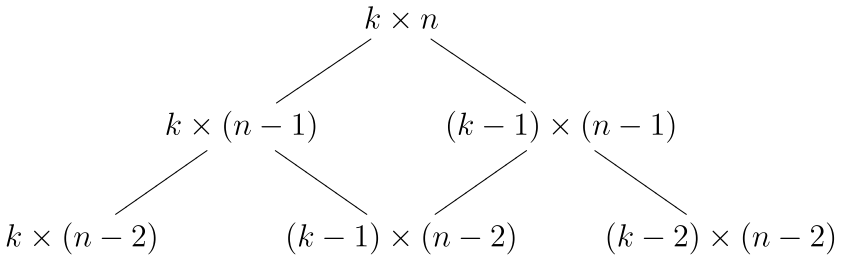
Theorem 2.3.4 *Let G be a $k \times n$ matrix over $\mathbb{F}_q$ with $n > k$ of the form $G^* = (I_k|P)$, where P is a $k \times (n-k)$ matrix over $\mathbb{F}_q$. Let $A \subseteq [k]$ and write P_A for the matrix formed by the rows of P indexed by A. Let $W_A(X,Y,T) = W_{P_A}(X,Y,T)$. Then the following holds:*

$$W_C(X,Y,T) = \sum_{l=0}^{k} \sum_{|A|=l} Y^l (X-Y)^{k-l} W_A(X,Y,T).$$

Proof: We use the formula of the last proposition recursively. We denote the construction of G^*-a by G_1 and the construction of G^*/a by G_2. Repeating this procedure, we get the matrices G_{11}, G_{12}, G_{21} and G_{22}. So we get for the weight enumerator

$$W_G = Y^2 W_{G_{11}} + Y(X-Y)W_{G_{12}} + Y(X-Y)W_{G_{21}} + (X-Y)^2 W_{G_{22}}.$$

Repeating this procedure k times, we get 2^k matrices with $n-k$ columns and $0,\ldots,k$ rows, which form exactly the P_A. In the diagram are the sizes of the matrices of the first two steps: note that only the $k \times n$ matrix on top has to be of full rank. The number of matrices of size $(k-i) \times (n-j)$ are given by the binomial coefficient $\binom{j}{i}$.



On the last line we have $W_0(X,Y,T) = X^{n-k}$. This proves the formula. □

We illustrate the working of the above theorem by an example. Let C be the even weight code of length $n = 6$ over $\mathbb{F}_2$. Then a generator matrix of C is the 5×6 matrix $G = (I_5|P)$ with $P = (1,1,1,1,1,1)^T$. So the matrices P_A are $l \times 1$ matrices with all ones. We have $W_0(X,Y) = X$ and $W_l(X,Y) = T^{l-1}(X + (T-1)Y)$ by part (ii) of Proposition 2.3.2. Therefore the weight enumerator of C is equal to

$$\begin{aligned}
W_C(X,Y) &= W_G(X,Y) \\
&= X(X-Y)^5 + \sum_{l=1}^{5} \binom{5}{l} Y^l (X-Y)^{5-l} T^{l-1}(X+(T-1)Y) \\
&= X^6 + 15(T-1)X^4Y^2 + 20(T^2-3T+2)X^3Y^3 \\
&\quad +15(T^3-4T^2+6T-3)X^2Y^4 \\
&\quad +6(T^4-5T^3+10T^2-10T+4)XY^5 \\
&\quad +(T^5-6T^4+15T^3-20T^2+15T-5)Y^6.
\end{aligned}$$

For $T = 2$ we get $W_C(X,Y,2) = X^6 + 15X^4Y^2 + 15X^2Y^4 + Y^6$, which we indeed recognise as the weight enumerator of the even weight code.

2.4 Generalized extended weight enumerator

Determining the generalized extended weight enumerator $W_C^r(X,Y,T)$ goes analogously to the determination of the extended weight enumerator. We will give the necessary definitions and theorems here, and leave the proofs to the reader. This polynomial contains no extra information with respect to $W_C^r(X,Y)$ and $W_C(X,Y,T)$, but is useful to determine both at once, or to determine $W_{C\otimes\mathbb{F}_{q^m}}^r(X,Y)$.

Definition 2.4.1 *Let C be a linear code over $\mathbb{F}_q$. Then we define*

$$B_J^r(T) = \begin{bmatrix} l(J) \\ r \end{bmatrix}_T$$

$$B_t^r(T) = \sum_{|J|=t} B_J^r(T)$$

The generalized extended weight enumerator *is given by*

$$W_C^r(X,Y,T) = \sum_{t=0}^{n} B_t^r(T)(X-Y)^t Y^{n-t}.$$

Proposition 2.4.2 *Let d and $d^\perp$ be the minimum distance of C and $C^\perp$ respectively. Then we have*

$$B_t^r(T) = \begin{cases} \binom{n}{t} \begin{bmatrix} k-t \\ r \end{bmatrix}_T & \text{for all } t < d^\perp \\ 0 & \text{for all } t > n - d_r \end{cases}$$

Theorem 2.4.3 *The following holds:*

$$W_C^r(X,Y,T) = \sum_{w=0}^{n} A_w^r(T) X^{n-w} Y^w$$

with $A_w^r(T) \in \mathbb{Z}[T]$ given by $A_0^r(T) = \delta_{0r}$ and

$$A_w^r(T) = \sum_{t=n-w}^{n} (-1)^{n+w+t} \binom{t}{n-w} B_t^r(T)$$

for $0 < w \leq n$.

Proposition 2.4.4 *The following formula holds:*

$$B_t^r(T) = \sum_{w=d_r}^{n-t} \binom{n-w}{t} A_w^r(T).$$

Proposition 2.4.5 *Let C be a linear $[n,k]$ code over $\mathbb{F}_q$. Then we have $W_C^r(X,Y,q^m) = W_{C\otimes\mathbb{F}_{q^m}}^r(X,Y)$.*

Proof: For $w = 0$ it is clear that $A_0^r(q^m) = A_0^r(C \otimes \mathbb{F}_{q^m})$, so assume $w \neq 0$. It is enough to show that $A_w^r(q^m) = A_w^r(C \otimes \mathbb{F}_{q^m})$. Let V be a linear vector space over $\mathbb{F}_{q^m}$ with $\dim V = l(J)$, then we have

$$\begin{aligned} B_J^r(q^m) &= |\{U \subseteq V : \dim U = r\}| \\ &= |\{D \subseteq (C \otimes \mathbb{F}_{q^m})(J) : \dim D = r\} \\ &= B_J^r(C \otimes \mathbb{F}_{q^m}). \end{aligned}$$

This implies that $B_t^r(q^m) = B_t^r(C \otimes \mathbb{F}_{q^m})$. We also know that $A_w^r(T)$ and $B_t^r(T)$ are related the same way as A_w^r and B_t^r. Combining this proves the statement. □

2.5 Notes

The determination of the generalized and extended weight enumerator is a generalisation of the method used by Pellikaan, Wu and Bulygin [10]. Lemma 2.1.9 and many other binomial identities can be found in Riordan [11]. Note also the similarity with Simonis [12]: for example, Lemma 2.1.2 is similar with proposition (i) from this article. The form $\sum_{t=0}^{n} B_t(X-Y)^t Y^{n-t}$ for the weight enumerator was first introduced in [5] and later in [13]. In the next chapters we will see more advantages of using this description of the weight enumerator.

The algorithm in section 2.3 is new material, and based on Tutte-Grothendieck decomposition of matrices. Greene [4] first used this decomposition for the determination of the weight enumerator. Proposition 2.3.3 is a generalization of his proposition 4.3.

Chapter 3

Connections

3.1 Extended in terms of generalized

Proposition 3.1.1 *Let C be a linear $[n,k]$ code over $\mathbb{F}_q$, and let C^m be the linear subspace consisting of the $m \times n$ matrices over $\mathbb{F}_q$ whose rows are in C. Then there is an isomorphism of $\mathbb{F}_q$-vector spaces between $C \otimes \mathbb{F}_{q^m}$ and C^m.*

Proof: Let α be a primitive m-th root of unity in $\mathbb{F}_{q^m}$. Then we can write an element of $\mathbb{F}_{q^m}$ in an unique way on the basis $(1, \alpha, \alpha^2, \ldots, \alpha^{m-1})$ with coefficients in $\mathbb{F}_q$. If we do this for all the coordinates of a word in $C \otimes \mathbb{F}_{q^m}$, we get an $m \times n$ matrix over $\mathbb{F}_q$. The rows of this matrix are words of C, because C and $C \otimes \mathbb{F}_{q^m}$ have the same generator matrix. This map is clearly injective. There are $(q^m)^k = q^{km}$ words in $C \otimes \mathbb{F}_{q^m}$, and the number of elements of C^m is $(q^k)^m = q^{km}$, so our map is a bijection. It is given by

$$\left(\sum_{i=0}^{m-1} c_{i1}\alpha^i, \sum_{i=0}^{m-1} c_{i2}\alpha^i, \ldots, \sum_{i=0}^{m-1} c_{in}\alpha^i \right) \mapsto$$

$$\begin{pmatrix} c_{01} & c_{02} & c_{03} & \cdots & c_{0n} \\ c_{11} & c_{12} & c_{13} & \cdots & c_{1n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ c_{(m-1)1} & c_{(m-1)2} & c_{(m-1)3} & \cdots & c_{(m-1)n} \end{pmatrix}.$$

We see that the map is linear, so it gives an isomorphism $C \otimes \mathbb{F}_{q^m} \to C^m$. □

Note that this isomorphism is not unique, because it depends on the choice of a primitive element α.

Lemma 3.1.2 *Let* $\mathbf{c} \in C \otimes \mathbb{F}_{q^m}$ *and* $M \in C^m$ *the corresponding* $m \times n$ *matrix under a given isomorphism. Let* $D \subseteq C$ *the subcode generated by* M*. Then* $\mathrm{wt}(\mathbf{c}) = \mathrm{wt}(D)$.

Proof: If the j-th coordinate c_j of $\mathbf{c}$ is zero, then the j-th column of M consists of only zero's, because the representation of c_j on the basis $(1, \alpha, \alpha^2, \ldots, \alpha^{m-1})$ is unique. On the other hand, if the j-th column of M consists of all zeros, then c_j is also zero. Therefore $\mathrm{wt}(\mathbf{c}) = \mathrm{wt}(D)$. □

Proposition 3.1.3 *Let* C *be a linear code over* $\mathbb{F}_q$*. Then the weight numerator of an extension code and the generalized weight enumerators are connected via*

$$A_w(q^m) = \sum_{r=0}^{m} [m, r]_q A_w^r.$$

Proof: We count the number of words in $C \otimes \mathbb{F}_{q^m}$ of weight w in two ways, using the bijection of proposition 3.1.1. The first way is just $A_w(T = q^m)$, the left side of the equation. For the second way, note that every $M \in C^m$ generates a subcode of C whose weight is equal to the weight of the corresponding word in $C \otimes \mathbb{F}_{q^m}$. Fix this weight w and a dimension r: there are A_w^r subcodes of C of dimension r and weight w. Every such subcode is generated by an $r \times n$ matrix whose rows are words of C. Left multiplication by an $m \times r$ matrix of rank r gives an element of C^m which generates the same subcode of C, and all such elements of C^m are obtained this way. The number of $m \times r$ matrices of rank r is $[m, r]_q$, so summation over all dimensions r gives

$$A_w(q^m) = \sum_{r=0}^{k} [m, r]_q A_w^r.$$

We can let the summation run to m, because $A_w^r = 0$ for $r > k$ and $[m, r]_q = 0$ for $r > m$. This proves the given formula. □

In general, we have the following theorem.

Theorem 3.1.4 *Let* C *be a linear code over* $\mathbb{F}_q$*. Then the extended weight numerator and the generalized weight enumerator are connected via*

$$W_C(X, Y, T) = \sum_{r=0}^{k} \left(\prod_{j=0}^{r-1} (T - q^j) \right) W_C^r(X, Y).$$

Proof: If we know A_w^r for all r, we can determine $A_w(q^m)$ for every m. If we have $k+1$ values of m for which $A_w(q^m)$ is known, we can use Lagrange interpolation to find $A_w(T)$, for this is a polynomial in T of degree at most k. In fact, we have

$$A_w(T) = \sum_{r=0}^{k} \left(\prod_{j=0}^{r-1} (T - q^j) \right) A_w^r.$$

This formula has the right degree and is correct for $T = q^m$ for all integer values $m \geq 0$, so we know it must be the correct polynomial. Therefore the theorem follows. □

3.2 Generalized in terms of extended

In this section, we give the relation found in the previous section the other way round.

Theorem 3.2.1 *Let C be a linear code over $\mathbb{F}_q$. Then the generalized weight enumerator and the extended weight enumerator are connected via*

$$W_C^r(X,Y) = \frac{1}{\langle r \rangle_q} \sum_{j=0}^{r} \begin{bmatrix} r \\ j \end{bmatrix}_q (-1)^{r-j} q^{(r-j)(r-j-1)/2} \, W_C(X,Y,q^j).$$

Proof: We consider the generalized weight enumerator in terms of Proposition 2.2.6. Then rewriting gives the following:

$$\begin{aligned} W_C^r(X,Y) &= \sum_{t=0}^{n} B_t^r (X-Y)^t Y^{n-t} \\ &= \sum_{t=0}^{n} \sum_{|J|=t} \begin{bmatrix} l(J) \\ r \end{bmatrix}_q (X-Y)^t Y^{n-t} \\ &= \sum_{t=0}^{n} \sum_{|J|=t} \left(\prod_{j=0}^{r-1} \frac{q^{l(J)} - q^j}{q^r - q^j} \right) (X-Y)^t Y^{n-t} \\ &= \frac{1}{\prod_{v=0}^{r-1} (q^r - q^v)} \sum_{t=0}^{n} \sum_{|J|=t} \left(\prod_{j=0}^{r-1} (q^{l(J)} - q^j) \right) (X-Y)^t Y^{n-t} \\ &= \frac{1}{\langle r \rangle_q} \sum_{t=0}^{n} \sum_{|J|=t} \sum_{j=0}^{r} \begin{bmatrix} r \\ j \end{bmatrix}_q (-1)^{r-j} q^{\binom{r-j}{2}} q^{j \cdot l(J)} (X-Y)^t Y^{n-t} \end{aligned}$$

$$= \frac{1}{\langle r\rangle_q}\sum_{j=0}^{r}\begin{bmatrix} r \\ j \end{bmatrix}_q (-1)^{r-j}q^{\binom{r-j}{2}}\sum_{t=0}^{n}\sum_{|J|=t}(q^j)^{l(J)}(X-Y)^tY^{n-t}$$

$$= \frac{1}{\langle r\rangle_q}\sum_{j=0}^{r}\begin{bmatrix} r \\ j \end{bmatrix}_q (-1)^{r-j}q^{(r-j)(r-j-1)/2}\, W_C(X,Y,q^j)$$

In the fourth step, we use the following identity (see [6]), which can be proven by induction:

$$\prod_{j=0}^{r-1}(Z-q^j) = \sum_{j=0}^{r}\begin{bmatrix} r \\ j \end{bmatrix}_q (-1)^{r-j}q^{\binom{r-j}{2}}Z^j.$$

□

3.3 Weight enumerator and Tutte polynomial

Definition 3.3.1 *For a matroid G with rank function r the* Tutte polynomial *is defined by*

$$t_G(X,Y) = \sum_{A\subseteq G}(X-1)^{r(G)-r(A)}(Y-1)^{|A|-r(A)}.$$

If we have a linear code C over $\mathbb{F}_q$ with generator matrix G, we can interpret the columns of G as a matroid. Dependance is the usual linear dependance, and the rank function is the column rank of the submatrix consisting of some columns of G, as we described in Lemma 2.1.2. Note that equivalent codes give the same matroid.

Proposition 3.3.2 *Let C be a linear code over $\mathbb{F}_q$ with generator matrix G, and consider G as a matroid. Then the Tutte polynomial associated with the code C is*

$$t_G(X,Y) = \sum_{t=0}^{n}\sum_{|J|=t}(X-1)^{l(J)}(Y-1)^{l(J)-(k-t)}.$$

Proof: In Lemma 2.1.2 we found that $r(J) = k - l(J)$. Inserting this in the Tutte polynomial and splitting the summation gives the above formula. □

The way we have rewritten the Tutte polynomial associated with a linear code C suggests a connection between the weight enumerator and the Tutte polynomial. This connection is given in the next theorem.

Theorem 3.3.3 *Let C be a linear $[n,k]$ code over $\mathbb{F}_q$ with generator matrix G. Then the following holds for the Tutte polynomial and the extended weight enumerator:*

$$W_C(X,Y,T) = (X-Y)^k Y^{n-k}\ t_G\left(\frac{X+(T-1)Y}{X-Y}, \frac{X}{Y}\right).$$

Proof: By using the previous proposition about the Tutte polynomial, rewriting, and Proposition 2.2.6 we get

$$\begin{aligned}
& (X-Y)^k Y^{n-k}\ t_G\left(\frac{X+(T-1)Y}{X-Y}, \frac{X}{Y}\right) \\
= \ & (X-Y)^k Y^{n-k} \sum_{t=0}^{n} \sum_{|J|=t} \left(\frac{TY}{X-Y}\right)^{l(J)} \left(\frac{X-Y}{Y}\right)^{l(J)-(k-t)} \\
= \ & (X-Y)^k Y^{n-k} \sum_{t=0}^{n} \sum_{|J|=t} T^{l(J)} Y^{k-t} (X-Y)^{-(k-t)} \\
= \ & \sum_{t=0}^{n} \sum_{|J|=t} T^{l(J)} (X-Y)^t Y^{n-t} \\
= \ & W_C(X,Y,T).
\end{aligned}$$

□

Note that we use the extended weight enumerator here. We do that because extending a code does not change the generator matrix and therefore not the matroid G. The converse of this theorem is also true: the Tutte polynomial is completely defined by the extended weight enumerator. We show this in the following theorem.

Theorem 3.3.4 *Let C be a linear code over $\mathbb{F}_q$ with generator matrix G. Then the following holds for the extended weight enumerator and the Tutte polynomial:*

$$t_G(X,Y) = Y^n (Y-1)^{-k} W_C(1, Y^{-1}, (X-1)(Y-1)).$$

Proof: The proof of this theorem goes analogous to the proof of the previous theorem.

$$\begin{aligned}
& Y^n (Y-1)^{-k} W_C(1, Y^{-1}, (X-1)(Y-1)) \\
= \ & Y^n (Y-1)^{-k} \sum_{t=0}^{n} \sum_{|J|=t} ((X-1)(Y-1))^{l(J)} (1-Y^{-1})^t Y^{-(n-t)}
\end{aligned}$$

$$\begin{aligned}
&= \sum_{t=0}^{n} \sum_{|J|=t} (X-1)^{l(J)}(Y-1)^{l(J)}Y^{-t}(Y-1)^{t}Y^{-(n-k)}Y^{n}(Y-1)^{-k} \\
&= \sum_{t=0}^{n} \sum_{|J|=t} (X-1)^{l(J)}(Y-1)^{l(J)-(k-t)} \\
&= t_G(X,Y).
\end{aligned}$$

□

We see that the Tutte polynomial depends on two variables, while the extended weight enumerator depends on three variables. This is no problem, because the weight enumerator is given in its homogeneous form here: we can view the extended weight enumerator as a polynomial in two variables via $W_C(Z,T) = W_C(1,Z,T)$.
We can also give expressions for the generalized weight enumerator in terms of the Tutte polynomial, and the other way round.

Theorem 3.3.5 *For the generalized weight enumerator of a linear code C and the associated Tutte polynomial we have*

$$W_C^r(X,Y) = \frac{1}{\langle r\rangle_q} \sum_{j=0}^{r} \begin{bmatrix} r \\ j \end{bmatrix}_q (-1)^{r-j} q^{(r-j)(r-j-1)/2}$$

$$\times (X-Y)^k Y^{n-k}\; t_G\left(\frac{X+(q^j-1)Y}{X-Y}, \frac{X}{Y}\right),$$

And, conversely,

$$t_G(X,Y) = Y^n(Y-1)^{-k} \sum_{r=0}^{k} \left(\prod_{j=0}^{r-1} ((X-1)(Y-1)-q^j) \right) W_C^r(1,Y^{-1}).$$

Proof: For the first formula, use Theorems 3.2.1 and 3.3.3. Use Theorems 3.1.4 and 3.3.4 for the second formula. □

3.4 Overview

In the previous sections we established relations between the generalized weight enumerators for $0 \le r \le k$, the extended weight enumerator and the

Tutte polynomial. We summarize this in the following diagram:

$$\begin{tikzcd}
W_C(X,Y) & & & \\
& & W_C(X,Y,T) \arrow[ull] \arrow[dl, shift left, "3.2.1"] \arrow[d, shift left, "3.3.4"] \arrow[ddr, leftrightarrow] & \\
& \{W_C^r(X,Y)\}_{r=0}^k \arrow[uul] \arrow[ur, shift left, "3.1.4"] \arrow[r, shift left, "3.3.5"] \arrow[drr, leftrightarrow] & t_G(X,Y) \arrow[u, shift left, "3.3.3"] \arrow[l, shift left, "3.3.5"] & \\
& & & \{W_C^r(X,Y,T)\}_{r=0}^k
\end{tikzcd}$$

We see that the Tutte polynomial, the extended weight enumerator and the collection of generalized weight enumerators all contain the same amount of information about a code, because they completely define each other. We use $W_C(X,Y) = W_C^0(X,Y) + (q-1)W_C^1(X,Y)$ and $W_C(x,Y) = W_C(X,Y,q)$ to find the weight enumerator $W_C(X,Y)$. The latter contains less information and therefore does not determine $W_C(X,Y,T)$ or $\{W_C^r(X,Y)\}_{r=0}^k$. An example will be given in section 5.2.

The generalized extended weight enumerator can be obtained from the extended weight enumerator in the following way. First we use q^m instead of q in Theorem 3.2.1 to obtain $\{W_{C\otimes\mathbb{F}_{q^m}}^r(X,Y)\}_{r=0}^k$. If we do this for $k+1$ values of m, we can determine $\{W_C^r(X,Y,T)\}_{r=0}^k$ by Lagrange interpolation. (See also the remark after Theorem 3.1.4.) So the generalized extended weight enumerator does not contain more information then the Tutte polynomial, the extended weight enumerator and the collection of generalized weight enumerators

3.5 Notes

The isomorphism given in Proposition 3.1.1 was suggested by Simonis [12]. Kløve [7] already proved Theorem 3.1.4, but did not use the given isomorphism in his proof. The second section, is new material. The connection with the Tutte polynomial was first given by Greene [4] in the form of Theorem 3.3.3 with $T = q$. His proof uses Tutte-Grothendieck invariants and a variation on the method described in section 2.3, and is much longer then our variant. Theorems 3.3.4 and 3.3.5 are newly derived.

Chapter 4

MacWilliams identities

4.1 Using the Tutte polynomial

In this section, we will prove the MacWilliams identities using the Tutte polynomial. We do this because of the following very useful relation between the Tutte polynomial of a matroid and its dual:

Theorem 4.1.1 *Let $t_G(X,Y)$ be the Tutte polynomial of a matroid G, and let G^* be the dual matroid. Then $t_G(X,Y) = t_{G^*}(Y,X)$.*

Proof (sketch): To prove the theorem, we use two relations between the rank function of a matroid G and its dual G^* with underlying set S:

$$r^*(G^*) + r(G) = |S|, \qquad r^*(A) = |A| + r(S-A) - r(G).$$

This relations can be proved using basic facts about matroids. Substituting the last relation into the definition of the Tutte polynomial for the dual code, gives

$$\begin{aligned} t_{G^*}(X,Y) &= \sum_{A\subseteq G^*} (X-1)^{r^*(G^*)-r^*(A)}(Y-1)^{|A|-r^*(A)} \\ &= \sum_{A\subseteq G} (X-1)^{r^*(G^*)-|A|-r(S-A)+r(G)}(Y-1)^{r(G)-r(S-A)} \\ &= \sum_{A\subseteq G} (X-1)^{|S-A|-r(S-A)}(Y-1)^{r(G)-r(S-A)} \\ &= t_G(Y,X) \end{aligned}$$

In the last step, we use that the summation over all $A \subseteq G$ is the same as a summation over all $S - A \subseteq G$. This proves the theorem. □

If we consider a code as a matroid, then the dual matroid is the dual code. Therefore we can use the above theorem to prove the MacWilliams relations.

Theorem 4.1.2 (MacWilliams) *Let C be a code and let $C^\perp$ be its dual. Then the extended weight enumerator of C completely determines the extended weight enumerator of $C^\perp$ and vice versa, via the following formula:*

$$W_{C^\perp}(X,Y,T) = T^{-k} W_C(X+(T-1)Y, X-Y, T).$$

Proof: Using the previous theorem, and the relation between the weight enumerator and the Tutte polynomial, we find

$$\begin{aligned}
& T^{-k} W_C(X+(T-1)Y, X-Y, T) \\
= \ & T^{-k}(TY)^k (X-Y)^{n-k} \ t_G\left(\frac{X}{Y}, \frac{X+(T-1)Y}{X-Y}\right) \\
= \ & Y^k (X-Y)^{n-k} \ t_G\left(\frac{X+(T-1)Y}{X-Y}, \frac{X}{Y}\right) \\
= \ & W_{C^\perp}(X,Y,T).
\end{aligned}$$

Notice in the last step that $\dim C^\perp = n-k$, and $n-(n-k)=k$. □

4.2 Generalized MacWilliams identities

We can use the relations in Theorems 3.1.4 and 3.2.1 to prove the MacWilliams identities for the generalized weight enumerator.

Theorem 4.2.1 *Let C be a code and let $C^\perp$ be its dual. Then the generalized weight enumerators of C completely determine the generalized weight enumerators of $C^\perp$ and vice versa, via the following formula:*

$$W_{C^\perp}^r(X,Y) =$$

$$\sum_{j=0}^{r}\sum_{l=0}^{j}(-1)^{r-j}\frac{q^{(r-j)(r-j-1)/2-j(r-j)-l(j-l)-jk}}{\langle r-j\rangle_q\langle j-l\rangle_q} W_C^l(X+(q^j-1)Y, X-Y).$$

Proof: We write the generalized weight enumerator in terms of the extended weight enumerator, use the MacWilliams identities for the extended weight enumerator, and convert back to the generalized weight enumerator.

$$\begin{aligned}
W^r_{C^\perp} &= \frac{1}{\langle r\rangle_q}\sum_{j=0}^{r}\begin{bmatrix} r \\ j\end{bmatrix}_q (-1)^{r-j}q^{(r-j)(r-j-1)/2}\ W_{C^\perp}(X,Y,q^i) \\
&= \sum_{j=0}^{r}(-1)^{r-j}\frac{q^{(r-j)(r-j-1)/2-j(r-j)}}{\langle j\rangle_q\langle r-j\rangle_q}q^{-jk}W_c(X+(q^j-1)Y,X-Y,q^j) \\
&= \sum_{j=0}^{r}(-1)^{r-j}\frac{q^{(r-j)(r-j-1)/2-j(r-j)-jk}}{\langle j\rangle_q\langle r-j\rangle_q} \\
&\quad\ \times\sum_{l=0}^{j}\frac{\langle j\rangle_q}{q^{l(j-l)}\langle j-l\rangle_q}W^l_C(X+(q^j-1,X-Y) \\
&= \sum_{j=0}^{r}\sum_{l=0}^{j}(-1)^{r-j}\frac{q^{(r-j)(r-j-1)/2-j(r-j)-l(j-l)-jk}}{\langle r-j\rangle_q\langle j-l\rangle_q} \\
&\quad\ \times W^l_C(X+(q^j-1,X-Y).
\end{aligned}$$

□

4.3 An alternative approach

In [12] Simonis proved an alternative version of the generalized MacWilliams identities. We will prove this theorem directly, by showing the equivalence with Theorem 4.2.1.

Theorem 4.3.1 *The generalized Hamming weights A^r_w of a linear $[n,k]$ code C over $\mathbb{F}_q$ and the generalized Hamming weights $\tilde{A}^r_w$ of the dual code $C^\perp$ are related via*

$$\sum_{i=0}^{n}\binom{n-i}{n-m}\tilde{A}^r_i = \sum_{l=0}^{r}q^{l(m-k+l-r)}\begin{bmatrix} m-k \\ r-l\end{bmatrix}_q\sum_{v=0}^{n}\binom{n-v}{m}A^l_v.$$

Proof: We will multiply both sides of the equation by Y^nU^{n-m} and sum over m from 0 to n. We start with the left hand side of the equation. Changing the order of summation, using the binomial expansion of $(U+1)^{n-i}$ and changing variables via $X = Y(U+1)$ gives

$$\sum_{m=0}^{n}Y^nU^{n-m}\sum_{i=0}^{n}\binom{n-i}{n-m}\tilde{A}^r_i$$

$$
\begin{aligned}
&= \sum_{i=0}^{n} \tilde{A}_i^r Y^n \sum_{m=0}^{n} \binom{n-i}{n-m} U^{n-m} \\
&= \sum_{i=0}^{n} \tilde{A}_i^r Y^n \sum_{m=0}^{n-i} \binom{n-i}{n-m-i} U^{n-m-i} \\
&= \sum_{i=0}^{n} \tilde{A}_i^r Y^n (U+1)^{n-i} \\
&= \sum_{i=0}^{n} \tilde{A}_i^r Y^i X^{n-i}.
\end{aligned}
$$

This is exactly the generalized weight enumerator of the dual code, and thus equal to the left hand side of Theorem 4.2.1.
The right hand side of the equation requires some more work. First, we rewrite the Gaussian binomial coefficient in the following way:

$$
\begin{aligned}
\begin{bmatrix} m-k \\ r-l \end{bmatrix}_q &= \frac{\prod_{t=0}^{r-l-1}(q^{m-k}-q^t)}{\langle r-l\rangle_q} \\
&= \frac{1}{\langle r-l\rangle_q} \sum_{t=0}^{r-l} \begin{bmatrix} r-l \\ t \end{bmatrix}_q (-1)^{r-l-t} q^{\binom{r-l-t}{2}} q^{t(m-k)} \\
&= \frac{1}{\langle r-l\rangle_q} \sum_{j=l}^{r} \begin{bmatrix} r-l \\ j-l \end{bmatrix}_q (-1)^{r-j} q^{\binom{r-j}{2}+(j-l)(m-k)} \\
&= \sum_{j=l}^{r} (-1)^{r-j} \frac{q^{\binom{r-j}{2}+(j-l)(m-k)}}{\langle r-l\rangle_q} \cdot \frac{q^{-(j-l)(r-j)}\langle r-l\rangle_q}{\langle j-l\rangle_q \langle r-j\rangle_q} \\
&= \sum_{j=l}^{r} (-1)^{r-j} \frac{q^{(r-j)(r-j-1)/2+(j-l)((m-k)-(r-j))}}{\langle r-j\rangle_q \langle j-l\rangle_q}.
\end{aligned}
$$

In the first step, we use the same identity as in the proof of Theorem 3.2.1, and in the second step we substitute $t = j - l$. If we substitute this expression, take together the powers of q and change the order of summation, we find the right hand side of the formula to be equal to

$$
\sum_{j=0}^{r} \sum_{l=0}^{j} (-1)^{r-j} \frac{q^{(r-j)(r-j-1)/2-j(r-j)-l(j-l)-jk}}{\langle r-j\rangle_q \langle j-l\rangle_q} \cdot q^{jm} \sum_{v=0}^{n} \binom{n-v}{m} A_v^l.
$$

This already looks a lot like the right hand side of Theorem 4.2.1. We are now ready to do the same procedure to the above as we did before to the left hand side: we multiply by $Y^n U^{n-m}$ and sum over m from 0 to n. All this does not depend on j nor l, so we can change the summations easily. It remains to show that

$$\sum_{m=0}^{n} Y^n U^{n-m} q^{jm} \sum_{v=0}^{n} \binom{n-v}{m} A_v^l = W_C^l(X+(q^j-1)Y, X-Y),$$

where $U+1 = XY^{-1}$. Changing the order of summation, substituting $u = n-m$, using the binomial expansion of $(U+q^j)^{n-v}$ and changing variables via $X = Y(U+1)$ gives indeed

$$\begin{aligned}
& \sum_{m=0}^{n} Y^n U^{n-m} q^{jm} \sum_{v=0}^{n} \binom{n-v}{m} A_v^l \\
= & \sum_{v=0}^{n} A_v^l Y^n U^v \sum_{m=0}^{n} \binom{n-v}{m} U^{n-m-v} q^{jm} \\
= & \sum_{v=0}^{n} A_v^l Y^n U^v \sum_{m=v}^{n} \binom{n-v}{n-m} U^{n-m} q^{j(m-v)} \\
= & \sum_{v=0}^{n} A_v^l Y^n U^v \sum_{u=0}^{n-v} \binom{n-v}{u} U^u q^{j(n-v-u)} \\
= & \sum_{v=0}^{n} A_v^l Y^n U^v (U+q^j)^{n-v} \\
= & \sum_{v=0}^{n} A_v^l (Y(U+1)+(q^j-1)Y)^{n-v} (Y(U+1)-Y)^v \\
= & \sum_{v=0}^{n} A_v^l (X+(q^j-1)Y)^{n-v} (X-Y)^v \\
= & W_C^l(X+(q^j-1)Y, X-Y).
\end{aligned}$$

This proves the theorem. □

Stating the generalized MacWilliams identities in this way has the advantage that we can now derive MacWilliams-like identities for the B_t^r.

Theorem 4.3.2 *Let C be a linear $[n,k]$ code over $\mathbb{F}_q$, with the associated B_t^r. Let $C^\perp$ be its dual, with associated $\tilde{B}_t^r$. Then the B_t^r and $\tilde{B}_t^r$ completely define each other in the following way:*

$$\tilde{B}_t^r = \sum_{l=0}^{r} q^{l(n-t-k+l-r)} \begin{bmatrix} n-t-k \\ r-l \end{bmatrix}_q B_{n-t}^l.$$

Proof: The formula is a direct consequence of the previous theorem and Proposition 2.1.6, with $n - m$ replaced by t. □

As we see, this formula is much more pleasant then the one in Theorems 4.2.1 and 4.3.1. This motivates the use of the B_t^r for the weight enumerator.

4.4 Notes

Greene [4] first used the Tutte polynomial to prove the MacWilliams identities, see also Brylawsky and Oxley [2]. Theorem 4.2.1 was proved by Kløve in [7], although he uses only half of the relations between the generalized weight enumerator and the extended weight enumerator. Using both makes the proof much shorter. The formula in Theorem 4.3.1 was proved by Simonis in [12] with use of an idea quite similar to the B_t^r we use. The connection between the two formulas is first proven here, and uses an argument similar to Blahut's proof [1] of the ordinary MacWilliams relations.

Chapter 5

Examples

5.1 MDS codes

We can use the theory in the second chapter to calculate the weight distribution, generalized weight distribution, and extended weight distribution of a linear $[n, k]$ code C. This is done by determining the values $l(J)$ for each $J \subseteq [n]$. In general, we have to look at the 2^n different subcodes of C to find the $l(J)$, but for the special case of MDS codes we can find the weight distributions much faster.

Proposition 5.1.1 *Let C be a linear $[n, k]$ MDS code, and let $J \subseteq [n]$. Then we have*

$$l(J) = \left\{ \begin{array}{cl} 0 & \textit{for } t > k \\ k - t & \textit{for } t \leq k \end{array} \right.$$

so for a given t the value of $l(J)$ is independent of the choice of J.

Proof: We know that the dual of a MDS code is also MDS, so $d^\perp = k + 1$. Now use $d = n - k + 1$ in lemma 2.1.3. □

Now that we know all the $l(J)$ for an MDS code, it is easy to find the weight distribution. We will give the construction for the generalized extended weight enumerator here, because all other cases can be deduced from it.

Theorem 5.1.2 *Let C be a MDS code with parameters $[n, k]$. Then the generalized extended weight distribution is given by*

$$A_w^r(T) = \binom{n}{w} \sum_{j=0}^{w-d} (-1)^j \binom{w}{j} \begin{bmatrix} w - d + 1 - j \\ r \end{bmatrix}_T .$$

Proof: We know from the proposition that for a MDS code, $B_J^r(T)$ depends only on the size of J, so $B_t^r(T) = \binom{n}{t}\begin{bmatrix} k-t \\ r \end{bmatrix}_T$. Using this in the formula for $A_w^r(T)$ and substituting $j = t - n + w$, we have

$$\begin{aligned}
A_w^r(T) &= \sum_{t=n-w}^{n-d_r} (-1)^{n+w+t} \binom{t}{n-w} B_t^r(T) \\
&= \sum_{t=n-w}^{n-d_r} (-1)^{t-n+w} \binom{t}{n-w} \binom{n}{t} \begin{bmatrix} k-t \\ r \end{bmatrix}_T \\
&= \sum_{j=0}^{w-d_r} (-1)^j \binom{n}{w} \binom{w}{j} \begin{bmatrix} k+w-n-j \\ r \end{bmatrix}_T \\
&= \binom{n}{w} \sum_{j=0}^{w-d_r} (-1)^j \binom{w}{j} \begin{bmatrix} w-d+1-j \\ r \end{bmatrix}_T .
\end{aligned}$$

In the second step, we are using the binomial equivalence

$$\binom{n}{t}\binom{t}{n-w} = \binom{n}{n-w}\binom{n-(n-w)}{t-(n-w)} = \binom{n}{w}\binom{w}{n-t}.$$

□

So, for all MDS-codes with given parameters $[n, k]$ the extended and generalized weight distributions are the same. But not all such codes are equivalent. We can conclude from this, that the generalized extended weight enumerator is not enough to distinguish between codes with the same parameters. We illustrate the non-equivalence of two MDS codes by an example.

Let C be a linear $[n, 3]$ MDS code over $\mathbb{F}_q$. It is possible to write the generator matrix G of C in the following form:

$$\begin{pmatrix} 1 & 1 & \dots & 1 \\ x_1 & x_2 & \dots & x_n \\ y_1 & y_2 & \dots & y_n \end{pmatrix} .$$

Because C is MDS we have $d = n - 2$. We now view the n columns of G as points in the projective plane $\mathrm{PG}(2, q)$, say $P_1, \dots, P_n$. The MDS property that every k columns of G are independent is now equivalent with saying that no three points are on a line. To see that these n points do not always determine an equivalent code, consider the following construction. Through the n points there are $\binom{n}{2} = N$ lines, the set $\mathcal{N}$. These lines determine (the generator matrix of) a $[N,3]$ code $\hat{C}$. The minimum distance of the code $\hat{C}$ is equal to the total number of lines minus the maximum number of lines from $\mathcal{N}$ through an arbitrarily point $P \in \mathrm{PG}(2,q)$. If $P \notin \{P_1,\dots,P_n\}$ then the maximum number of lines from $\mathcal{N}$ through P is at most $\frac{1}{2}n$, since no three points of $\mathcal{N}$ lie on a line. If $P = P_i$ for some $i \in [n]$ then P lies on exactly $n-1$ lines of $\mathcal{N}$, namely the lines P_iP_j for $j \neq i$. Therefore the minimum distance of $\hat{C}$ is $d = N - n + 1$.

We now have constructed a $[N,3,N-n+1]$ code $\hat{C}$ from the original code C. Notice that two codes C_1 and C_2 are equivalent if and only if $\hat{C}_1$ and $\hat{C}_2$ are equivalent. The (generalized extended) weight enumerators $W^r_C(X,Y,T)$ are completely determined, but this is not generally true for $W^r_{\hat{C}}(X,Y,T)$.

Take for example $n = 4$ and $q = 4$, so $\hat{C}$ is a $[6,3,3]$ code. The six columns can be viewed as six points in the projective plane. We know that every 5-tuple of points in the projective plane determines a unique conic. If we take five of our six points and look at the conic they are on, there are three possibilities for the last point: it can also be on the conic, it can be the nucleus of the conic (every tangent of the conic runs through the same point because q is even), or it can be some other point. Every possibility can occur with $d = 3$ for the final code, so not all $[6,3,3]$ codes are equivalent. Therefore also not all $[4,3,2]$ codes are equivalent.

5.2 Two $[6,3]$ codes

In this section, we will determine the generalized and extended weight distribution of two linear $[6,3]$ codes. Let C_1 and C_2 be the codes over $\mathbb{F}_2$ respectively determined by the following generator matrices:

$$\begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}, \qquad \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}.$$

As we can see from the generator matrices, we have $d = 2$ for both codes, and also $d^{\perp} = 2$. Now we determine the values of $l(J)$ for both codes, using

Lemma 2.1.3 and writing out the other cases.

t	C_1	C_2
$t=0$	$l(J)=3$	$l(J)=3$
$t=1$	$l(J)=2$	$l(J)=2$
$t=2$	$3\times\ l(J)=2$ $12\times\ l(J)=1$	$3\times\ l(J)=2$ $12\times\ l(J)=1$
$t=3$	$1\times\ l(J)=2$ $9\times\ l(J)=1$ $10\times\ l(J)=0$	$12\times\ l(J)=1$ $8\times\ l(J)=0$
$t=4$	$3\times\ l(J)=1$ $12\times\ l(J)=0$	$3\times\ l(J)=1$ $12\times\ l(J)=0$
$t=5$	$l(J)=0$	$l(J)=0$
$t=6$	$l(J)=0$	$l(J)=0$

From this table, we see that the two codes only differ at $t=3$, so in the following we will only distinguish between C_1 and C_2 for $t=3$. For $B_t^r(T)$ we find the following general formulas:

$$\begin{aligned}
B_0^r(T) &= \begin{bmatrix}3\\r\end{bmatrix}_T\\
B_1^r(T) &= 6\cdot\begin{bmatrix}2\\r\end{bmatrix}_T\\
B_2^r(T) &= 3\cdot\begin{bmatrix}2\\r\end{bmatrix}_T+12\cdot\begin{bmatrix}1\\r\end{bmatrix}_T\\
B_3^r(T)_1 &= \begin{bmatrix}2\\r\end{bmatrix}_T+9\cdot\begin{bmatrix}1\\r\end{bmatrix}_T+10\cdot\begin{bmatrix}0\\r\end{bmatrix}_T\\
B_3^r(T)_2 &= 12\cdot\begin{bmatrix}1\\r\end{bmatrix}_T+8\cdot\begin{bmatrix}0\\r\end{bmatrix}_T\\
B_4^r(T) &= 3\cdot\begin{bmatrix}1\\r\end{bmatrix}_T+12\cdot\begin{bmatrix}0\\r\end{bmatrix}_T\\
B_5^r(T) &= 6\cdot\begin{bmatrix}0\\r\end{bmatrix}_T\\
B_6^r(T) &= \begin{bmatrix}0\\r\end{bmatrix}_T
\end{aligned}$$

The first B_3^r is for C_1 and the second is for C_2. Working this out for $r = 0, 1, 2, 3$ we get $B_t^0(T) = \binom{6}{t}$ for $r = 0$. For $r = 1$ we have

$$\begin{aligned}
B_0^1(T) &= T^2 + T + 1\\
B_1^1(T) &= 6T + 6\\
B_2^1(T) &= 3T + 15\\
B_3^1(T)_1 &= T + 10\\
B_3^1(T)_2 &= 12\\
B_4^1(T) &= 3\\
B_5^1(T) &= 0\\
B_6^1(T) &= 0
\end{aligned}$$

And for $r = 2$ we find

$$\begin{aligned}
B_0^2(T) &= T^2 + T + 1\\
B_1^2(T) &= 6\\
B_2^2(T) &= 3\\
B_3^2(T)_1 &= 1\\
B_3^2(T)_2 &= 0\\
B_4^2(T) &= 0\\
B_5^2(T) &= 0\\
B_6^2(T) &= 0
\end{aligned}$$

For $r = 3$ we have $B_t^3(T) = 0$, except for $B_0^3(T) = 1$.

We now can determine the $A_w^r(T)$ for $r = 0, 1, 2, 3$ using Theorem 2.4.3. For both codes $A_w^0(T)$ and $A_w^3(T)$ are the same, because $B_t^0(T)$ and $B_t^3(T)$ are the same. When $r = 0$ we got

$$A_w^0 = \sum_{t=6-w}^{6} (-1)^{6-w-t} \binom{t}{6-w} \binom{6}{t} = \delta_{6,6-w}.$$

Furthermore, we have $A_w^3(T) = 0$ except for $A_6^3(T) = 1$. For C_1 we have

$$\begin{aligned}
A_0^1(T) &= 0\\
A_1^1(T) &= 0\\
A_2^1(T) &= 3\\
A_3^1(T) &= T - 2\\
A_4^1(T) &= 3\\
A_5^1(T) &= 3T - 6\\
A_6^1(T) &= T^2 - 3T + 3
\end{aligned}$$

when $r = 1$, and for $r = 2$ we have

$$\begin{array}{rcl} A_0^2(T) &=& 0 \\ A_1^2(T) &=& 0 \\ A_2^2(T) &=& 0 \\ A_3^2(T) &=& 1 \\ A_4^2(T) &=& 0 \\ A_5^2(T) &=& 3 \\ A_6^2(T) &=& T^2 + T - 3 \end{array}$$

For C_2 we find that

$$\begin{array}{rcl} A_0^1(T) &=& 0 \\ A_1^1(T) &=& 0 \\ A_2^1(T) &=& 3 \\ A_3^1(T) &=& 0 \\ A_4^1(T) &=& 3T - 3 \\ A_5^1(T) &=& 0 \\ A_6^1(T) &=& T^2 - 2T + 1 \end{array}$$

when $r = 1$, and for $r = 2$ we find

$$\begin{array}{rcl} A_0^2(T) &=& 0 \\ A_1^2(T) &=& 0 \\ A_2^2(T) &=& 0 \\ A_3^2(T) &=& 0 \\ A_4^2(T) &=& 3 \\ A_5^2(T) &=& 0 \\ A_6^2(T) &=& T^2 + T - 2 \end{array}$$

We notice that for the two codes the $A_w^r(T)$ look much more different than the $B_t^r(T)$. This is because for $w > 2$, we need $B_3^r(T)$ in our calculations, and this values differ for C_1 and C_2. Likewise, for $w \leq 2$ the $A_w^r(T)$ are the same.

We are now ready to actually calculate the weight distributions of C_1 and C_2. First, we give the generalized weight distributions. They are obtained by substituting $T = 2$ in the above expressions.

r	C_1	C_2
0	$(1, 0, 0, 0, 0, 0, 0)$	$(1, 0, 0, 0, 0, 0, 0)$
1	$(0, 0, 3, 0, 3, 0, 1)$	$(0, 0, 3, 0, 3, 0, 1)$
2	$(0, 0, 0, 1, 0, 3, 3)$	$(0, 0, 0, 0, 3, 0, 4)$
3	$(0, 0, 0, 0, 0, 0, 1)$	$(0, 0, 0, 0, 0, 0, 1)$

So, the normal weight distributions of C_1 and C_2 are the same, but the higher order weight distributions are not. Therefore the generalized weight distribution is a better invariant to distinguish between codes.
The first extended weight distributions of C_1 and C_2 are

m	C_1	C_2
1	$(1,0,3,0,3,0,1)$	$(1,0,3,0,3,0,1)$
2	$(1,0,9,6,9,18,21)$	$(1,0,9,0,27,0,27)$
3	$(1,0,21,42,21,126,301)$	$(1,0,21,0,147,0,343)$
4	$(1,0,45,210,45,630,3165)$	$(1,0,45,0,675,0,3375)$
5	$(1,0,93,930,93,2790,28861)$	$(1,0,93,0,2883,0,29791)$
6	$(1,0,189,3906,189,11718,246141)$	$(1,0,189,0,11907,0,250047)$
7	$(1,0,381,16002,381,48006,2032381)$	$(1,0,381,0,48387,0,2048383)$

We used $W_{C\otimes\mathbb{F}_{q^m}}(X,Y) = W_C^0(X,Y,q^m)+(q^m-1)W_C^1(X,Y,q^m)$ to find these values.

5.3 Notes

The weight distribution of MDS codes has been known for long, see MacWilliams and Sloane [8] page 330. The examples in section 5.2 are from Simonis [12].

Chapter 6

Implementation and complexity

6.1 About the computer implementation

We use the computer algebra package Magma to implement the calculations of the weight enumerator. This is done by the method described in the second chapter, via calculation of $l(J)$ and B_t^r. We will give a short description of the functions in the code, with references to the corresponding theorems.

`GaussianNumber(k,r,q)`
Unfortunately, the existing Magma command `GaussianBinomial(n,k,v)` does not mean what we want it to mean. So a new function is written.

`Brt(C,r,t)`
This function calculates B_t^r in the way of Definition 2.1.4. We use of the Magma command `ShortenCode(C,J)`, which is the same as giving $C(J)$ with the all-zero coordinates removed. This removing does not change the dimension, so we can determine $l(J)$ from it.

`GeneralizedWeightDistribution(C,r)`
This function determines the r-th generalized Hamming weights of a code C, using the function `Brt(C,r,t)` and Proposition 2.1.8.

`BtT(C,t,m)`
This function calculates $B_t(T)$ in the way of Definition 2.2.1. As in the function `Brt(C,r,t)`, we use of the Magma command `ShortenCode(C,J)`.

`ExtendedWeightDistribution(C,m)`
This function determines the weight distribution of the m-th extension of a

code C, using the function `BtT(C,t,m)` and Theorem 2.2.3.

`ExtendedWeightDistribution2(C,m)`
Another way to determine the extended weight distribution, now using Theorem 3.1.4 and the function `GeneralizedWeightDistribution(C,r)`.

`GeneralizedWeightDistribution2(C,r)`
Another way to determine the generalized weight enumerator, now using Theorem 3.2.1 and the function `ExtendedWeightDistribution(C,m)`.

6.2 The implementation

```
function GaussianNumber(k,r,q)
N:=1;
for i in [0..r-1] do
  N *:= (q^(k-i)-1)/(q^(r-i)-1);
end for;
return N;
end function;

function Brt(C,r,t)
// C code, t in [0..n], r in [0..k]
n:=Length(C);
q:=#Field(C);
B:=0;
S:={i:i in [1..n]};
if t ne n then
// ShortenCode does not work for t=n, in that case B=0
  for J in Subsets(S,t) do
    l:=Dimension(ShortenCode(C,J));
    B +:= GaussianNumber(l,r,q);
  end for;
elif r eq 0 then
  B:=1;
end if;
return B;
end function;

function GeneralizedWeightDistribution(C,r)
```

```
// C code, r in [0..k]
n:=Length(C);
B:=[0];
for t in [0..n] do
  B[t+1]:=Brt(C,r,t);
end for;
A:=[0];
for w in [0..n] do
  A[w+1]:=0;
  for t in [n-w..n] do
    A[w+1] +:= (-1)^(n+w+t)*Binomial(t,n-w)*B[t+1];
  end for;
end for;
return A;
end function;

function BtT(C,t,m)
// C code, t in [0..n]
n:=Length(C);
q:=#Field(C);
B:=0;
S:={i:i in [1..n]};
if t ne n then
// ShortenCode does not work for t=n, in that case B=0
  for J in Subsets(S,t) do
    l:=Dimension(ShortenCode(C,J));
    B +:= q^(m*l)-1;
  end for;
end if;
return B;
end function;

function ExtendedWeightDistribution(C,m)
// C code, m extension degree
n:=Length(C);
B:=[0];
for t in [0..n] do
  B[t+1]:=BtT(C,t,m);
end for;
A:=[1];
for w in [1..n] do
```

```
  A[w+1]:=0;
  for t in [n-w..n] do
    A[w+1] +:= (-1)^(n+w+t)*Binomial(t,n-w)*B[t+1];
  end for;
end for;
return A;
end function;

function ExtendedWeightDistribution2(C,m)
// C code, m extension degree
n:=Length(C);
q:=#Field(C);
A:=[0];
for w in [0..n] do
  A[w+1]:=0;
end for;
A:=Vector(A);
for r in [0..m] do
  mr:=1;
  for i in [0..r-1] do
    mr *:= (q^m-q^i);
  end for;
  A +:= mr*Vector(GeneralizedWeightDistribution(C,r));
end for;
A:=ElementToSequence(A);
return A;
end function;

function GeneralizedWeightDistribution2(C,r)
// C code, r in [0..k]
n:=Length(C);
q:=#Field(C);
A:=[0];
for w in [0..n] do
  A[w+1]:=0;
end for;
A:=Vector(A);
for i in [0..r] do
  A +:= GaussianNumber(r,i,q)*(-1)^(r-i)*q^((r-i)*(r-i-1)/2)
        *Vector(ExtendedWeightDistribution(C,i));
end for;
```

```
rr:=1;
for i in [0..r-1] do
  rr *:= (q^r-q^i);
end for;
A:=ElementToSequence(A);
for w in [0..n] do
  A[w+1] /:= rr;
end for;
return A;
end function;
```

6.3 Example: the ternary Golay code

As an example of the computer code in the previous section, we use it to find the generalized and extended weight distributions of the ternary Golay code and its extension. (By extension here we do not mean an enlargement of the ground field, but the addition of a coordinate to make the sum of the coordinates zero.) We construct the codes in a cyclic way:

```
K:=FiniteField(3);
L:=FiniteField(3^5);
b:=L.1^22;
Golay:=CyclicCode(11,{b},K);
ExtendedGolay:=ExtendCode(Golay);
```

This indeed gives the desired codes:

```
> Golay;
[11, 6, 5] Cyclic Linear Code over GF(3)
Generator matrix:
[1 0 0 0 0 0 2 0 1 2 1]
[0 1 0 0 0 0 1 2 2 2 1]
[0 0 1 0 0 0 1 1 1 0 1]
[0 0 0 1 0 0 1 1 0 2 2]
[0 0 0 0 1 0 2 1 2 2 0]
[0 0 0 0 0 1 0 2 1 2 2]

> ExtendedGolay;
[12, 6, 6] Linear Code over GF(3)
Generator matrix:
[1 0 0 0 0 0 2 0 1 2 1 2]
```

```
[0 1 0 0 0 0 1 2 2 2 1 0]
[0 0 1 0 0 0 1 1 1 0 1 1]
[0 0 0 1 0 0 1 1 0 2 2 2]
[0 0 0 0 1 0 2 1 2 2 0 1]
[0 0 0 0 0 1 0 2 1 2 2 1]
```

First we ask for the generalized weight distributions of both codes.

```
> for i in [0..6] do
for>   GeneralizedWeightDistribution(Golay,i);
for> end for;
[ 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 ]
[ 0, 0, 0, 0, 0, 66, 66, 0, 165, 55, 0, 12 ]
[ 0, 0, 0, 0, 0, 0, 0, 330, 825, 2695, 4125, 3036 ]
[ 0, 0, 0, 0, 0, 0, 0, 0, 165, 1705, 9405, 22605 ]
[ 0, 0, 0, 0, 0, 0, 0, 0, 0, 55, 1221, 9735 ]
[ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 11, 353 ]
[ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1 ]

> for i in [0..6] do
for>   GeneralizedWeightDistribution(ExtendedGolay,i);
for> end for;
[ 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 ]
[ 0, 0, 0, 0, 0, 0, 132, 0, 0, 220, 0, 0, 12 ]
[ 0, 0, 0, 0, 0, 0, 0, 0, 495, 880, 2970, 3960, 2706 ]
[ 0, 0, 0, 0, 0, 0, 0, 0, 0, 220, 1980, 9900, 21780 ]
[ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 66, 1320, 9625 ]
[ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 12, 352 ]
[ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1 ]
```

These values agree with the known weight distributions of the (extended) ternary Golay code. (See for example [8] or [10].) Now we do the same for the extended weight distributions. We will not take $m > 3$ because the numbers will get very large.

```
> for m in [0..3] do
for>   ExtendedWeightDistribution(Golay,m);
for> end for;
[ 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 ]
[ 1, 0, 0, 0, 0, 132, 132, 0, 330, 110, 0, 24 ]
[ 1, 0, 0, 0, 0, 528, 528, 15840, 40920, 129800, 198000, 145
824 ]
```

```
[ 1, 0, 0, 0, 0, 1716, 1716, 205920, 2372370, 20833670, 1082
10
960, 255794136 ]

> for m in [0..3] do
for>   ExtendedWeightDistribution(ExtendedGolay,m);
for> end for;
[ 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 ]
[ 1, 0, 0, 0, 0, 0, 264, 0, 0, 440, 0, 0, 24 ]
[ 1, 0, 0, 0, 0, 0, 1056, 0, 23760, 44000, 142560, 190080, 1
29984 ]
[ 1, 0, 0, 0, 0, 0, 3432, 0, 308880, 3025880, 24092640, 1136
67840,
246321816 ]
```

It is also possible to do the above calculations with the alternative functions `GeneralizedWeighDistribution2(C,r)` which uses the extended weight enumerator, and `ExtendedWeightDistribution2(C,m)`, which use the generalized weight enumerator. These functions give the same values, but the second functions are much slower.

6.4 Complexity calculations

We discussed multiple ways to determine the weight enumerator of a linear code. In this chapter, we look at the complexity of these calculations.

Definition 6.4.1 *The complexity* $\mathrm{Comp}(n,R)$ *of the calculation of the weight enumerator of a linear* $[n,k]$ *code over* $\mathbb{F}_q$ *with information rate* $R=\frac{k}{n}$ *is given as a function of* n *and* R*. The* exponent *of this function is defined as*

$$E(R)=\lim_{n\to\infty}\frac{\log_q \mathrm{Comp}(n,R)}{n}.$$

Remark that the complexity also depends on the used algorithm. Which algorithm is used, should be clear from the context.

The most straightforward way to calculate the weight enumerator is the brute force-method: we simply go through all words and determining their weight. There are q^k words of length n, so the complexity of this brute force-method is nq^k and therefore $E(R)=R$. Because of the MacWilliams relations, we may assume $R \leq \frac{1}{2}$.

If we use the $B_t^r(T)$ to determine the weight enumerator, we have to look at all $J \subseteq [n]$ and determine the dimension of $C(J)$. Therefore the complexity is $\mathcal{O}(2^n \cdot n^3)$ and $E(R) = \log_q 2$. So this method is faster than the brute force-method if $R > \log_q 2$. We already saw we can assume $R \leq \frac{1}{2}$, so it follows that this method is faster than the brute force method for $\log_q 2 < R < \frac{1}{2}$, in case $q > 4$. Unfortunately, in practice we often use binary codes, so $q = 2$. On the other hand, using the $B_t^r(T)$ gives us the generalized extended weight enumerator, not only the ordinary weight enumerator. That means we also determine the generalized weight enumerators $\{W_C^r(X,Y,T)\}_{r=0}^k$ and the weight enumerator $W_{C\otimes\mathbb{F}_{q^m}}(X,Y)$ of all extensioncodes over C.
Moreover, we can improve this exponent, because by Lemma 2.1.3 we know $l(J)$ for $t < d^\perp$ and $t > n - d$. The values of d and $d^\perp$ can be very small, but a code in general achieves the Gilbert–Varshamov bound (see [3]). That means we have the following values for d and $d^\perp$:

$$d \sim nH_q^{-1}(1-R) \qquad \text{and} \qquad d^\perp \sim nH_q^{-1}(R)$$

where $H(x) = -x\log_q x - (1-x)\log_q(1-x) + \log_q(q-1)$ is the q-ary entropy function. The complexity is thus equal to

$$\mathrm{Comp}(n,R) = \sum_{t=nH_q^{-1}(R)}^{n-nH_q^{-1}(1-R))} \binom{n}{t}.$$

6.5 Notes

For the binary Golay code of length 23, the given implementation is much to inefficient: Magma runs out of memory before any results are found. Unfortunately, there was no time to implement the method of section 2.3.
The problem of finding the weight distribution of a code is proven to be NP hard by Vardy [14]. More about codes achieving the Gilbert-Varshamov bound can be found in [3].

Chapter 7

Conclusions

In the second chapter, we described ways to determine the generalized weight enumerator, the extended weight enumerator and the generalized extended weight enumerator. These methods are very similar and based on the determination of $l(J)$ for all $J \subseteq [n]$, where $l(J) = \dim\{\mathbf{c} \in C : c_j = 0 \text{ for all } j \in J\}$. Besides this method, we discovered a way to determine the extended weight enumerator by matrix decomposition.
It tuns out that the generalized weight enumerators, the extended weight enumerator and the Tutte polynomial all contain the same amount of information about a code (or matroid). This is a better invariant then the original weight enumerator. All the in between links have been made explicit and an overview can be found in section 3.4. Further extension or generalization does not give more information about the code.
For the three equivalent polynomials we proved MacWilliams-like relations between the polynomial and the polynomial of the dual object – code or matroid. With use of the language described in the second chapter, we simplified known results and proved their equivalence. Using $B_t^r(T)$ in the definition of the weight enumerators turns out to be quite useful. Also, we gave examples of the theory.

Bibliography

[1] R.E. Blahut. *Algebraic Codes for Data Transmission.* Camebridge University Press, Camebridge, 2003.

[2] T.H. Brylawsky and J.G. Oxley. The tutte polynomial and its applications. In N. White, editor, *Matroid Applications.* Cambridge University Press, Cambridge, 1992.

[3] J.T. Coffey and R.M. Goodman. Any code of which we cannot think is good. *IEEE Transactions on Information Theory*, 36:1453–1461, 1990.

[4] C. Greene. Weight enumeration and the geometry of linear codes. *Studies in Applied Mathematics*, 55:119–128, 1976.

[5] G.L. Katsman and M.A. Tsfasman. Spectra of algebraic-geometric codes. *Problemy Peredachi Informatsii*, 23:19–34, 1987.

[6] T. Kløve. The weight distribution of linear codes over GF(q^l) having generator matrix over GF(q)*. *Discrete Mathematics*, 23:159–168, 1978.

[7] T. Kløve. Support weight distribution of linear codes. *Discrete Matematics*, 106/107:311–316, 1992.

[8] F.J. MacWilliams and N.J.A. Sloane. *The Theory of Error-Correcting Codes.* North-Holland Mathematical Library, Amsterdam, 1977.

[9] J.G. Oxley. *Matroid theory.* Oxford University Press, Oxford, 1992.

[10] R. Pellikaan, X.-W. Wu, and S. Bulygin. Codes and cryptography on algebraic curves. preliminary version, to be published by Cambridge University Press, 2008.

[11] J. Riordan. *Combinatorial Identities.* Robert E. Krieger Publishing Company, New York, 1979.

[12] J. Simonis. The effective length of subcodes. *AAECC*, 5:371–377, 1993.

[13] M.A. Tsfasman and S.G. Vlădut. *Algebraic-geometric codes.* Kluwer Academic Publishers, Dordrecht, 1991.

[14] A. Vardy. The intractibility of computing the minimum distance of a code. *IEEE Transactions on Information Theory*, 43:1757–1766, 1997.

[15] V.K. Wei. Generalized Hamming weights for linear codes. *IEEE Transactions on Information Theory*, 37:1412–1418, 1991.