UNIVERSIDADE FEDERAL DE MINAS GERAIS
Instituto de Ciências Exatas
Programa de Pós-graduação em Matemática

José Gustavo Coelho

Diophantine equations over finite fields and applications

Belo Horizonte
2024

José Gustavo Coelho

Diophantine equations over finite fields and applications

Tese de doutorado apresentada como parte dos requisitos para obtenção do título de Doutor pelo Departamento de Matemática do Instituto de Ciências Exatas da Universidade Federal de Minas Gerais.

Orientador : Fabio Enrique Brochero Martínez

Belo Horizonte
Agosto de 2024

Coelho, José Gustavo.

C672d Diophantine equations over finite Fields and applications [recurso eletrônico] / José Gustavo Coelho. – 2024
1 recurso online (74 f. il.) : pdf.

Orientador: Fabio Enrique Brochero Martínez.

Tese (doutorado) - Universidade Federal de Minas Gerais, Instituto de Ciências Exatas, Departamento de Matemática.
Referências: f. 63-65.

1. Matemática - Teses. 2. Corpos finitos (Álgebra) - Teses. 3. Equações diofantinas – Teses. I. Brochero Martínez, Fabio Enrique. III. Universidade Federal de Minas Gerais, Instituto de Ciências Exatas, Departamento de Matemática. IV. Título.

CDU 51(043)

Ficha catalográfica elaborada pela bibliotecária Irénquer Vismeg Lucas Cruz
CRB 6/819 - Universidade Federal de Minas Gerais - ICEx

UNIVERSIDADE FEDERAL DE MINAS GERAIS

DEPARTAMENTO DE MATEMÁTICA

PROGRAMA DE PÓS-GRADUAÇÃO EM MATEMÁTICA

FOLHA DE APROVAÇÃO

Diophantine equations over finite fields and applications

JOSÉ GUSTAVO COELHO

Tese defendida e aprovada pela banca examinadora constituída por:

Prof. Fabio Enrique Brochero Martínez
Orientador - UFMG

Prof. Alonso Sepúlveda Castellanos
UFU

Prof. José Alves Oliveira
UFLA

Prof. Osnel Broche Cristo
UFLA

Prof. Victor Gonzalo Lopez Neumann
UFU

Belo Horizonte, 02 de agosto de 2024.

Documento assinado eletronicamente por **Fabio Enrique Brochero Martinez**, **Professor do Magistério Superior**, em 20/08/2024, às 12:27, conforme horário oficial de Brasília, com fundamento no art. 5° do Decreto n° 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Alonso Sepúlveda Castellanos**, **Usuário Externo**, em 20/08/2024, às 16:41, conforme horário oficial de Brasília, com fundamento no art. 5º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **José Alves Oliveira**, **Usuário Externo**, em 20/08/2024, às 17:34, conforme horário oficial de Brasília, com fundamento no art. 5º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Osnel Broche Cristo**, **Usuário Externo**, em 20/08/2024, às 18:08, conforme horário oficial de Brasília, com fundamento no art. 5º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Victor Gonzalo Lopez Neumann**, **Usuário Externo**, em 20/08/2024, às 18:13, conforme horário oficial de Brasília, com fundamento no art. 5º do Decreto nº 10.543, de 13 de novembro de 2020.

A autenticidade deste documento pode ser conferida no site https://sei.ufmg.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0, informando o código verificador **3472724** e o código CRC **2F3700EA**.

Referência: Processo nº 23072.246274/2024-81 SEI nº 3472724

Resumo

A primeira parte desta tese foca-se em contar o número de soluções de equações diofantinas sobre corpos finitos para famílias que estão intimamente relacionadas com equações diagonais, apresentando resultados significativos nesta área. A segunda parte da tese está centrada na contagem de palavras-código de baixo peso em códigos cíclicos, apresentando alguns resultados novos. O capítulo introdutório estabelece as bases, introduzindo conceitos fundamentais que servem de fundamento para os capítulos subsequentes.

O Capítulo 2 estuda polinômios completamente triangulares, caracterizados por expressões da forma

$$f(x_1,\dots,x_n) = a_1 x_1^{d_{1,1}} + a_2 x_1^{d_{1,2}} x_2^{d_{2,2}} + \cdots + a_n x_1^{d_{1,n}} \cdots x_n^{d_{n,n}} - b,$$

onde $a_i \in \mathbb{F}_q^*$ e $b \in \mathbb{F}_q$, com $d_{i,j} > 0$ para todo $1 \le i \le j \le n$. Para esses polinômios, são derivadas fórmulas explícitas para a contagem do número de soluções sob condições aritméticas que os relacionam com polinômios diagonais de graus 1 e 2.

O Capítulo 3 desloca o foco para polinômios completos, definidos como

$$f(x_1,\dots,x_n) = a_1 x_1^{d_{1,1}} \cdots x_n^{d_{1,n}} + \cdots + a_s x_1^{d_{s,1}} \cdots x_n^{d_{s,n}} - b,$$

onde $a_i \in \mathbb{F}_q^*$ e $b \in \mathbb{F}_q$, com $d_{i,j} > 0$ para $1 < i, j < n$. Este capítulo investiga a contagem de soluções para esses polinômios quando eles estão relacionados a polinômios diagonais da forma

$$g(x_1,\dots,x_s) = a_1 x_1^d + \cdots + a_s x_s^d - b,$$

onde $n \ge s$, e os expoentes e coeficientes estão sujeitos a certas condições aritméticas.

Finalmente, no Capítulo 4, apresentamos vários resultados sobre palavras-código de peso 2 e 3 em códigos cíclicos, junto com um teorema que relaciona a contagem de soluções de sistemas de equações diagonais sobre corpos finitos com característica 2 à distribuição de peso dos códigos cíclicos binários correspondentes.

O Capítulo 2 baseia-se no conteúdo do nosso artigo publicado [1], enquanto os capítulos 3 e 4 apresentam o conteúdo dos artigos [2, 3] que ainda não foram publicados. No anexo, apresentamos comandos do SageMath usados para criar e verificar os exemplos que são discutidos ao longo da tese.

Palavras-chave: corpos finitos; equações diofantinas; equações sobre corpos finitos; códigos cíclicos; distribuição de pesos; palavras de peso baixo

Abstract

The first part of this thesis focuses on counting the number of solutions to Diophantine equations over finite fields for families that are closely related to diagonal equations, presenting significant results in this area. The second part of the thesis is centered on counting low-weight codewords in cyclic codes, presenting some novel results. The introductory chapter lays the groundwork by introducing fundamental concepts that serve as the basis for the subsequent chapters.

Chapter 2 studies fully triangular polynomials, characterized by expressions of the form

$$f(x_1,\dots,x_n) = a_1 x_1^{d_{1,1}} + a_2 x_1^{d_{1,2}} x_2^{d_{2,2}} + \cdots + a_n x_1^{d_{1,n}} \cdots x_n^{d_{n,n}} - b,$$

where $a_i \in \mathbb{F}_q^*$ and $b \in \mathbb{F}_q$, with $d_{i,j} > 0$ for all $1 \le i \le j \le n$. For these polynomials, explicit formulas for counting the number of solutions are derived under arithmetic conditions that relate them to diagonal polynomials of degrees 1 and 2.

Chapter 3 shifts the focus to full polynomials, defined as

$$f(x_1,\dots,x_n) = a_1 x_1^{d_{1,1}} \cdots x_n^{d_{1,n}} + \cdots + a_s x_1^{d_{s,1}} \cdots x_n^{d_{s,n}} - b,$$

where $a_i \in \mathbb{F}_q^*$ and $b \in \mathbb{F}_q$, with $d_{i,j} > 0$ for $1 < i, j < n$. This chapter investigates the solution count for these polynomials when they are related to diagonal polynomials of the form

$$g(x_1,\dots,x_s) = a_1 x_1^d + \cdots + a_s x_s^d - b,$$

where $n \ge s$, and the exponents and coefficients are subject to certain arithmetic conditions.

Lastly, in Chapter 4 we produce many results concerning weight 2 and 3 codewords in cyclic codes, along with a theorem linking the solution count of systems of diagonal equations over finite fields with characteristic 2 to the weight distribution of corresponding binary cyclic codes.

Chapter 2 is based on the contents of our published paper [1], while chapters 3 and 4 present the contents of articles [2, 3] that are yet to be published. In the annex, we present SageMath commands used to create and verify the examples that are discussed throughout the thesis.

Keywords: finite fields; diophantine equations; equations over finite fields; cyclic codes; weight distribution; low-weight codewords.

SUMMARY

CHAPTER 1

INTRODUCTION

"E agora, José?"
— Carlos Drummond de Andrade

Finding the number of solutions of equations over a fixed finite field is an ongoing question, which has seen considerable study. That is essentially the study of diophantine equations over finite fields, but this change from defining it over the integers to over finite fields made necessary the use of resourceful and distinct number theoretic methods.

Let $\mathbb{F}_q$ be a finite field with $q = p^m$ elements, where p is a prime and m a positive integer. Good results were obtained in the study of diagonal equations over a fixed finite field, i.e., equations of the form

$$a_1 x_1^{d_1} + a_2 x_2^{d_2} + \cdots + a_s x_s^{d_s} = b,$$

where $a_i \in \mathbb{F}_q^*$, $d_i \in \mathbb{Z}_{>0}$ for all $1 \leq i \leq s$ and $b \in \mathbb{F}_q$. In particular, the case where $d_i = 1$ for all $1 \leq i \leq s$ is trivial, and the case where $d_i \in \{1, 2\}$ for all $1 \leq i \leq s$ has been determined (see Chapter 10 of [12]). Other authors have studied the number of solutions of diagonal equations when the monomials produce pure Gauss sums [19], which is an arithmetic condition for which a character-based approach is straightforward. Some cases of this property being used to count the solutions of diagonal polynomials can be seen in [18].

Other authors have studied the number of solutions of specific equations that are related to diagonal equations. For instance, Carlitz [9] and Baoulina [8] studied equations

of the form

$$a_1x_1^{d_1} + a_2x_2^{d_2} + \cdots + a_sx_s^{d_s} = bx_1\cdots x_s, \tag{1.1}$$

where $d_j \in \mathbb{Z}_{>0}$, $a_j \in \mathbb{F}_q^*$ for all $1 \le j \le n$ and $b \in \mathbb{F}_q^*$. Another studied family of equations are the Markoff-Hurwitz's equations, i.e, those of the form

$$x_1^{m_1} + x_2^{m_2} + \cdots + x_n^{m_n} = bx_1^{t_1}x_2^{t_2}\ldots x_n^{t_n}. \tag{1.2}$$

where $m_j, t_j > 0$ for all $j = 1,\ldots,n$ and $b \in \mathbb{F}_q^*$. The equation in the case $m_1 = m_2 = \cdots = m_n$ and $t_1 = \cdots = t_n = 1$ defines a hypersurface known as Calabi-Yau's hypersurface which has been intensively studied by some authors (see [10, 11]). Other results about the number of solutions for the general equation (1.2) can be found in the literature; for instance, the number of solutions over $\mathbb{F}_q$ was calculated by Carlitz in the case where $\gcd(m\sum_{i=1}^n t_i/m_i - m, q-1) = 1$ and $m = m_1m_2\cdots m_n$. The case when $\gcd(m\sum_{i=1}^n t_i/m_i - m, q-1) > 1$ was considered by Cao, Jiang and Gao [4, 6] assuming some arithmetic conditions.

These cases are mostly sparse, as most of the terms have only one variable. In the other extreme are the full equations that have every variable in every term, except possibly for a constant term. Cao, Wen, and Wang [7] have determined the number of solutions for full equations of the form

$$a_1x_1^{d_{1,1}}\cdots x_n^{d_{1,n}} + \cdots + a_sx_1^{d_{s,1}}\cdots x_n^{d_{s,n}} = 0,$$

where $d_{i,j} > 0$, i.e., all exponents are positive, assuming that the matrix $(d_{i,j})_{i,j}$ is row-equivalent to a diagonal matrix D, when the elements in the diagonal of D are only 1's and 2's.

The first part of this thesis refers to two works of the author: the first one referring to the paper where the number of roots of polynomials of the form

$$f(x_1,\ldots,x_n) = a_1x_1^{d_{1,1}} + a_2x_1^{d_{2,1}}x_2^{d_{2,2}} + \cdots + x_1^{d_{n,1}}\cdots x_n^{d_{n,n}},$$

when the polynomials are related to linear or quadratic diagonal polynomials [1]. The second work on this line of research is about determining the number of solutions of equations of the form

$$a_1x_1^{d_{1,1}}\cdots x_n^{d_{1,n}} + \cdots + a_sx_1^{d_{s,1}}\cdots x_n^{d_{s,n}} - b, \quad a_i \in \mathbb{F}_q^*, b \in \mathbb{F}_q,$$

where all exponents are positive, and the matrix $(d_{i,j})_{i,j}$ is row-equivalent to a sparse matrix and the exponents satisfy an arithmetic condition [2].

The second part of this text refers to the work of the author on how to determine low-weight cyclic codewords and the proof of a recursive relation between the weight distribution of binary cyclic codes and the number of solutions of related systems of equations [3].

1.1 Characters

Let $(G,*)$ be an abelian group. A character of G is an homomorphism from G to the multiplicative group of complex numbers, that is, a mapping $\eta : G \to \mathbb{C}^*$ such that $\eta(g_1 * g_2) = \eta(g_1)\eta(g_2)$ for all $g_1, g_2 \in G$. For any character η we define its conjugate character as $\bar{\eta}(x) = \overline{\eta(x)}$, where the bar denotes complex conjugation.

Along the text, let p be a prime number, $q = p^m$ a power of p, and consider the finite field $\mathbb{F}_q$. For any positive integer d we define $\zeta_d = \exp\left(\frac{2\pi I}{d}\right)$, a fixed primitive $d-$th complex root of unity, where I is the imaginary unit.

In a finite field we can consider either the additive group itself or the multiplicative group $\mathbb{F}_q^*$. Characters over the additive group are called additive characters, and any additive character φ satisfies

$$\varphi(g_1 + g_2) = \varphi(g_1)\varphi(g_2), \quad \text{for all } g_1, g_2 \in \mathbb{F}_q.$$

Let

$$\begin{aligned} \mathrm{Tr} : \mathbb{F}_q &\to \mathbb{F}_p \\ x &\mapsto x^p + x^{p^2} + \cdots + x^{p^{m-1}}, \end{aligned}$$

be the trace function from $\mathbb{F}_q$ to $\mathbb{F}_p$. It is a classic result (see Theorem 5.7 in [15]) that all additive characters of $\mathbb{F}_q$ can be written as

$$\psi_a(x) := \zeta_p^{a\mathrm{Tr}(x)},$$

where $a \in \mathbb{F}_q$. We remark that the trivial additive character $\varphi_0(x)$ is simply defined as $\varphi_0(c) = 1$ for all $c \in \mathbb{F}_q$, and we will denote it as ψ_0. The character $\varphi_1(x)$ is defined as the canonical additive character of $\mathbb{F}_q$, and we will denote it as ψ.

Now, let us consider the multiplicative group $\mathbb{F}_q^*$. Characters on this group are called multiplicative characters. Since $\mathbb{F}_q^*$ is a cyclic group, it has a primitive generator γ. The multiplicative characters are defined as

$$\chi_j(\gamma^k) := \zeta_{q-1}^{jk}, \tag{1.3}$$

with $0 \leq j \leq q-2$.

It is a classic result that the group of multiplicative characters is cyclic with order $q-1$. Any multiplicative character η can be extended to an homomorphism from $\mathbb{F}_q$ to $\mathbb{C}$ while keeping the multiplicative property by defining $\eta(0) = 0$. So from now on, we will implicitly define multiplicative characters as maps from $\mathbb{F}_q$ to $\mathbb{C}$. The trivial multiplicative character is

$$\eta_1(c) := \begin{cases} 1, & \text{if } c \neq 0, \\ 0, & \text{if } c = 0. \end{cases}$$

The general definition for multiplicative characters in (1.3) won't be used in the text, as we find it more useful to fix multiplicative characters for each $d \mid (q-1)$, since doing so makes calculations clearer. First, let us introduce the discrete logarithm over a finite field, which will be used from here on.

Definition 1.1. *Let $\gamma \in \mathbb{F}_q$ be a primitive element. For $c \in \mathbb{F}_q^*$, we define the discrete logarithm of c in relation to γ as*

$$\log_\gamma(c) := \min\{n \in \mathbb{Z}_{\geq 0} : \gamma^n = c\},$$

i.e., the smallest non-negative integer n such that γ^n equals c.

We will denote $\log_\gamma$ as log when there is no ambiguity. For each positive integer d that divides $q-1$ we fix a primitive d-th complex root of unity ζ_d and define a multiplicative character $\eta_d : \mathbb{F}_q^* \to \mathbb{C}$ with order d as

$$\eta_d(x) = \zeta_d^{\log(x)}.$$

This definition does not apply to $d = 1$, which we define as the trivial multiplicative character η_1 where $\eta_1(x) = 1$ for all $x \in \mathbb{F}_q^*$. This will be the notation for multiplicative characters we will use going forward. We will also denote by $\omega(x)$ the multiplicative character

$$\omega(x) = \zeta_{q-1}^{\log(x)}, \tag{1.4}$$

which is a generator for the group of multiplicative characters. The following lemma will be useful.

Lemma 1.2. *Let $x \in \mathbb{F}_q$, then*

$$\sum_{j=0}^{d-1} \eta_d^j(x) = \begin{cases} d, & \text{if } \eta_d(x) = 1, \\ 0, & \text{otherwise.} \end{cases}$$

Proof. If $\eta_d(x) = 1$, the expression is simply adding 1 to itself d times, which equals d. If $\eta_d(x) \neq 1$, it is a geometric sum that evaluates to 0. ■

Similarly, the following theorem about character sums is important:

Theorem 1.3. *(Orthogonality relation.) If χ is a nontrivial character of a finite abelian group $(G, +)$, then*

$$\sum_{g \in G} \chi(g) = 0.$$

Proof. See Theorem 5.4 in [15]. ■

The orthogonality relation can be used with additive or multiplicative characters. The following corollary explicitly enumerates the forms it can be stated.

Corollary 1.4. *Let η and φ be respectively a multiplicative character and an additive character over $\mathbb{F}_q$. Then,*

$$\sum_{c \in \mathbb{F}_q^*} \eta(c) = \begin{cases} 0, & \text{if } \eta \neq \eta_1, \\ q-1, & \text{if } \eta = \eta_1, \end{cases}$$

and

$$\sum_{c \in \mathbb{F}_q} \varphi(c) = \begin{cases} 0, & \text{if } \varphi \neq \psi_0, \\ q, & \text{if } \varphi = \psi_0. \end{cases}$$

Proof. Follows directly from Theorem 1.3. ■

A consequence of this result is that we can use character sums to construct an indicator function. It can be done with either additive characters or multiplicative characters, but sums with additive characters have nicer properties and are the ones used in general.

Corollary 1.5. *Let φ be a nontrivial additive character over $\mathbb{F}_q$. Then*

$$\frac{1}{q} \sum_{c \in \mathbb{F}_q} \varphi(cx) = \begin{cases} 1, & \text{if } x = 0, \\ 0, & \text{if } x \neq 0. \end{cases}$$

Proof. Follows directly from Corollary 1.4. ■

This type of sum can be used as an indicator function for expressions over $\mathbb{F}_q$. For instance, if $f(x)$ is a function in $\mathbb{F}_q[x]$ then $a \in \mathbb{F}_q$ is a root of $f(x)$ if and only if

$$\frac{1}{q}\sum_{c\in\mathbb{F}_q}\psi(cf(a)) = 1.$$

Let $\mathbb{F}_q[x_1,\ldots,x_n]$ be the polynomial ring over $\mathbb{F}_q$ with n variables. For each $D = (d_1,\ldots,d_n) \in \mathbb{Z}^n_{\geq 0}$ we will use the notation $X^D = x_1^{d_1}\cdots x_n^{d_n}$. For a polynomial $f \in \mathbb{F}_q[x_1,\ldots,x_n]$ of the form

$$f(x_1,\ldots,x_n) = \sum_{j=1}^{s} a_j X^{D_j}, \tag{1.5}$$

where $D_j = (d_{1j},\ldots,d_{nj}) \in \mathbb{Z}^n_{\geq 0}$ and $a_j \neq 0$ for all $j = 1,\ldots,s$, we define $N(f)$ as the number of roots of $f(x_1,\ldots, x_n)$ over $\mathbb{F}_q^n$ and $N^*(f)$ as the number of roots over $(\mathbb{F}_q^*)^n$, i.e., roots without any coordinate equal to zero.

Corollary 1.5 allows us to use character sums as indicator functions to count the number of roots of a polynomial: let f be a polynomial of the form (1.5), the number of solutions over $(\mathbb{F}_q^*)^n$ is

$$\begin{aligned} N^*(f) &= \sum_{x_1,\ldots,x_n\in\mathbb{F}_q^*}\frac{1}{q}\sum_{c\in\mathbb{F}_q}\psi(cf(x_1,\ldots,x_n)) \\ &= \frac{1}{q}\sum_{c\in\mathbb{F}_q}\sum_{x_1,\ldots,x_n\in\mathbb{F}_q^*}\psi(cf(x_1,\ldots,x_n)). \end{aligned} \tag{1.6}$$

1.2 Gaussian sums

Let η be a multiplicative character over $\mathbb{F}_q^*$. We define the Gauss sum of η as

$$G(\eta) = \sum_{c\in\mathbb{F}_q^*}\psi(c)\eta(c). \tag{1.7}$$

The following inversion lemma involving Gauss sums will be useful.

Lemma 1.6. *For all $a \in \mathbb{F}_q^*$ the Gauss sums satisfy the interpolation relation*

$$\psi(a) = \sum_{v=0}^{q-2}\frac{G(\omega^v)}{q-1}\omega(a^{-1})^v.$$

Proof. See [6, Pages 278-279]. ■

There are some multiplicative characters for which the Gauss sum is known. The following definition will be useful.

Definition 1.7. *Let $q = p^m$ be a prime power. An integer $d \geq 3$ such that $d \mid (q-1)$ is (p,r)-admissible if r is the least positive integer satisfying $d \mid (p^r+1)$.*

Lemma 1.8. *Let $d \geq 3$ be a (p,r)-admissible integer. Then,*

$$G(\eta_d^j) = \begin{cases} q^{1/2}(-1)^{jh+h+1} & \text{if } 2 \mid d \text{ and } 2 \nmid (p^r+1)/d, \\ q^{1/2}(-1)^{h+1} & \text{if } 2 \nmid d \text{ or } 2 \mid (p^r+1)/d. \end{cases}$$

where $h = m/(2r)$.

Proof. See [19, Theorem 1]. ■

We will show how Gaussian sums are useful in order to calculate the number of solutions of equations over finite fields. For each equation

$$\sum_i a_i X^{D_i} = \sum_j b_j X^{D_j},$$

we define its corresponding polynomial $f(X) \in \mathbb{F}_q[x_1,\ldots,x_n]$ as

$$f(X) = \sum_i a_i X^{D_i} - \sum_j b_j X^{D_j}.$$

This turns the problem of finding the number of solutions of an equation into a problem of finding the number of roots of a polynomial. Hence we will exclusively use polynomial notation going forward.

Let f be a polynomial of the form (1.5). We define its degree matrix as $D_f = (D_1^T, \cdots, D_s^T)$ and the augmented degree matrix of f as $\tilde{D}_f = (\tilde{D_1}^T, \ldots, \tilde{D}_s^T)$, where $\tilde{D}_j = (1, D_j)$.

Let ω be a generator of the multiplicative characters group as seen in (1.4), i.e., $\widehat{\mathbb{F}_q^*} = \{\omega^k : k = 0,1,\ldots,q-2\}$. The following result allows us to express $N^*(f)$ in terms of the augmented degree matrix and multiplicative characters:

Lemma 1.9. *Let f be a polynomial of the form (1.5), then*

$$N^*(f) = \frac{(q-1)^n}{q} + \frac{(q-1)^{n+1-s}}{q} \sum \prod_{j=1}^{s} G(\omega^{v_j}) \omega(a_j^{-1})^{v_j},$$

where the sum is taken over all vectors $v = (v_1,\ldots,v_s)$ with $0 \leq v_i \leq q-2$ for $i = 1,\ldots,s$ such that $\tilde{D}_f v^T \equiv 0 \pmod{q-1}$.

Proof. The proof can be seen in [4, Lemma 2.5], but we will repeat it here because this result is very important. Since f is of the form (1.5), from (1.6) we have

$$\begin{aligned} qN^*(f) &= \sum_{x_0\in\mathbb{F}_q, x_1,\ldots,x_n\in\mathbb{F}_q^*} \psi(a_j X^{\tilde{D}_j}) \\ &= (q-1)^n + \sum_{x_0,x_1,\ldots,x_n\in\mathbb{F}_q^*} \prod_{j=1}^{m} \psi(a_j X^{\tilde{D}_j}). \end{aligned}$$

Using Lemma 1.6 to rewrite the additive character sum as a sum of Gauss sums we obtain:

$$\begin{aligned} qN^*(f) &= (q-1)^n + \sum_{x_0,x_1,\ldots,x_n\in\mathbb{F}_q^*} \prod_{j=1}^{s} \sum_{v_j}^{q-2} \frac{G(\omega^{v_j})}{q-1}\omega(a_j^{-1})^{v_j}\omega(X^{-\tilde{D}_j})^{v_j} \\ &= (q-1)^n + \sum_{v_1=0}^{q-2}\cdots\sum_{v_s}^{q-2}\left(\prod_{j=1}^{s}\frac{G(\omega^{v_j})}{q-1}\omega(a_j^{-1})^{v_j}\right)\cdot \sum_{x_0,x_1,\ldots,x_n\in\mathbb{F}_q^*}\omega(X^{-v_1\tilde{D}_1-\cdots-v_m\tilde{D}_m}). \end{aligned}$$

The sum

$$\sum_{x_0,x_1,\ldots,x_n\in\mathbb{F}_q^*}\omega(X^{-v_1\tilde{D}_1-\cdots-v_m\tilde{D}_m})$$

can be rewritten as

$$\sum_{x_0\in\mathbb{F}_q^*}\omega(x_0^{-v_1-\cdots-v_m})\sum_{x_1\in\mathbb{F}_q^*}\omega(x_1^{-v_1d_{1,1}-\cdots-v_md_{m,1}})\cdots\sum_{x_n\in\mathbb{F}_q^*}\omega(x_n^{-v_1d_{1,n}-\cdots-v_md_{m,n}}), \tag{1.8}$$

where $d_{i,j}$ for $1\le i\le m$ are the elements in the augmented degree matrix $\tilde{D}_f$. We notice that if all exponents in (1.8) are multiples of $q-1$, then Corollary 1.4 implies that the expression (1.8) will evaluate to $(q-1)^{n+1}$. Likewise, if any of the exponents inside of the sums is not divisible by $q-1$, then this entire expression vanishes. Thus,

$$qN^*(f) = (q-1)^n + (q-1)^{n+1-m}\sum\prod_{j=1}^{m} G(\omega^{v_j})\omega(a_j^{-1})^{v_j},$$

where the sum is restricted only to the values of $v_1,\ldots,v_m$ such that every exponent is a multiple of $q-1$, i.e., $v_1\tilde{D}_1+\cdots+v_m\tilde{D}_m\equiv 0 \pmod{q-1}$.

■

To explain the utility of this result, we will introduce some important definitions.

Definition 1.10. *Two polynomials* $f = \sum_{j=1}^{s} a_j X_j^D$ *and* $g = \sum_{j=1}^{s} a_j X^{D'_j}$ *are said to be* $*$*-equivalent if they have the same coefficient vector* $(a_1, \ldots, a_s)$ *and the congruences* $\tilde{D}_f v^T \equiv 0 \pmod{q-1}$ *and* $\tilde{D}_g v^T \equiv 0 \pmod{q-1}$ *have the same set of solutions.*

The reason why Lemma 1.9 is useful is that if f and g are $*$-equivalent polynomials, then $N^*(f) = N^*(g)$. That is, they have the same number of roots with all coordinates in $\mathbb{F}_q^*$, which we will call $*$-roots. This is because each coefficient a_j, $1 \leq j \leq m$ will be the same and the sum in Lemma 1.9 will be over the same number of set of vectors, so the expression is the same.

For the purpose of proving $*$-equivalence, it is easy to check if the coefficient vectors of two polynomials are equal, but it is more difficult to check if two linear systems $\tilde{D}_f v^T = 0$ and $\tilde{D}_g v^T = 0$ have the same set of solutions.

It can be verified that two matrices D and E with coefficients in $\mathbb{Z}_{q-1}$ such that $Dv^T \equiv 0 \pmod{q-1}$ and $Ev^T \equiv 0 \pmod{q-1}$ have the same set of solutions if there is an invertible matrix M over $\mathbb{Z}_{q-1}$ such that $MD = E$, and in this case, we say that D and E are row-equivalent.

Hence, if two polynomials f and g have the same coefficient vector and there is an invertible matrix M over $\mathbb{Z}_{q-1}$ such that $M\tilde{D}_f = \tilde{D}_g$, then f and g are $*$-equivalent. This is a sufficient condition. Row-equivalence between D and E over $\mathbb{Z}_{q-1}$ is not strictly necessary for $Dv^T \equiv 0 \pmod{q-1}$ and $Ev^T \equiv 0 \pmod{q-1}$ to have the same solution set.

In particular, the elementary row operations can be represented by multiplying invertible matrices, so if we can apply elementary operations on the matrix $\tilde{D}_f$ to obtain $\tilde{D}_g$, then the linear systems have the same solutions. This is a sufficient criterion to prove $*$-equivalency. In particular, the elementary row operations are

(i) swapping two rows;

(ii) adding a multiple of a row to another;

(iii) multiplying a row by an element in $\mathbb{Z}_{q-1}^*$;

which can be represented by multiplying invertible matrices, so if we can apply these operations in $\tilde{D}_f$ to obtain $\tilde{D}_g$, the congruence systems have the same solutions.

We remark that even though $N^*(f) = N^*(g)$ for two $*$-equivalent polynomials f and g, that does not mean they have the same set of $*$-roots. For instance, the polynomials $f(x,y) = x^2y^3 + xy^2$ and $g(x,y) = xy + x^3y^2$ in $\mathbb{F}_5[x,y]$ are $*$-equivalent, but it can be verified that $(2,2)$ is a root of f but not a root of g.

We also remark that two polynomials being $*$-equivalent does not mean that they have the same number of roots in $\mathbb{F}_q^n$. For instance, the polynomials $f(x,y,z) = 11x^{13} + 5x^{21}y^{19} + 12x^2y^3z^{17}$ and $g(x,y,z) = 11x + 5y + 12z$ are $*$-equivalent in $\mathbb{F}_{31}[x,y,z]$, thus $N^*(f) = 870 = N^*(g)$. However, it can be verified that $N(f) = 1861 \neq 961 = N(g)$.

1.3 Linear and quadratic polynomials

Let us consider a linear polynomial over $\mathbb{F}_q$ given by

$$g(x_1,\ldots,x_n) = a_1x_1 + \cdots + a_nx_n - b, \tag{1.9}$$

where $a_1,\ldots,a_n \in \mathbb{F}_q^*$ and $b \in \mathbb{F}_q$.

Lemma 1.11. *For a linear polynomial g of the form (1.9), the number of roots of g satisfies*

$$N(g) = q^{n-1}, \tag{1.10}$$

and

$$N^*(g) = \begin{cases} \frac{(q-1)^n}{q} - \frac{(-1)^n}{q}, & \text{if } b \neq 0, \\ \frac{(q-1)^n}{q} - \frac{(-1)^n}{q} + (-1)^n, & \text{if } b = 0. \end{cases} \tag{1.11}$$

Proof. For $N(g)$, see [12, Theorem 10.0.2]. The proof is given for $\mathbb{F}_p$, but the case over $\mathbb{F}_q$ is similar. The value of $N^*(g)$ is obtained using the inclusion–exclusion principle, applying the value formula for $N(g)$. ■

A quadratic diagonal polynomial is a diagonal polynomial where every term has degree 2, except for a possible constant term.

$$g(x_1,\ldots,x_n) = a_1x_1^2 + \cdots + a_nx_n^2 - b, \tag{1.12}$$

where $a_1,\ldots,a_n \in \mathbb{F}_q^*$ and $b \in \mathbb{F}_q$. The following result about quadratic forms is classic.

Theorem 1.12. *Let $\mathbb{F}_q$ be a finite field, where q is odd, and g a polynomial as in (1.12). The number of roots of $g(x_1,\ldots,x_n)$ in $\mathbb{F}_q^n$ is*

$$N(g) = \begin{cases} q^{n-1} - \eta_2((-1)^{n/2}a_1\cdots a_n)q^{(n-2)/2}, & \text{if } n \text{ even and } b \neq 0, \\ q^{n-1} + \eta_2((-1)^{(n-1)/2}ba_1\cdots a_n)q^{(n-1)/2}, & \text{if } n \text{ odd and } b \neq 0, \\ q^{n-1} + \eta_2((-1)^{n/2}a_1\cdots a_n)(q^{n/2} - q^{(n-2)/2}), & \text{if } n \text{ even and } b = 0, \\ q^{n-1}, & \text{if } n \text{ odd and } b = 0, \end{cases} \tag{1.13}$$

where η_2 is the quadratic multiplicative character in $\mathbb{F}_q$.

Proof. See Theorem 10.5.1 in [12]. ■

We notice that this result implies that the number of roots only depends on the values of $\eta(a_j)$ for $1 \leq j \leq n$ and $\eta(b)$.

1.4 Linear codes

The typical use of codes is to introduce redundancy on a message that will be transferred through a noisy channel that can corrupt individual bits of information. Typically the messages are divided into codewords of fixed length and multiplied by the generator matrix to be transformed into vectors of greater length. The introduced redundancy allows some algorithm proper to the type of code to extract the original message even if a few bits of it are corrupted. The minimum distance is a very important parameter, as it is the minimum degree of "separation" between codewords in the code, and the greater it is the more errors can be corrected. In this text we will talk about linear codes, i.e., codes that are vector subspaces.

Let F be a field, a linear code of length n is a vector subspace of F^n. For a given codeword $\bar{c} = (c_0, \ldots, c_{n-1}) \in F^n$, we define the weight of $\bar{c}$ as

$$w(\bar{c}) = \#\{0 \leq i \leq n-1 : c_i \neq 0\},$$

i.e., the number of coordinates in the codeword that are different from 0. The weight distribution of a code with length n is the sequence of non-negative integers $A_0, A_1, A_2, \ldots, A_n$ where A_w is the number of codewords in the code with weight equal to w. We remark that in a linear code the zero codeword is always present, so $A_0 = 1$.

Let us consider a finite field $\mathbb{F}_q$ and $C \subset \mathbb{F}_q^n$ a linear code. The parameters of the linear code are the tuple of integers (n, k, d) where k is the dimension of C over $\mathbb{F}_q$ and d represents the minimum distance of C, that is defined as the lowest possible weight for a non-zero codeword in the code. We remark that the minimum weight is the lowest positive integer d such that $A_d > 0$.

Let $B = \{\bar{c_1}, \ldots, \bar{c_k}\}$ be a basis of C and consider the matrix G where the rows are the vectors $\bar{c_i} = (c_{i,1}, \ldots, c_{i,n})$, $i = 1, \ldots, k$, that is,

$$G = \begin{bmatrix} \bar{c_1} \\ \vdots \\ \bar{c_k} \end{bmatrix} = \begin{bmatrix} c_{1,1} & c_{1,2} & \ldots & c_{1,n} \\ \vdots & \vdots & & \vdots \\ c_{k,1} & c_{k,2} & \ldots & c_{k,n} \end{bmatrix}.$$

The matrix G is called the generator matrix of C associated with the basis B. This matrix defines a bijection from $\mathbb{F}_q^k$ to C by matrix multiplication, i.e., every codeword in C can be represented uniquely as

$$(v_1,\dots,v_k)G = v_1\bar{c_1} + \cdots + v_k\bar{c_k},$$

for $(v_1,\dots,v_k) \in \mathbb{F}_q^k$.

While the generator matrix allows for encoding, it is not very efficient for determining if a codeword $\bar{c}$ is part of the code, as that would require checking if the system $\bar{v}G = \bar{c}$ has any solution. For that, we will need to introduce the concept of a parity check matrix.

For a linear code C of length n, we define its dual code as

$$C^\perp = \{\bar{v} \in \mathbb{F}_q^n : \bar{c}\cdot\bar{v} = 0, \forall \bar{c} \in C\},$$

where $\cdot$ denotes the vector dot product. It is well known (see Theorem 1.5.7 in [21]) that $C^\perp$ is a vector subspace of $\mathbb{F}_q^n$, that $\dim C^\perp = n-k$, and $\bar{v} \in C^\perp$ if and only if $G\bar{v}^T = 0$. The following result is important:

Lemma 1.13. *Let C be a linear code, and suppose H is a generator matrix of $C^\perp$. Then $\bar{c} \in C$ if and only if $H\bar{c}^T = 0$.*

Proof. This is a consequence of Theorem 1.2.1 in [21]. ■

The generator matrix of $C^\perp$ is called parity check matrix and allows us to characterize elements in a code G through a straighforward matrix multiplication.

1.5 Cyclic codes

A linear code $C \subset \mathbb{F}_q^n$ is called a cyclic code if for every $\bar{c} = (c_0,\dots,c_{n-1})$ in C the shifted vector $(c_{n-1},c_0,\dots,c_{n-2})$ is also in C. Let $\mathbb{F}_q[x]$ be the polynomial ring with coefficients in $\mathbb{F}_q$ and let us define the quotient ring $R_n = \mathbb{F}_q[x]/(x^n-1)$. The bijection

$$T(\bar{c}) := [c(x)] = [c_0 + c_1x + \cdots + c_{n-1}x^{n-1}]$$

assigns to each codeword in C a class of polynomial in R_n. Since $1,[x],\dots,[x^{n-1}]$ is a basis for R_n as a vector space over $\mathbb{F}_q$, this is an isomorphism between $\mathbb{F}_q^n$ and R_n. The action of shifting the code one coordinate to the right is translated in R_n as a multiplication of the representative polynomial by x. The following lemma then provides another structure for cyclic codes.

Lemma 1.14. *Let V be a vector subspace of R_n. Then, V is an ideal of the ring R_n if and only if V is invariant under multiplication by $[x]$.*

Proof. See the proof of Theorem 4.2.1 in [21]. ■

Thus, a subspace C of $\mathbb{F}_q^n$ is a cyclic code if and only if $T(C)$ is an ideal in R_n. To characterize a cyclic code as an ideal, we use the following lemma:

Lemma 1.15. *Every ideal of $\mathbb{F}_q[x]/p(x)$, where $p(x)$ is a polynomial, is of the form $I = ([f(x)])$, where $f(x)$ is a divisor of $p(x)$.*

Proof. This follows from Corollary 4.2.2 in [21]. ■

Thus, every cyclic code is isomorphic to an ideal in R_n generated by some divisor $g(x)$ of $x^n - 1$. This $g(x)$ is called the generator polynomial. Therefore, a codeword $c(x)$ is in the code generated by $g(x)$ if and only if it is divisible by $g(x)$.

This condition is easier to check using the parity check polynomial

$$h(x) = \frac{x^n - 1}{g(x)}.$$

It has the important property that a codeword $c(x) \in R_n$ is in the code generated by $g(x)$ if and only if $[h(x)] \cdot [c(x)] = [0]$, since that would imply $g(x)$ divides it.

For a codeword $\bar{c}$ such that $T(\bar{c}) = [c_0 + c_1 x + \cdots + c_{n-1}x^{n-1}]$, we will denote the class representative $c(x) = c_0 + c_1 x + \cdots + c_{n-1}x^{n-1}$ as the polynomial form of $\bar{c}$.

1.6 Parity check matrices

Let m be a positive integer, $q = p^m$ and let us consider the cyclic codes over $\mathbb{F}_p$ with length $n = p^m - 1$. Let γ be a primitive element of $\mathbb{F}_q$ and let us define $g_a(x)$ as the minimal polynomial over $\mathbb{F}_p[x]$ of γ^a for $0 \le a \le q - 2$. As $g_a(x)$ is a minimal polynomial its other roots are conjugates of γ^a in the field extension $\mathbb{F}_q/\mathbb{F}_p$, i.e., the set $\{\gamma^b : b \equiv p^j a \pmod{q-1}\}$ where the exponents of γ are in the same $p-$cyclotomic class of a.

For integers $t_1, \ldots, t_s$ representing different p-cyclotomic classes, let us define the code $C_{t_1,\ldots,t_s}$ as the code generated by the polynomial $g(x) = g_{t_1}(x)g_{t_2}(x)\cdots g_{t_s}(x)$.

Theorem 1.16. *Let $C_{t_1,\dots,t_s}$ be a cyclic code of length n and $\bar{c}$ be a codeword with polynomial form $c(x)$. Let us define the matrix*

$$H := \begin{bmatrix} 1 & \gamma^{t_1} & \gamma^{2t_1} & \cdots & \gamma^{(n-1)t_1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \gamma^{t_s} & \gamma^{2t_s} & \cdots & \gamma^{(n-1)t_s} \end{bmatrix}. \tag{1.14}$$

The following are equivalent:

(i) $\bar{c}$ *is in* $C_{t_1,\dots,t_s}$.

(ii) $c(\gamma^{t_j}) = 0$ *for* $1 \le j \le s$.

(iii) $H\bar{c}^T = 0$.

Proof. Since the code is generated by $g(x) = g_{t_1}(x)\cdots g_{t_s}(x)$, a codeword is in the code if and only if it is divisible by $g(x)$ in polynomial form. But a polynomial is divisible by $g(x)$ if and only if it has $\gamma^{t_1},\dots,\gamma^{t_s}$ as roots.

The condition $g_1(\gamma^{t_j}) = 0$ can be rewritten as

$$\begin{bmatrix} 1 & \gamma^{t_j} & \gamma^{2t_j} & \dots & \gamma^{(n-1)t_j} \end{bmatrix} \bar{c}^T = 0.$$

Hence, a codeword $\bar{c}$ is in the code if and only if $H\bar{c}^T = 0$.

■

We notice that the matrix H in (1.14) is not a parity check matrix for the code, since the entries are in $\mathbb{F}_q$ instead of $\mathbb{F}_p$. However, it can be converted into a parity check matrix by fixing a base for $\mathbb{F}_q$ over $\mathbb{F}_p$, turning each row into m rows where each coordinate in the i-th row is the coefficient of the corresponding element of the original row on the i-th basis element, and then excluding rows until the matrix is linearly independent. Hence, we will also refer to such a matrix as the parity check matrix.

CHAPTER 2

TRIANGULAR POLYNOMIALS

As mentioned in the introduction, many authors have studied various techniques to count the number of solutions of equations over finite fields. In particular, Cao, Wen, and Wang [7] have determined the number of solutions in $\mathbb{F}_q^n$ for equations of the form

$$a_1 x_1^{d_{1,1}} \cdots x_n^{d_{1,n}} + \cdots + a_n x_1^{d_{n,1}} \cdots x_n^{d_{n,n}} = 0,$$

where $d_{i,j} > 0$, i.e., all exponents are positive, assuming that the matrix $(d_{i,j})_{i,j}$ is row equivalent to a diagonal matrix D, with the diagonal elements of D being only 1's and 2's.

In this chapter we will determine the number of solutions of equations

$$a_1 x_1^{d_{1,1}} + a_2 x_1^{d_{1,2}} x_2^{d_{2,2}} + \cdots + a_n x_1^{d_{1,n}} \cdots x_n^{d_{n,n}} = b,$$

where $d_{i,j} > 0$ for all $1 \le i \le j \le n$ and the exponents $d_{i,j}$ satisfy certain arithmetical conditions. We show sufficient conditions for the equation to have the same number of solutions in $(\mathbb{F}_q^*)^n$ as a simpler equation. Specifically, we consider the case when the equations are $*$-equivalent. Then, we use this equivalence to calculate the total number of solutions. The content of this chapter is part of the paper [1], written by the author of this thesis.

2.1 Triangular polynomials

Let f be a polynomial in $\mathbb{F}_q[x_1,\dots,x_n]$, and let us define $f_k \in \mathbb{F}_q[x_1,\dots,x_k]$ as

$$f_k(x_1,\dots,x_k) = f(x_1,\dots,x_k,0,\dots,0).$$

We say that f and g are totally $*$-equivalent if f_k is $*$-equivalent to g_k for all $1 \le k \le n$. In general, it is not true that f being $*$-equivalent to g implies that f_k is $*$-equivalent to g_k for all k.

Let us introduce a class of polynomials for which a sufficient criterion for total $*$-equivalence can be determined. We say that $f \in \mathbb{F}_q[x_1,\dots,x_n]$ is a triangular polynomial if it is of the form

$$f(x_1,\dots,x_n) = \sum_{i=1}^{n} a_j X^{D_j} - b; \quad a_1,\dots,a_n \in \mathbb{F}_q^*, b \in \mathbb{F}_q, \tag{2.1}$$

where $D_j = (d_{1,j},\dots,d_{j,j},0,\dots,0)$, $d_{j,j} > 0$ for all $1 \le j \le n$. If we additionally have that $d_{i,j} > 0$ for all $1 \le i \le j \le n$, we refer to this polynomial as a *fully* triangular polynomial.

Lemma 2.1. *Let $f,g \in \mathbb{F}_q[x_1,\dots,x_n]$ be two $*$-equivalent triangular polynomials, and suppose there is an invertible $(n+1)\times(n+1)$ matrix M over $\mathbb{Z}_{q-1}$ such that $M\tilde{D}_f = \tilde{D}_g$. For each integer $0 \le k \le n$, let us denote M_k as the submatrix obtained from M by selecting the first $k+1$ rows and columns. If M_k is invertible, then f_k and g_k are also $*$-equivalent.*

Proof. Since f and g are triangular polynomials, we can partition their degree matrices and the matrix M into blocks

$$\tilde{D}_f = \begin{bmatrix} \tilde{D}_{f_k} & D_1 \\ 0 & D_2 \end{bmatrix}, \qquad \tilde{D}_g = \begin{bmatrix} \tilde{D}_{g_k} & E_1 \\ 0 & E_2 \end{bmatrix}, \qquad M = \begin{bmatrix} M_k & N_1 \\ N_2 & N_3 \end{bmatrix},$$

such that $\tilde{D}_{g_k}$ and $\tilde{D}_{f_k}$ are $(k+1)\times k$ blocks, M_k is a $(k+1)\times(k+1)$ block and the blocks $D_1,D_2,E_1,E_2,N_1,N_2,N_3$ have appropriate dimensions. From the $*$-equivalency between f and g, we know that

$$\begin{aligned} \begin{bmatrix} \tilde{D}_{g_k} & E_1 \\ 0 & E_2 \end{bmatrix} = \tilde{D}_g &= M\tilde{D}_f \\ &= \begin{bmatrix} M_k & N_1 \\ N_2 & N_3 \end{bmatrix} \begin{bmatrix} \tilde{D}_{f_k} & D_1 \\ 0 & D_2 \end{bmatrix} \\ &= \begin{bmatrix} M_k\tilde{D}_{f_k} & M_kD_1 + N_1D_2 \\ N_2\tilde{D}_{f_k} & N_2D_1 + N_3D_2 \end{bmatrix}, \end{aligned}$$

where considering the equality of the upper left block gives us $M_k\tilde{D}_{f_k} = \tilde{D}_{g_k}$, implying that f_k and g_k are $*$-equivalent because M_k is invertible. ■

Hence, if f and g are two $*$-equivalent triangular polynomials, with $M\tilde{D}_f = \tilde{D}_g$ and the submatrices M_k are invertible for every $1 \leq k \leq n-1$, then f and g are totally $*$-equivalent. We now present a specific set of operations that always result in transformation matrices satisfying these conditions.

Lemma 2.2. *Let $f \in \mathbb{F}_q[x_1,\ldots,x_n]$ be a triangular polynomial. Let us denote by $r_1,\ldots,r_{n+1}$ the rows in $\tilde{D}_f$ and consider the following invertible row operations:*

(i) $r_i \leftarrow c \cdot r_i$, $2 \leq i \leq n+1$, $c \in \mathbb{Z}_{q-1}^*$.

(ii) $r_j \leftarrow r_j + c \cdot r_i$, $2 \leq j < i$, $c \in \mathbb{Z}_{q-1}$.

Any $*$-equivalency obtained using only these row operations is totally $*$-equivalent.

Proof. Any matrix M obtained from those operations is of the form

$$M = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & m_{1,1} & m_{1,2} & \cdots & m_{1,n-1} & m_{1,n} \\ 0 & 0 & m_{2,2} & \cdots & m_{2,n-1} & m_{2,n} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & m_{n-1,n-1} & m_{n-1,n} \\ 0 & 0 & 0 & \cdots & 0 & m_{n,n} \end{bmatrix}, \tag{2.2}$$

where, $m_{i,j} \in \mathbb{Z}_{q-1}$ and the elements on the diagonal are invertible. For every $1 \leq k < n-1$, the determinant of M_k is $\prod_{i=1}^{k} m_{i,i}$, which is invertible over $\mathbb{Z}_{q-1}$. Thus, every M_k is invertible, making the $*$-equivalency total.

■

Although one might believe that all complete equivalences between triangular matrices can be attained using those two operations, this assumption is not correct. For instance, let $f = x_1 + x_1^3 x_2^5$, $g = x_1^2 + x_1^4 x_2 \in \mathbb{F}_7[x_1, x_2]$. These are totally $*$-equivalent polynomials, i.e., $M\tilde{D}_f = \tilde{D}_g$ where M is the invertible matrix

$$M = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 5 \end{bmatrix},$$

but from a straightforward calculation it can be proved that there is no invertible upper triangular matrix N that satisfies $N\tilde{D}_f = \tilde{D}_g$. The following result tells us when a triangular polynomial is totally $*$-equivalent to a diagonal polynomial using the two operations given in Lemma 2.2.

Theorem 2.3. *Let f be a triangular polynomial of the form (2.1), and let*

$$g(x_1,\ldots,x_n) = a_1x_1^{e_1} + \cdots + a_nx_n^{e_n} - b,$$

be a diagonal polynomial where $e_1,\ldots,e_n \in \mathbb{Z}_{>0}$. Then f is totally $*$-equivalent to g if the following two conditions are true:

(i) for all $1 \le j \le n$ there is a $m_{j,j} \in \mathbb{Z}_{q-1}^*$ such that $d_{j,j} = m_{j,j}e_j$,

(ii) for all $1 \le i < j \le n$ we have $\gcd(d_{j,j}, q-1) \mid d_{i,j}$.

Proof. The augmented degree matrices of f and g are

$$\tilde{D}_f = \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 & 1 & 1 \\ d_{1,1} & d_{1,2} & d_{1,3} & \cdots & d_{1,n-1} & d_{1,n} & 0 \\ 0 & d_{2,2} & d_{2,3} & \cdots & d_{2,n-1} & d_{2,n} & 0 \\ 0 & 0 & d_{3,3} & \cdots & d_{3,n-1} & d_{3,n} & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & d_{n-1,n-1} & d_{n-1,n} & 0 \\ 0 & 0 & 0 & \cdots & 0 & d_{n,n} & 0 \end{bmatrix}, \quad \tilde{D}_g = \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 & 1 & 1 \\ e_1 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & e_2 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 0 & e_3 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & e_{n-1} & 0 & 0 \\ 0 & 0 & 0 & \cdots & 0 & e_n & 0 \end{bmatrix},$$

where the last columns are present only if $b \neq 0$.

Let us suppose that conditions (i) and (ii) are true. Since (i) implies that there is an element $m_{j,j} \in \mathbb{Z}_{q-1}^*$ such that $d_{j,j} = m_{j,j}e_j$, condition (ii) becomes $\gcd(e_jm_{j,j}, q-1) \mid d_{i,j}$. As $\gcd(m_{j,j}, q-1) = 1$, that implies condition (ii) is equivalent to $(e_j, q-1) \mid d_{i,j}$, which is in turn equivalent to the existence of $m_{i,j} \in \mathbb{Z}_{q-1}$ such that $d_{i,j} = m_{i,j}e_j$ over $\mathbb{Z}_{q-1}$. We can then use these values of $m_{i,j}$ to construct an invertible matrix M of the form (2.2), such that when we multiply M by $\tilde{D}_g$ it gives us:

$$M\tilde{D}_g = \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 & 1 & 1 \\ m_{1,1}e_1 & m_{1,2}e_2 & m_{1,3}e_3 & \cdots & m_{1,n-1}e_{n-1} & m_{1,n}e_n & 0 \\ 0 & m_{2,2}e_2 & m_{2,3}e_3 & \cdots & m_{2,n-1}e_{n-1} & m_{2,n}e_n & 0 \\ 0 & 0 & m_{3,3}e_3 & \cdots & m_{3,n-1}e_{n-1} & m_{3,n}e_n & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & m_{n-1,n-1}e_{n-1} & m_{n-1,n}e_n & 0 \\ 0 & 0 & 0 & \cdots & 0 & m_{n,n}e_n & 0 \end{bmatrix},$$

that is equal to $\tilde{D}_f$. ■

For the specific cases where the diagonal polynomials are linear or quadratic this criterion is simpler.

Corollary 2.4. *Let f be a triangular polynomial of the form (2.1), and let*

$$g(x_1,\dots,x_n) = a_1x_1^e + \cdots + a_nx_n^e - b,$$

be a diagonal polynomial, where $e \in \mathbb{Z}_{>0}$.

a) If $e = 1$ and $\gcd(d_{j,j}, q-1) = 1$ for all $1 \le j \le n$, then f is totally $*$-equivalent to g.

b) If $e = 2$, q is odd, and there exists an $m_{j,j} \in \mathbb{Z}_{q-1}^*$ such that $d_{j,j} = 2m_{j,j}$ for all $1 \le j \le n$ and that $2 \mid d_{i,j}$ for all $1 \le i < j \le n$, then f is totally $*$-equivalent to g.

Proof. The statements in each item imply the conditions given in Theorem 2.3. In fact,

a) If $\gcd(d_{j,j}, q-1) = 1$, then condition (ii) is always verified and $d_{j,j}$ is invertible, verifying condition (i).

b) The statement that there is an invertible $m_{j,j}$ in $\mathbb{Z}_{q-1}$ such that $d_{j,j} = 2m_{j,j}$ for all $1 \le j \le n$ is, in this case, equivalent to condition (i). Since $m_{j,j}$ is invertible, we have $\gcd(m_{j,j}, q-1) = 1$, which implies that $\gcd(2, q-1) = \gcd(2m_{j,j}, q-1) = \gcd(d_{j,j}, q-1)$. Considering q odd, we have $\gcd(2, q-1) = 2$, and the statement $2 \mid d_{i,j}$ for all $1 \le i < j \le n$ is equivalent to condition (ii).

■

Let f be a fully triangular polynomial. For any root $(c_1, c_2, \dots, c_n)$ of f, if $c_j = 0$ and j is the smallest index that satisfies this condition, then $f(c_1,\dots,c_{j-1},0,c'_{j+1},\dots,c'_n) = 0$ for any $c'_{j+1},\dots,c'_n \in \mathbb{F}_q$. This is due to the fact that the terms involving the variables $x_{j+1},\dots,x_n$ vanish, thereby not impacting the value of the polynomial. Thus, by adding over the indices of the first coordinates that are equal to 0 among the roots, we derive the following identity:

$$N(f) = \begin{cases} N^*(f) + \sum_{k=1}^{n-1} N^*(f_k) q^{n-k-1}, & \text{if } b \neq 0, \\ q^{n-1} + N^*(f) + \sum_{k=1}^{n-1} N^*(f_k) q^{n-k-1}, & \text{if } b = 0. \end{cases} \tag{2.3}$$

Now, let f be totally $*$-equivalent to a polynomial g. From Lemma 1.9, we have that $N^*(f_k) = N^*(g_k)$ for all $1 \le k \le n$. Thus,

$$N(f) = \begin{cases} N^*(g) + \sum_{k=1}^{n-1} N^*(g_k) q^{n-k-1}, & \text{if } b \neq 0, \\ q^{n-1} + N^*(g) + \sum_{k=1}^{n-1} N^*(g_k) q^{n-k-1}, & \text{if } b = 0. \end{cases} \tag{2.4}$$

Therefore, if we know that f is totally $*$-equivalent to a polynomial g, and $N^*(g_k)$ is known for any k, we can substitute these values into (2.4) to compute $N(f)$.

For instance, let us consider the polynomials $f, g \in \mathbb{F}_{10007}[x,y]$ given by $f(x,y,z) = x^{1001}+x^{2001}y^{3001}+x^{4001}y^{5001}z^{6001}+7001$ and $g(x,y,z) = x+y+z+7001$. By straightforward calculation, we can verify that f is totally $*$-equivalent to g, so

$$\begin{aligned} N(f) &= N^*(x+y+z+7001) + N^*(x+y+7001) + qN^*(x+7001) \\ &= 100110031 + 10005 + 10007 \cdot 1 \\ &= 100130043. \end{aligned}$$

2.2 Roots with non-zero coordinates for diagonal polynomials

The number of solutions $N^*(g_k)$ when g is a linear polynomial of the form (1.9) is as seen in Lemma 1.11, so the case where f is totally $*$-equivalent to a linear polynomial can be solved using this result and (2.4).

For the case when f is totally $*$-equivalent to a quadratic diagonal polynomial g of the form (1.12) , we will need a way to compute the number of roots of g_k, $1 \le k \le n$ in $(\mathbb{F}_q^*)^k$. We remark that Theorem 1.12 allows us to calculate $N(g_k)$ for any k. We will need the following definitions and results in order to calculate $N^*(g_k)$ for any k.

Definition 2.5. *Let η_2 be the quadratic character in $\mathbb{F}_q$. For a coefficient vector $(a_1, a_2, \ldots, a_n) \in (\mathbb{F}_q^*)^n$ let us define the following functions:*

$$r(k) = \#\{1 \le j \le k : \eta(a_j) = 1\}, \quad s(k) = \#\{1 \le j \le k : \eta(a_j) = -1\}, \quad 1 \le k \le n.$$

For simplicity, let us denote $r = r(n)$, $s = s(n)$.

Let us partition the set of roots of g_k into classes $A_{i,j}$, fixing the numbers i and j of non-zero coordinates of the roots whose corresponding coefficients are squares and non-squares, respectively. For any root in $A_{i,j}$, let $\{u_1, \ldots, u_{i+j}\}$ be the indices of the $i+j$ non-zero coordinates. Then, the non-zero coordinates, arranged in the same order, form a root in $(\mathbb{F}_q^*)^{i+j}$ of the polynomial

$$g_{u_1,\ldots,u_{i+j}} = a_{u_1}x_{u_1}^2 + \cdots + a_{u_{i+j}}x_{u_{i+j}}^2 - b.$$

Let $g_{u'_1,\ldots,u'_{i+j}}$ be any other polynomial of the same form with the same numbers i and j of square and non-square coefficients. It is easy to construct a bijection between

the roots of $g_{u_1,\dots,u_{i+j}}$ and $g_{u'_1,\dots,u'_{i+j}}$ in $\mathbb{F}_q^{i+j}$, and also in $(\mathbb{F}_q^*)^{i+j}$. Thus, $N(g_{u_1,\dots,u_{i+j}})$ and $N^*(g_{u_1,\dots,u_{i+j}})$ depend only on i and j. Since the number of roots is the only information that matters to us, we will denote any such polynomial simply by $g_{i,j}$ and the quantities as $N(g_{i,j})$ and $N^*(g_{i,j})$. Thus, the number of roots in each class $A_{i,j}$ is $\binom{r(k)}{i}\binom{s(k)}{j}N^*(g_{i,j})$, and the total number of roots of g_k is

$$N(g_{r(k),s(k)}) = N(g_k) = \sum_{\substack{0\le i\le r(k)\\ 0\le j\le s(k)}} \binom{r(k)}{i}\binom{s(k)}{j} N^*(g_{i,j}). \tag{2.5}$$

We remark that the meaning of $N^*(g_{0,0};0,0)$, which appears in a summand in the above expression, is not obvious. It is 1 when $b = 0$ and 0 when $b = 1$. Similarly, $N^*(g_{r(k),s(k)};r(k),s(k))$ is simply $N^*(g_k)$.

The following Binomial Inversion Lemma will allow us to obtain an expression for $N^*(g_k)$ in terms of $N(g_{i,j})$.

Lemma 2.6. *Let G be an abelian group and $f:\mathbb{Z}_{\ge 0}\to G$ a function. Define F by $F(r) = \sum_{i=0}^{r}\binom{r}{i}f(i)$, then f can be expressed in terms of F as*

$$f(r) = \sum_{i=0}^{r}(-1)^{r+i}\binom{r}{i}F(i).$$

Proof. See Section 5.3 in [14]. ∎

Using Lemma 2.6 twice in (2.5), it follows that

$$N^*(g_{r(k),s(k)}) = N^*(g_k) = \sum_{i=0}^{r(k)}\sum_{j=0}^{s(k)}(-1)^{r(k)+s(k)+i+j}\binom{r(k)}{i}\binom{s(k)}{j}N(g_{i,j}). \tag{2.6}$$

From Theorem 1.12, and the fact that $\eta(-1) = (-1)^{(q-1)/2}$, we obtain that

$$N(g_{i,j}) = \begin{cases} q^{i+j-1} - (-1)^j(-1)^{(q-1)(i+j)/4}q^{(i+j-2)/2}, & \text{if } i+j \text{ even and } b\neq 0,\\ q^{i+j-1} + (-1)^j(-1)^{(q-1)(i+j-1)/4}\eta(b)q^{(i+j-1)/2}, & \text{if } i+j \text{ odd and } b\neq 0,\\ q^{i+j-1} + (-1)^j(-1)^{(q-1)(i+j)/4}(q^{(i+j)/2} - q^{(i+j-2)/2}), & \text{if } i+j \text{ even and } b=0,\\ q^{i+j-1}, & \text{if } i+j \text{ odd and } b=0. \end{cases} \tag{2.7}$$

By expressing $N^*(g_k)$ in terms of $N(g_{i,j})$, we can use (2.7) to obtain an explicit value for $N^*(g_k)$, which can then be used in (2.4) to determine $N(f)$.

Theorem 2.7. *Let $\mathbb{F}_q$ be a finite field where q is an odd prime power, g a quadratic diagonal polynomial as in (1.12), and $r(k), s(k)$ be as in Definition 2.5. We fix a complex number θ such that $\theta^2 = q(-1)^{(q-1)/2}$, and we define the complex constants*

$$\zeta_1 = \theta - 1, \quad \zeta_2 = -\theta - 1.$$

We have that

a) if $b \neq 0$, then

$$N^*(g_k) = \frac{(q-1)^k}{q} - \frac{1}{2q}(\zeta_1^{r(k)}\zeta_2^{s(k)} + \zeta_2^{r(k)}\zeta_1^{s(k)}) + \frac{\eta(b)}{2\theta}(\zeta_1^{r(k)}\zeta_2^{s(k)} - \zeta_2^{r(k)}\zeta_1^{s(k)}). \tag{2.8}$$

b) if $b = 0$, then

$$N^*(g_k) = \frac{(q-1)^k}{q} - \frac{q-1}{2q}(\zeta_1^{r(k)}\zeta_2^{s(k)} + \zeta_2^{r(k)}\zeta_1^{s(k)}). \tag{2.9}$$

Proof. Firstly, we remark that both constants ζ_1 and ζ_2 are related to the values of geometric sums. For any positive integer u, we have

$$\zeta_1^u = (-1)^u \sum_{i=0}^{u} \binom{u}{i} (-\theta)^i, \quad \zeta_2^u = (-1)^u \sum_{i=0}^{u} \binom{u}{i} \theta^i.$$

We will now establish the proof for the case where $b \neq 0$ and the analogous case can be reasoned in a similar fashion. Since $b \neq 0$, (2.7) reduces to

$$N(g_{i,j}) = \begin{cases} q^{i+j-1} - (-1)^j (-1)^{(q-1)(i+j)/4} q^{(i+j-2)/2}, & \text{if } i+j \text{ even}, \\ q^{i+j-1} + (-1)^j (-1)^{(q-1)(i+j-1)/4} \eta(b) q^{(i+j-1)/2}, & \text{if } i+j \text{ odd}, \end{cases}$$

which can be rewritten as

$$N(g_{i,j}) = q^{i+j-1} + \frac{(1+(-1)^{i+j})}{2}\left(-(-1)^j(-1)^{(q-1)(i+j)/4} q^{(i+j-2)/2}\right) + \frac{(1-(-1)^{i+j})}{2}\left((-1)^j(-1)^{(q-1)(i+j-1)/4}\eta(b) q^{(i+j-1)/2}\right).$$

This can be used in (2.6) to obtain an equation with a right-hand side that can be partitioned into geometric sums, yielding our result through a straightforward computation.

■

2.3 Main results

In this section we will present the main results published by us in the paper [1]. We are going to determine the number of roots of fully triangular polynomials in some cases, starting with the simplest case, when it is totally $*$-equivalent to a linear polynomial. The following theorem has also been obtained by Cao and Sun as a special case of Corollary 3.2 in [16].

Theorem 2.8. *Let f be a fully triangular polynomial as in (2.1) that is totally $*$-equivalent to a linear polynomial g of the form (1.9). Then,*

$$N(f) = \begin{cases} \frac{q^n-(-1)^n}{q+1}, & \text{if } b \neq 0, \\ \frac{2q^n+(-1)^n(q-1)}{q+1}, & \text{if } b = 0. \end{cases}$$

Proof. We will only show the case where $b \neq 0$, because the other is analogous. In this case, (1.11) from Lemma 1.11 tells us $N^*(g_k) = \frac{(q-1)^k}{q} - \frac{(-1)^k}{q}$, and (2.4) implies $N(f) = N^*(g) + \sum_{k=1}^{n-1} N^*(g_k) q^{n-k-1}$. Therefore,

$$\begin{aligned} N(f) &= \frac{(q-1)^n}{q} - \frac{(-1)^n}{q} + \sum_{k=1}^{n-1} q^{n-k-1} \left[\frac{(q-1)^k}{q} - \frac{(-1)^k}{q} \right] \\ &= \frac{(q-1)^n}{q} - \frac{(-1)^n}{q} + q^{n-2} \left[\sum_{k=1}^{n-1} \left(\frac{q-1}{q} \right)^k - \left(\frac{-1}{q} \right)^k \right] \\ &= \frac{q^n-(-1)^n}{q+1}. \end{aligned}$$

■

We remark that notably, the number of roots in $\mathbb{F}_q^n$ of the fully triangular polynomial f is *not* equal to the number of roots of the linear polynomial g, which is equal to q^{n-1}. We also remark that diagonal polynomials are triangular but not fully triangular, thus any result that requires f to be fully triangular cannot be used on diagonal polynomials.

Notice that the coefficient vector of the fully triangular polynomial does not affect the number of roots of fully triangular polynomials $*$-equivalent to linear polynomials, yielding the following result:

Corollary 2.9. *Let f and h be two fully triangular polynomials with n variables of the form (2.1), such that the constant terms are zero in both of them or non-zero in both of them. If f and h are totally $*$-equivalent to linear polynomials they have the same number of roots.*

Proof. Even when f and h have different coefficients, Theorem 2.8 implies that the number of roots depends only on the number of variables and if the constant term is zero or not. ■

In the following result, we consider the cases when the fully triangular polynomial is totally $*$-equivalent to a quadratic diagonal polynomial.

Theorem 2.10. *Let $\mathbb{F}_q$ be a finite field where q is an odd prime power and let f be a fully triangular polynomial of the form (2.1). Suppose that f is totally $*$-equivalent to a quadratic diagonal polynomial g of the form (1.12). Let θ, ζ_1, ζ_2 be the constants defined in Theorem 2.7. We have that*

a) if $b \neq 0$, then

$$N(f) = q^{n-2}(q-1) - \frac{1}{2q}\cdot(\zeta_1^r\zeta_2^s + \zeta_2^r\zeta_1^s) + \frac{\eta(b)}{2\theta}\cdot(\zeta_1^r\zeta_2^s - \zeta_2^r\zeta_1^s) + \sum_{k=1}^{n-1} q^{n-k-1}\left(-\frac{1}{2q}\cdot(\zeta_1^{r(k)}\zeta_2^{s(k)} + \zeta_2^{r(k)}\zeta_1^{s(k)}) + \frac{\eta(b)}{2\theta}\cdot(\zeta_1^{r(k)}\zeta_2^{s(k)} - \zeta_2^{r(k)}\zeta_1^{s(k)})\right);$$

b) if $b = 0$, then

$$N(f) = q^{n-1} + q^{n-2}(q-1) + \left(\frac{q-1}{2q}\right)(\zeta_1^r\zeta_2^s + \zeta_2^r\zeta_1^s) + \left(\frac{q-1}{2}\right)\sum_{k=1}^{n-1} q^{n-k-2}(\zeta_1^{r(k)}\zeta_2^{s(k)} + \zeta_2^{r(k)}\zeta_1^{s(k)}).$$

Proof. We will only show the proof in the case where $b \neq 0$, because the other case is analogous. In this case, (2.8) in Theorem 2.7 implies

$$N^*(g_k) = \frac{(q-1)^k}{q} - \frac{1}{2q}(\zeta_1^{r(k)}\zeta_2^{s(k)} + \zeta_2^{r(k)}\zeta_1^{s(k)}) + \frac{\eta(b)}{2\theta}(\zeta_1^{r(k)}\zeta_2^{s(k)} - \zeta_2^{r(k)}\zeta_1^{s(k)}),$$

and from (2.4) we have $N(f) = N^*(g) + \sum_{k=1}^{n-1} N^*(g_k)q^{n-k-1}$. Therefore,

$$N(f) = \frac{(q-1)^n}{q} - \frac{1}{2q}\cdot(\zeta_1^r\zeta_2^s + \zeta_2^r\zeta_1^s) + \frac{\eta(b)}{2\theta}\cdot(\zeta_1^r\zeta_2^s - \zeta_2^r\zeta_1^s) + \sum_{k=1}^{n-1} q^{n-k-1}\left(\frac{(q-1)^k}{q} - \frac{1}{2q}\cdot(\zeta_1^{r(k)}\zeta_2^{s(k)} + \zeta_2^{r(k)}\zeta_1^{s(k)}) + \frac{\eta(b)}{2\theta}\cdot(\zeta_1^{r(k)}\zeta_2^{s(k)} - \zeta_2^{r(k)}\zeta_1^{s(k)})\right),$$

which simplifies to obtain the desired result. ■

Notice that the specific values in the coefficient vector $(a_1, a_2, \ldots, a_n, -b)$ do not matter. In fact, to determine the number of roots, it is sufficient to determine which of the coefficients a_j's and b are squares or not. Thus, we have the following result:

Corollary 2.11. *Let $\mathbb{F}_q$ be a finite field where q is an odd prime power. Let f and h be two fully triangular polynomials with n variables of the form (2.1) with coefficient vectors $(a_1, a_2, \ldots, a_n, -b)$ and $(c_1, c_2, \ldots, c_n, -d)$ respectively. Let us suppose that $\eta(a_1) = \eta(c_1), \ldots, \eta(a_n) = \eta(c_n), \eta(b) = \eta(d)$. If f and h are totally $*$-equivalent to quadratic diagonal polynomials, they have the same number of roots.*

Proof. The values of $r(k)$ and $s(k)$ for $1 \leq k \leq n$ will be the same for both polynomials. Hence from Theorem 2.10 they both have the same number of roots in $\mathbb{F}_q^n$. ■

Then, for every choice of coefficients in the coefficient vector we have fixed values for $r(k), s(k)$ for $1 \leq k \leq n$. We can substitute these values into Theorem 1.12 to find a closed expression for the number of roots. We will do this for two specific cases.

Corollary 2.12. *Let $\mathbb{F}_q$ be a finite field where q is an odd prime power, and let f be a fully triangular polynomial of the form (2.1) such that the coefficients $a_1, \ldots, a_n$ are either all squares, or are all non squares in $\mathbb{F}_q$. Let us suppose that f is totally $*$-equivalent to a quadratic diagonal polynomial g of the form (1.12). We have that*

a) if $b \neq 0$, then

$$N(f) = q^{n-2}(q-1) - \frac{1}{2q} \cdot \left(\frac{\zeta_1^{n+1} - (q-1)\zeta_1^n - \zeta_1 q^{n-1}}{\zeta_1 - q} + \frac{\zeta_2^{n+1} - (q-1)\zeta_2^n - \zeta_2 q^{n-1}}{\zeta_2 - q} \right) + \frac{\varepsilon \eta(b)}{2\theta} \cdot \left(\frac{\zeta_1^{n+1} - (q-1)\zeta_1^n - \zeta_1 q^{n-1}}{\zeta_1 - q} - \frac{\zeta_2^{n+1} - (q-1)\zeta_2^n - \zeta_2 q^{n-1}}{\zeta_2 - q} \right);$$

where

$$\varepsilon = \begin{cases} 1, & \text{if } a_1, \ldots, a_n \text{ are squares,} \\ -1, & \text{if } a_1, \ldots, a_n \text{ are non squares.} \end{cases}$$

b) if $b = 0$, then

$$N(f) = 2q^{n-1} - q^{n-2} + \frac{q-1}{2q} \left(\frac{\zeta_1^{n+1} - (q-1)\zeta_1^n - \zeta_1 q^{n-1}}{\zeta_1 - q} + \frac{\zeta_2^{n+1} - (q-1)\zeta_2^n - \zeta_2 q^{n-1}}{\zeta_2 - q} \right),$$

in both cases.

Proof. In the case when $a_1, \ldots, a_n$ are squares, we have $r(k) = k, s(k) = 0$ for $1 \leq k \leq n$. Substituting into the expressions in Theorem 2.10 and computing the geometric sums yields the result.

For the case when none of the coefficients $a_1,\dots,a_n$ is a square, we can multiply f by a nonsquare coefficient a to obtain a polynomial with exactly the same roots, but whose coefficients $aa_1,\dots,aa_n$ are squares, and the constant term ab is such that $\eta(ab) = -\eta(b)$. Thus, in the case when $b = 0$, the number of roots is exactly the same as the all squares case, and in the case $b \neq 0$, the expression is essentially the same with a couple of signs changed. ■

Let us conclude this chapter with some examples. Consider the finite field $\mathbb{F}_9$ and the fully triangular polynomial

$$f(x) = x^2 + x^2y^2 + x^2y^6z^2$$

over $\mathbb{F}_9[x,y,z]$. This polynomial is totally $*$-equivalents to the polynomial

$$g(x) = x^2 + y^2 + z^2,$$

where all coefficients are squares, since all coefficients are the same and

$$\begin{bmatrix} 1 & 1 & 1 \\ 2 & 2 & 2 \\ 0 & 2 & 6 \\ 0 & 0 & 2 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 \\ 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 2 \end{bmatrix}.$$

Under these parameters, we can compute the values of ζ_1 and ζ_2:

$$\zeta_1 = 2, \quad \zeta_2 = -4.$$

Then we can use the second formula in Corollary 2.12 to compute $N(f)$:

$$\begin{aligned} N(f) &= 2\cdot 9^2 - 9^1 + \frac{9-1}{2\cdot 9}\left(\frac{2^4 - (9-1)2^3 - 2\cdot 9^2}{2-9} + \frac{(-4)^4 - (9-1)(-4)^3 - (-4)9^2}{-4-9}\right) \\ &= 129. \end{aligned}$$

According to the same corollary, a polynomial of the same form where the coefficients are nonsquares will also have 129 roots. An example of that is

$$f(x) = 2x^2 + 2x^2y^2 + 2x^2y^6z^2.$$

Let us consider the polynomial

$$f(x,y) = x^2 + 2x^2y^2$$

in $\mathbb{F}_5[x,y]$. It is totally $*$-equivalent to the quadratic diagonal polynomial

$$g(x,y) = x^2 + 2y^2$$

since the coefficient vector $(1,2)$ is the same and

$$\begin{bmatrix} 1 & 1 \\ 2 & 2 \\ 0 & 2 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 2 & 0 \\ 0 & 2 \end{bmatrix}.$$

Under these parameters, we can compute the values of ζ_1, ζ_2 and $r(k), s(k)$ for $k \in \{0,1\}$:

$$\zeta_1 = \sqrt{5} - 1, \quad \zeta_2 = -\sqrt{5} - 1, \quad r(1) = r(2) = s(2) = 1, \quad s(1) = 0.$$

Then we can use the second formula in Theorem 2.10 to compute $N(f)$:

$$\begin{aligned} N(f) &= 5^1 + 5^0(5-1) + \left(\frac{5-1}{2\cdot 5}\right)((\sqrt{5}-1)^1(-\sqrt{5}-1)^1 + (-\sqrt{5}-1)^1(\sqrt{5}-1)^1) \\ &\quad + \left(\frac{5-1}{2}\right)5^{-1}((\sqrt{5}-1)^1(-\sqrt{5}-1)^0 + (-\sqrt{5}-1)^1(\sqrt{5}-1)^0) \\ &= 5. \end{aligned}$$

CHAPTER 3

FULL POLYNOMIALS

In the previous chapter, we discussed the number of solutions of fully triangular polynomials related to diagonal polynomials.

In this chapter, we will present the content of a paper that we have submitted and is yet to be published [2]. In it we discuss the number of solutions of polynomials where the augmented degree matrix is full, i.e., every term in the polynomial, with the possible exception of a constant term, has every value with positive exponent.

To do so, we will assume $*$-equivalency to diagonal polynomials where the exponents guarantee pure Gauss sums. We will start by determining the number of roots over $\mathbb{F}_q^*$ of those polynomials.

3.1 Diagonal polynomials and pure sums

Lemma 3.1. *Let g be a diagonal polynomial of the form*

$$g(X) = a_1 x_1^{d_1} + \cdots + a_s x_1^{d_s} - b,$$

where $a_1, \ldots, a_s \in \mathbb{F}_q^*$ and $b \in \mathbb{F}_q$. Then

$$N^*(g) = \frac{(q-1)^s}{q} + \frac{1}{q} \sum_{c \in \mathbb{F}_q^*} \bar{\psi}(cb) \prod_{i=1}^{s} S(ca_i, d_i), \tag{3.1}$$

where $S(u,d) := \sum_{x \in \mathbb{F}_q^*} \psi\left(ux^d\right)$.

Proof. The expression in (1.6) simplifies to

$$\begin{aligned} N^*(g) &= \frac{1}{q}\sum_{c\in\mathbb{F}_q}\sum_{x_1,\dots,x_s\in\mathbb{F}_q^*}\psi\left(c\left(-b+\sum_{i=1}^{s}a_i x_i^{d_i}\right)\right) \\ &= \frac{(q-1)^s}{q}+\frac{1}{q}\sum_{c\in\mathbb{F}_q^*}\bar{\psi}(cb)\prod_{i=1}^{n}\sum_{x\in\mathbb{F}_q^*}\psi\left(ca_i x^{d_i}\right). \end{aligned}$$

■

Lemma 3.1 implies that being able to compute $S(ca_j,d_j)$ for $1\le j\le s$ would assist us in counting the number of $*$-roots of a diagonal polynomial. The following result addresses that when the exponents are (p,r_j)-admissible for suitable positive integers r_j.

Lemma 3.2. *Let u be an element in $\mathbb{F}_q^*$, $d\ge 3$ a (p,r)-admissible positive integer and $h=m/(2r)$. If p is odd, we have*

$$S(u,d)=\begin{cases}-1-(-1)^h(d-1)q^{1/2}, & \text{if } \eta_d(u)=(-1)^{h(p^r+1)/d},\\ -1+(-1)^h q^{1/2}, & \text{otherwise.}\end{cases}$$

And if $p=2$,

$$S(u,d)=\begin{cases}-1-(-1)^h(d-1)q^{1/2}, & \text{if } \eta_d(u)=1,\\ -1+(-1)^h q^{1/2}, & \text{otherwise.}\end{cases}$$

Proof. We use the Lemma 1.2 to rewrite the sum:

$$\begin{aligned} S(u,d) &= \sum_{x\in\mathbb{F}_q^*}\psi\left(ux^d\right)\\ &= \sum_{y\in\mathbb{F}_q^*}\psi(uy)\sum_{j=0}^{d-1}\eta_d^j(y)\\ &= \sum_{y\in\mathbb{F}_q^*}\psi(uy)\sum_{j=0}^{d-1}\eta_d^j(y)\cdot\eta_d^j(u)\bar{\eta_d}^j(u)\\ &= \sum_{j=0}^{d-1}\bar{\eta_d}^j(u)\sum_{y\in\mathbb{F}_q^*}\psi(uy)\eta_d^j(uy)\\ &= \sum_{j=0}^{d-1}\bar{\eta_d}^j(u)\sum_{y\in\mathbb{F}_q^*}\psi(y)\eta_d^j(y)\\ &= -1+\sum_{j=1}^{d-1}\bar{\eta_d}^j(u)G(\eta_d^j). \end{aligned}$$

Our assumption that d is (p,r)-admissible guarantees that the Gauss sums are pure. Let us define

$$\Delta(d,j) = \begin{cases} (-1)^{jh}, & \text{if } 2 \mid d \text{ and } 2 \nmid (p^r+1)/d, \\ 1, & \text{if } 2 \nmid d \text{ or } 2 \mid (p^r+1)/d. \end{cases} \tag{3.2}$$

Using Lemma 1.8 and Equation (3.2) we obtain

$$\begin{aligned} S(u,d) &= -1 + \sum_{j=0}^{d-1} \bar{\eta_d}^j(u)(-1)^{h+1} q^{1/2} \Delta(d,j) \\ &= -1 + (-1)^{h+1} q^{1/2} T(u,d), \end{aligned} \tag{3.3}$$

where $T(u,d) = \sum_{j=1}^{d-1} \bar{\eta_d}^j(u)\Delta(d,j)$.

If $2 \mid d$ and $2 \nmid (p^r+1)/d$, then $\Delta(d,j) = (-1)^{jh}$. Thus,

$$T(u,d) = \sum_{j=1}^{d-1} ((-1)^h \bar{\eta_d}(u))^j = \begin{cases} d-1 & \text{if } \eta_d(u) = (-1)^h, \\ -1 & \text{if } \eta_d(u) \neq (-1)^h. \end{cases} \tag{3.4}$$

Similarly, if $2 \nmid d$ or $2 \mid (p^r+1)/d$ then $\Delta(d,j) = 1$, in which case

$$T(u,d) = \sum_{j=1}^{d-1} (\bar{\eta_d}(u))^j = \begin{cases} d-1 & \text{if } \eta_d(u) = 1, \\ -1 & \text{if } \eta_d(u) \neq 1. \end{cases} \tag{3.5}$$

We claim that combining the cases (3.4) and (3.5) we get

$$T(u,d) = \begin{cases} d-1 & \text{if } \eta_d(u) = (-1)^{h(p^r+1)/d}, \\ -1 & \text{if } \eta_d(u) \neq (-1)^{h(p^r+1)/d}, \end{cases} \tag{3.6}$$

when p is odd. To prove this claim we just need to show that

$$(-1)^{h(p^r+1)/d} = \begin{cases} (-1)^h & \text{if } 2 \mid d \text{ and } 2 \nmid (p^r+1)/d, \\ 1 & \text{if } 2 \nmid d \text{ or } 2 \mid (p^r+1)/d. \end{cases}$$

For the first case, we notice that the condition $2 \nmid (p^r+1)/d$ implies $(-1)^{h(p^r+1)/d} = (-1)^h$. Analogously if $2 \mid (p^r+1)$ then $(-1)^{h(p^r+1)/d} = 1$. The condition $2 \nmid d$ on the second case also implies $(-1)^{h(p^r+1)/d} = 1$ since p being odd implies $2 \mid p^r+1$ in that case. This proves claim (3.6), which when applied to (3.3) gives us the desired formula for $S(u,d)$.

If $p = 2$, since $d \mid p^r+1$ we must have $2 \nmid d$. Thus $T(u,d)$ is as given in (3.5), which when applied to (3.3) gives us the desired result.

■

For convenience, let us introduce the following definitions.

Definition 3.3. *Let $d \geq 3$ be a (p,r)-admissible integer and $h = m/(2r)$. We define the functions A and B as*

$$A(d) = -1 - (-1)^h (d-1) q^{1/2},$$
$$B(d) = -1 + (-1)^h q^{1/2}.$$

We remark that since q is fixed and r and h are defined in terms of d, only d is necessary to compute these values.

Applying Lemma 3.2 to Equation (3.1) will help us to compute the number of $*$-roots for diagonal polynomials satisfying that special condition. From now on, let us consider a diagonal polynomial $g \in \mathbb{F}_q[x_1, \ldots, x_s]$ of the form

$$g(X) = a_1 x_1^d + \cdots + a_s x_s^d - b, \tag{3.7}$$

where $\eta_d(a_1) = \cdots = \eta_d(a_s)$ for some integer $d \geq 3$ that is a (p,r)-admissible integer for a suitable positive integer r.

Theorem 3.4. *Let $g \in \mathbb{F}_q[x_1, \ldots, x_s]$ be a polynomial of the form (3.7) where $b = 0$. Then*

$$N^*(g) = \frac{1}{q}\left((q-1)^s + \frac{q-1}{d} A(d)^s + \frac{(q-1)(d-1)}{d} B(d)^s\right).$$

Proof. From (3.1), we have

$$N^*(g) = \frac{(q-1)^s}{q} + \frac{1}{q} \sum_{c \in \mathbb{F}_q^*} \prod_{i=1}^{n} S(ca_i, d).$$

Let us suppose p is odd and $(-1)^{h(p^r+1)/d} = 1$ so that the condition we need to check to determine the value of $S(ca_i, d)$ is whether $\eta_d(ca_i)$ equals 1. Since $\eta_d(a_1) = \cdots = \eta_d(a_s)$, for each $c \in \mathbb{F}_q^*$ the value of $\eta_d(ca_i)$ is the same for every $1 \leq i \leq s$, which gives us

$$\sum_{c \in \mathbb{F}_q^*} \prod_{i=1}^{s} S(ca_i, d) = \sum_{c \in \mathbb{F}_q^*} \left(S(ca_1, d)\right)^s.$$

As we have seen in Lemma 3.2 the value of $S(ca_1, d)$ will be $A(d)$ if ca_1 is a d-th power, and $B(d)$ if it is not. Hence,

$$\sum_{c \in \mathbb{F}_q^*} \left(S(ca_1, d)\right)^s = \frac{q-1}{d} A(d)^s + \frac{(q-1)(d-1)}{d} B(d)^s,$$

which yields the result when applied in (3.1). Other cases are analogous.

■

For instance, the diagonal polynomial $g(x,y) = x^4 + y^4$ over $\mathbb{F}_{81}$ is such that the exponent $d = 4$ is $(3,2)$-admissible, so we can compute $A(4) = -28$, $B(4) = 8$ and use Theorem 3.4 to conclude the number of $*$-roots is

$$\begin{aligned} N^*(g) &= \frac{1}{81}\left(80^2 + 20(-28)^2 + 60(8)^2\right) \\ &= 320. \end{aligned}$$

Theorem 3.5. *Let $g \in \mathbb{F}_q[x_1,\ldots,x_s]$ be a polynomial of the form (3.7) where $b \neq 0$. Then*

$$N^*(g) = \begin{cases} \frac{1}{q}\left((q-1)^s - B(d)^s + \frac{A(d)(A(d)^s - B(d)^s)}{d}\right) & \text{if } \eta_d(a_1/b) = 1, \\ \frac{1}{q}\left((q-1)^s - B(d)^s + \frac{B(d)(A(d)^s - B(d)^s)}{d}\right) & \text{if } \eta_d(a_1/b) \neq 1. \end{cases}$$

Proof. In the equation (3.1), we have

$$N^*(g) = \frac{(q-1)^s}{q} + \frac{1}{q}\sum_{c\in\mathbb{F}_q^*} \bar{\psi}(cb)\prod_{i=1}^{s} S(ca_i,d).$$

Since $\eta_d(a_1) = \cdots = \eta_d(a_s)$ implies $S(ca_1,d) = \cdots = S(ca_s,d)$, we can write

$$\begin{aligned} N^*(g) &= \frac{(q-1)^s}{q} + \frac{1}{q}\sum_{c\in\mathbb{F}_q^*} \bar{\psi}(cb)S(ca_1,d)^s \\ &= \frac{(q-1)^s}{q} + \frac{1}{q}\sum_{c\in\mathbb{F}_q^*} \bar{\psi}(cba_1^{-1})S(c,d)^s. \end{aligned} \tag{3.8}$$

Let us assume that p is odd and $(-1)^{h(p^r+1)/d} = 1$. In this case, $S(c,d) = A(d)$ if c is a d-th power and $B(d)$ otherwise. Since $\mathbb{F}_q^* = \{\gamma^k : 1 \leq k \leq q-1\}$, we can separate the sum $\sum_{c\in\mathbb{F}_q^*} \bar{\psi}(cba_1^{-1})S(c,d)^s$ according to whether c is a d-th power of some element of $\mathbb{F}_q^*$:

$$\begin{aligned} \sum_{c\in\mathbb{F}_q^*} \bar{\psi}(cba_1^{-1})S(c,d)^s &= \sum_{l=1}^{(q-1)/d} \bar{\psi}(\gamma^{ld}ba_1^{-1})A(d)^s + \left(\sum_{c\in\mathbb{F}_q^*} \bar{\psi}(cba_1^{-1}) - \sum_{l=1}^{(q-1)/d} \bar{\psi}(\gamma^{ld}ba_1^{-1})\right)B(d)^s \\ &= -B(d)^s + \left(A(d)^s - B(d)^s\right)\cdot \sum_{l=1}^{(q-1)/d} \bar{\psi}(\gamma^{ld}ba_1^{-1}) \\ &= -B(d)^s + \frac{(A(d)^s - B(d)^s)}{d}\cdot \sum_{x\in\mathbb{F}_q^*} \bar{\psi}(ba_1^{-1}x^d) \\ &= -B(d)^s + \frac{(A(d)^s - B(d)^s)}{d}\cdot \sum_{x\in\mathbb{F}_q^*} \psi(-ba_1^{-1}x^d), \end{aligned}$$

where the sum $\sum_{x\in\mathbb{F}_q^*}\psi(-ba_1^{-1}x^d)$ is $S(-b/a_1,d)$, which is $A(d)$ if $\eta_d(-b/a_1)=1$ and $B(d)$ otherwise. Since $-1=\gamma^{(q-1)/2}$ we have $\eta_d(-1)=1$ if $d\mid(q-1)/2$. Since $2r\mid m$, $(q-1)/(p^r+1)=\sum_{j=0}^{m/r-1}p^{rj}$ is even. Then, since $d\mid p^r+1$ we have that $d\mid(q-1)/2$; hence the condition $\eta_d(-b/a_1)=1$ is equivalent to $\eta_d(a_1/b)=1$. Therefore,

$$N^*(g)=\begin{cases}\frac{1}{q}\left((q-1)^s-B(d)^s+\frac{A(d)(A(d)^s-B(d)^s)}{d}\right) & \text{if } \eta_d(a_1/b)=1,\\ \frac{1}{q}\left((q-1)^s-B(d)^s+\frac{B(d)(A(d)^s-B(d)^s)}{d}\right) & \text{if } \eta_d(a_1/b)\neq 1.\end{cases} \tag{3.9}$$

The case where $p=2$ is analogous to this case. Lastly, let us assume p is odd and $(-1)^{h(p^r+1)/d}=-1$. The argument is analogous until (3.8). Then $S(c,d)=A(d)$ if $\eta_d(c,d)=-1$ and $S(c,d)=B(d)$ otherwise, i.e., $S(\gamma^k,d)=A(d)$ if and only if $k=ld+d/2$, for $1\le l\le(q-1)/d$. Thus, the sum $\sum_{c\in\mathbb{F}_q^*}\bar{\psi}(cba_1^{-1})S(c,d)^s$ can be written as

$$\begin{aligned}\sum_{c\in\mathbb{F}_q^*}\bar{\psi}(cba_1^{-1})S(c,d)^s &= \sum_{l=1}^{(q-1)/d}\bar{\psi}(\gamma^{ld+d/2}ba_1^{-1})A(d)^s+\left(\sum_{c\in\mathbb{F}_q^*}\bar{\psi}(cba_1^{-1})-\sum_{l=1}^{(q-1)/d}\bar{\psi}(\gamma^{ld+d/2}ba_1^{-1})\right)B(d)^s\\ &= -B(d)^s+\left(A(d)^s-B(d)^s\right)\cdot\sum_{l=1}^{(q-1)/d}\bar{\psi}(\gamma^{ld+d/2}ba_1^{-1})\\ &= -B(d)^s+\frac{\left(A(d)^s-B(d)^s\right)}{d}\cdot\sum_{x\in\mathbb{F}_q^*}\bar{\psi}(ba_1^{-1}\gamma^{d/2}x^d)\\ &= -B(d)^s+\frac{\left(A(d)^s-B(d)^s\right)}{d}\cdot\sum_{x\in\mathbb{F}_q^*}\psi(-ba_1^{-1}\gamma^{d/2}x^d),\end{aligned}$$

where the internal sum $\sum_{x\in\mathbb{F}_q^*}\psi(-ba_1^{-1}\gamma^{d/2}x^d)$ is $S(-ba_1^{-1}\gamma^{d/2},d)$, which is $A(d)$ if

$$\eta_d(-ba_1^{-1}\gamma^{d/2})=-\eta_d(-b/a_1)=-1$$

and $B(d)$ otherwise. Hence, the number of solutions is again as seen in (3.9).

■

For instance, let us consider the polynomial $g(x,y,z)=x^4+y^4+z^4-1\in\mathbb{F}_{81}[x,y,z]$. It satisfies the condition $\eta_4(a_1/b)=\eta_4(1)=1$, thus we use the first formula. Since $A(4)=-28$ and $B(4)=8$, we have

$$\begin{aligned}N^*(g) &= \frac{1}{81}\left(80^3-8^3-7((-28)^3-8^3)\right)\\ &= 8256.\end{aligned}$$

As another example, let us consider the diagonal polynomial $g(x,y)=\gamma x^{17}+\gamma^{18}y^{17}-1\in\mathbb{F}_{256}[x,y]$, where γ is a primitive element of $\mathbb{F}_{256}$ that is not a 17-th power of any other element. The exponent $d=17$ is $(2,4)$-admissible, and since $\eta_{17}(a_1/b)=\eta_{17}(\gamma)\neq 1$, we use the second formula. Since $A(17)=255$ and $B(17)=-17$ over this field we get

$$\begin{aligned}N^*(g) &= \frac{1}{256}\left(255^2-(-17)^2+\frac{(-17)}{17}((255)^2-(-17)^2)\right)\\ &= 0.\end{aligned}$$

3.2 Main results

Let us define full polynomials as polynomials of the form

$$f(X) = a_1 x_1^{d_{1,1}} x_2^{d_{2,1}} \cdots x_n^{d_{n,1}} + \cdots + a_n x_1^{d_{1,n}} x_2^{d_{2,1}} \cdots x_n^{d_{n,n}} - b, \tag{3.10}$$

where every exponent $d_{i,j}$ in the degree matrix is greater than zero.

The number of solutions with at least one variable equal to zero is easy to compute: since every term has every variable, if at least one of them is zero then every non-constant term vanishes. Thus when requiring at least one of the coordinates to be 0, the number of roots is 0 if $b \neq 0$ and $q^n - (q-1)^n$ if $b = 0$. Hence if we suppose that a polynomial of the form (3.10) is $*$-equivalent to a polynomial $g(X)$ for which the number of $*$-roots is known, then we can calculate $N(f)$ as

$$N(f) = \begin{cases} N^*(g), & \text{if } b \neq 0; \\ q^n - (q-1)^n + N^*(g), & \text{if } b = 0. \end{cases} \tag{3.11}$$

Theorem 3.6. *Let f be a full polynomial of the form (3.10) in $\mathbb{F}_q[x_1,\ldots,x_n]$ that is $*$-equivalent to a diagonal polynomial g of the form (3.7) with s variables, where $n \geq s$, $d \geq 3$ is (p,r)-admissible for some positive integer r such that $2r \mid m$, and $\eta_d(a_1) = \cdots = \eta_d(a_s)$. Then*

$$N(f) = \begin{cases} q^n - (q-1)^n + \frac{(q-1)^{n-s}}{q}\left((q-1)^s + \frac{q-1}{d}A(d)^s + \frac{(q-1)(d-1)}{d}B(d)^s\right) & \text{if } b = 0, \\ \frac{(q-1)^{n-s}}{q}\left((q-1)^s - B(d)^s + \frac{A(d)(A(d)^s - B(d)^s)}{d}\right) & \text{if } b \neq 0 \text{ and } \eta_d(a_1/b) = 1, \\ \frac{(q-1)^{n-s}}{q}\left((q-1)^s - B(d)^s + \frac{B(d)(A(d)^s - B(d)^s)}{d}\right) & \text{if } b \neq 0 \text{ and } \eta_d(a_1/b) \neq 1, \end{cases}$$

where $A(d)$ and $B(d)$ are as stated in Definition 3.3.

Proof. Here $g(x_1,\ldots,x_s)$ is seen as a polynomial in $\mathbb{F}_q[x_1,\ldots,x_n]$, so there are $n-s$ free variables that can assume any value without changing the value of the polynomial. Thus $N_s^*(g)$ is $(q-1)^{n-s}$ times the values obtained from Theorems 3.4 and 3.5. Substituting these into (3.11) yields the result. ■

Let us conclude with some examples. Let us fix $p = 2$, $m = 4$ and $d = 5$ such that $q = 16$, d is $(2,2)$-admissible and count the total number of roots of the polynomial $f(x,y,z) = x^6y^2z + xy^7z^{11} \in \mathbb{F}_q[x,y,z]$, which is $*$-equivalent to $g(x,y,z) = x^5 + y^5$, since we have the following row-equivalence between the matrices $\tilde{D}_f$ and $\tilde{D}_g$:

$$\begin{bmatrix} 1 & 1 \\ 6 & 1 \\ 2 & 7 \\ 1 & 11 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 2 & 0 & 1 & 0 \\ 1 & 0 & 2 & 1 \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 5 & 0 \\ 0 & 5 \\ 0 & 0 \end{bmatrix}.$$

The total number of variables is $n = 3$ and the diagonal polynomial g can be considered as a polynomial on only $s = 2$ variables. We compute $A(5) = 15$ and $B(5) = -5$. Since the constant term is 0, we can apply the first formula in Theorem 3.6 to obtain

$$\begin{aligned} N(f) &= 16^3 - (15)^3 + \frac{15}{16}\left(15^2 + \frac{15}{5}15^2 + \frac{15\cdot 4}{5}(-5)^2\right) \\ &= 1846. \end{aligned}$$

As another example, let us fix $p = 3$, $m = 6$ and $d = 7$, $q = 729$, γ a primitive element of $\mathbb{F}_q$ that is not a 7-th power of any other element. We will count the total number of roots of the polynomial $f(x,y) = x^{14}y^7 + 2x^{21}y^{21} - \gamma \in \mathbb{F}_q[x,y]$, which is $*$-equivalent to $g(x,y) = x^7 + y^7 - \gamma$ since they have the same coefficient vector and $\tilde{D}_f$ and $\tilde{D}_g$ are row-equivalent:

$$\begin{bmatrix} 1 & 1 & 1 \\ 14 & 21 & 0 \\ 7 & 21 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & 3 \\ 0 & 1 & 3 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 \\ 7 & 0 & 0 \\ 0 & 7 & 0 \end{bmatrix}.$$

The number of variables is the same for both polynomials, $n = s = 2$. Since 7 is $(3,3)-$admissible and, we compute $A(7) = 161$ and $B(7) = -28$. Since the $\eta_7(a_1/b) = \eta_7(\gamma) \neq 1$, we use the third formula in Theorem 3.6 to obtain

$$\begin{aligned} N(f) &= \frac{728^0}{729}\left(728^2 - (-28)^2 + \frac{(-28)(161^2 - (-28)^2)}{7}\right) \\ &= 588. \end{aligned}$$

CHAPTER 4

LOW WEIGHT CODEWORDS

In this chapter, we will present the content of a paper that we have submitted and is yet to be published [3].

Generally, when talking about weight distributions of a code, the problem is to determine the entire weight distribution. This is a difficult problem where the methods differ depending on the cyclic code, but typically require the cyclic code to have few possible weights for its codewords. Many examples of this can be found in [26] and [27].

However, some works take a different approach. Instead of trying to determine the entire weight distribution, another idea is to determine the number of codewords with specific small weights. There are some benefits to this: the code having many possible weights typically does not introduce great difficulties, and it can be used to determine the minimum weight of a code without computing the entire weight distribution. Examples of this approach can be seen in [20] and [23].

This chapter will expand on the studies of low-weight codewords in cyclic codes.

4.1 General results

Some of the results we will present don't require the cyclic code to be binary, while others only apply to binary codes. Let us start with the results that also apply to cyclic codes that are non-binary.

The first question we can ask ourselves about codes over $\mathbb{F}_p$ of the form $C_{t_1,\ldots,t_s}$ with

length $n = p^m - 1$ is if there are codewords with weight 2 and to determine the number of codewords with weight 2, i.e., A_2. The following theorem provides a sufficient and necessary condition and determines A_2.

Theorem 4.1. *Let us consider $q = p^m$ to be a prime power. The code $C_{t_1,\dots,t_s}$ of length $n = q - 1$ has a minimum distance of 2 if and only if the greatest common divisor*

$$D(t_1,\dots,t_s) := \gcd(q-1, t_1(p-1), t_2 - t_1, t_3 - t_1, \dots, t_s - t_1)$$

is greater than 1. In that case, the number of codewords with weight 2 is given by

$$A_2 = \frac{(p-1)(q-1)(\sigma_0(D(t_1,\dots,t_s)) - 1)}{2},$$

where $\sigma_0(D(t_1,\dots,t_s))$ is the number of divisors of $D(t_1,\dots,t_s)$.

If $p = 2$ the expression for $D(t_1,\dots,t_s)$ simplifies to $\gcd(q-1, t_1, \dots, t_s)$.

Proof. A codeword of weight 2 is a vector of length $n = q - 1$ over $\mathbb{F}_p$ such that only two coordinates are non-zero. We can use some shifts to make one of the indices the first index and multiply by the inverse of the constant in the coordinate to make it a one. Thus, the code has a codeword with weight 2 if and only if it has a codeword of the form

$$c(i, \alpha) = 1 + \alpha x^i$$

where $1 \le i \le q - 2$ and $\alpha \in \mathbb{F}_p^*$. Now, multiplying the vector form of this codeword by the matrix H given in (1.14), we get the system

$$\begin{cases} 1 + \alpha\gamma^{it_1} = 0, \\ \vdots \\ 1 + \alpha\gamma^{it_s} = 0, \end{cases}$$

which is equivalent to

$$\begin{cases} \gamma^{it_1} = \cdots = \gamma^{it_s}, \\ \gamma^{it_1} \in \mathbb{F}_p^*. \end{cases} \tag{4.1}$$

The first equation in (4.1) is equivalent to

$$i(t_k - t_1) \equiv 0 \pmod{q-1} \tag{4.2}$$

for $k = 2,\dots,s$. For each $2 \le k \le s$, (4.2) has at least one solution if and only if $\gcd(t_k - t_1, q - 1) > 1$, and the indices for which this is true are the multiples of $(q-1)/\gcd(t_k - t_1, q - 1)$.

The second equation in (4.1) is satisfied if and only if $\gcd(t_1(p-1), q-1) > 1$, which is always true, with the possible exception of when $p = 2$. The values of i for which this condition is satisfied are the multiples of $(q-1)/\gcd(t_1(p-1), q-1)$. Thus all the conditions in (4.1) are satisfied if and only if $D(t_1, \ldots, t_s) = \gcd(q-1, t_1(p-1), t_2-t_1, t_3-t_1, \ldots, t_s-t_1) > 1$, and the indices i that satisfy the system are the multiples of $(q-1)/D(t_1, \ldots, t_s)$ smaller than $q-1$, and each one of those will give us a word of the form $c(i, \alpha)$. There are $\sigma_0(D(t-1, \ldots, t_s)) - 1$ such multiples. We can then generate every word with weight 2 in the code doing $n = q-1$ shifts and multiplying by the $p-1$ elements in $\mathbb{F}_p^*$. This will produce each codeword twice, thus $A_2 = \frac{(p-1)(q-1)(\sigma_0(D(t_1,\ldots,t_s))-1)}{2}$.

■

For instance, let $p = 2$, $q = 2^3$ and consider the binary cyclic code $C_{1,5}$ of length $n = 7$ over $\mathbb{F}_q$. Since $D(1,5) = 1$, this code has no codewords with weight 2.

For a case where the minimum distance is 2, let $p = 3$, $q = 3^2$ and consider the cyclic code $C_{1,5}$ of length $n = 8$ over $\mathbb{F}_q$. We have

$$D(1,5) = \gcd(9-1, 1(3-1), 5-1) = 2,$$

thus $A_2 = 8$. The codewords with weight 2 in this code are $(1,0,0,0,1,0,0,0)$, its shifts and their multiples.

4.2 Codewords with weight 3

We will now introduce a sufficient condition for a cyclic code of the form $C_{t_1,\ldots,t_s}$ to have a minimum distance of 3. Note that it is not a necessary condition, so it is not an exhaustive characterization of all cyclic codes with a minimum distance of 3.

Let us denote by $K_g(t)$ the p-cyclotomic coset of $t \pmod{p^g - 1}$, i.e.,

$$K_g(t) = \{tp^k \pmod{p^g - 1} : k = 0, \ldots, g-1\}.$$

We say that an integer $0 \le i \le q-2$ belongs to $K_g(t)$ if there is an integer $0 \le j \le g-1$ such that $ip^j \equiv t \pmod{p^g - 1}$.

Theorem 4.2. *Let $t_1, \ldots, t_s$ be integers that are not in the same p-cyclotomic coset $p \pmod{p^m - 1}$, but are in the same p-cyclotomic coset $K_g(t)$ where $t \not\equiv 0 \pmod{p^g - 1}$ and g is a fixed divisor of m. Then the cyclic code $C_{t_1,\ldots,t_s}$ has a minimum distance $d \le 3$.*

Moreover, the number A_3 of codewords with weight 3 for $C_{t_1,\ldots,t_s}$ satisfies the following bound:

$$A_3 \geq (p-1)(q-1)(p^g-2)/6.$$

Proof. A proof for the case when the code is binary can be seen in [20, Theorem 1, Theorem 3]. We will prove that the code has a minimum distance $d \leq 3$ by constructing codewords of weight 3 and demonstrating they are in the code.

Let $\beta = \gamma^u$, where $u = (q-1)/(p^g-1)$. It is a primitive element of $\mathbb{F}_{p^g}$, and since β generates $\mathbb{F}_{p^g}^*$, for each $1 \leq a \leq p^g-2$ there is an integer $1 \leq b \leq p^g-2$ such that

$$1+\beta^a+\beta^b = 0.$$

Let us define

$$c(x) = 1 + x^{uat^{-1}} + x^{ubt^{-1}}, \tag{4.3}$$

where the inverse t^{-1} is calculated in the ring $\mathbb{Z}_{p^g-1}$. To check if $c(x)$ is a codeword in $C_{t_1,\ldots,t_s}$, according to Lemma 1.16 we need to verify if γ^{t_j} are roots of the polynomial $c(x)$ for $1 \leq j \leq s$. Since $t_j \in K_g(t)$, there are non-negative integers k_1 and k_2 such that $t_j = k_1(p^g-1) + p^{k_2}t$. Thus,

$$\begin{aligned}
c(\gamma^{t_j}) &= 1 + \gamma^{ut_jat^{-1}} + \gamma^{ut_jbt^{-1}} \\
&= 1 + \beta^{t_jat^{-1}} + \beta^{t_jbt^{-1}} \\
&= 1 + \beta^{p^{k_2}tat^{-1}} + \beta^{p^{k_2}tbt^{-1}} \\
&= 1 + \beta^{p^{k_2}a} + \beta^{p^{k_2}b} \\
&= (1+\beta^a+\beta^b)^{p^{k_2}} \\
&= 0.
\end{aligned}$$

Hence, $C_{t_1,\ldots,t_s}$ has at least this codeword with weight 3. We have p^g-2 pairs (a,b) that generate codewords of this form. However, notice that (a,b) and (b,a) generate the same codeword, thus there are $(p^g-2)/2$ codewords of this form. By counting the codewords obtained from shifts and multiplications by constants of these codewords, we obtain the lower bound for A_3. ■

For instance, consider $p=5$, $m=2$, $g=1$, and $t_1, t_2 = 2, 5$. The constants 1 and 9 are not in the same 5-cyclotomic class mod 24, but are in the same 5-cyclotomic class mod 4,

i.e., they are both in $K_1(1)$. We will use this to construct a codeword with weight 3. We have $u = (25-1)/(5-1) = 6$ and can verify that $\beta = \gamma^6$ satisfies the equation

$$1 + \beta + \beta^5,$$

thus $c(x) = 1 + x^6 + x^{30}$ is a codeword with weight 3 in $C_{2,5}$.

We will now introduce a class of cyclic codes for which there is an interesting criterion for the existence of weight 3 codewords.

Let us consider binary cyclic codes $C_{1,t}$ of length $n = q - 1 = 2^m - 1$ and generator polynomial $g(x) = g_1(x)g_t(x)$. From Theorem 4.1 we know that this code can't have codewords with weight 2. Hence the next step in determining the existence of low weight codewords is to determine if the minimum distance is 3. The parity check matrix for this polynomial is

$$H = \begin{bmatrix} 1 & \gamma^1 & \gamma^2 & \cdots & \gamma^{(n-1)} \\ 1 & \gamma^t & \gamma^{2t} & \cdots & \gamma^{(n-1)t} \end{bmatrix}, \tag{4.4}$$

Since the code is cyclic a codeword can be shifted so the coordinates with ones are $0, i, j$ where $0 < i < j \leq q-2$. Multiplying this codeword by H, we have that the codeword is in the code if and only if

$$\begin{cases} 1 + \gamma^i + \gamma^j = 0, \\ 1 + \gamma^{it} + \gamma^{jt} = 0. \end{cases}$$

Since the indices run through 1 to $q-2$, there are indices i, j that construct a weight 3 codeword if and only if there is a solution (x, y) with $x, y \in \mathbb{F}_q \setminus \{0, 1\}$ to the system

$$\begin{cases} 1 + x + y = 0, \\ 1 + x^t + y^t = 0. \end{cases}$$

We can isolate $y = x + 1$ and substitute it into the second equation to obtain that the condition is that the polynomial

$$U_t(x) := 1 + x^t + (1 + x)^t$$

should have a root in $\mathbb{F}_q \setminus \{0, 1\}$.

Theorem 4.3. *Let $C_{1,t}$ be the binary cyclic code with minimal polynomial $g(x) = g_1(x)$ $g_t(x)$, and let us suppose that the irreducible factors of*

$$U_t(x) = 1 + x^t + (1 + x)^t$$

are $x, (x+1), f_1(x), \ldots, f_l(x)$*. Then the code has a codeword with weight* 3 *if and only if at least one of the irreducible factors* $f_1, \ldots, f_l$ *has a root in* $\mathbb{F}_q$*.*

Specifically, if $m_j := \deg f_j(x)$ *for* $1 \leq j \leq l$*, then*

i) If $m_j \nmid m$ *for all* $1 \leq j \leq l$ *then the code has no codewords with weight* 3*.*

ii) If $m_j \mid m$ *for some* $1 \leq j \leq l$ *then the code has codewords with weight* 3*.*

Proof. The roots of $U_t(x)$ in $\mathbb{F}_q \setminus \{0,1\}$, if they exist, must come from the irreducible factors different from x and $x+1$. The roots of an irreducible polynomial in $\mathbb{F}_2[x]$ must be in the extension of degree equal to the degree of the polynomial. Thus, if $m_j = \deg f_j(x)$, the roots are in $\mathbb{F}_q$ if and only if $\mathbb{F}_{2^{m_j}} \subseteq \mathbb{F}_q$, which happens if and only if $m_j \mid m$. Thus, if no irreducible polynomial is such that the degree of the polynomial divides m, then there are no codewords with weight 3 in the code $C_{1,t}$. Similarly, if $\deg f_j(x) \mid m$ for some $1 \leq j \leq l$ then the code has a codeword with weight 3. ■

For instance, no matter what is the positive integer m such that $q = 2^m$, the code $C_{1,3}$ with length $n = q - 1$ must have minimum distance greater than 3, since $U_3(x) = x^2 + x$ only has 0 and 1 as roots.

Corollary 4.4. *Let* $C_{1,t}$ *be the binary cyclic code with minimal polynomial* $g(x) = g_1(x)$ $g_t(x)$*. Then* $C_{1,t}$ *has minimum distance equal to* 3 *if and only if*

$$\gcd(U_t(x), x^q + x) \neq x(x+1).$$

Proof. The factors $x, x+1$ are present both in $U_t(x)$ and $x^q + x$. Any irreducible factor in $U_t(x)$ has degree dividing m if and only if its roots are in $\mathbb{F}_q$, which happens if and only if it divides $x^q + x$. It is straightforward to verify that $x^2 + x$ divides both $U_t(x)$ and $x^q + x$, and that the multiplicity on $U_t(x)$ is 1. Thus there are other irreducible factors if and only if $\gcd(U_t(x), x^q + x) \neq x(x+1)$. ■

The factoring of polynomials over finite fields is also a challenging problem, with few cases where it can be done without exhaustive search. The following corollary is a case where the minimum distance of 3 can be easily verified.

Corollary 4.5. *Let* $m = p_1^{e_1} \cdots p_s^{e_s}$ *be the decomposition of* m *into prime factors. If* $t < p_i + 3$ *for all* $1 \leq i \leq s$*, the code* $C_{1,t}$ *has minimum distance* $d > 3$*.*

Proof. Let us suppose that the irreducible factors of

$$U_t(x) = 1 + x^t + (1+x)^t$$

are $x, (x+1), f_1(x), \ldots, f_l(x)$, and denote $m_j = \deg f_j$ for all $1 \leq j \leq l$. Theorem 4.3 tells us that the code has minimum distance 3 if and only if $m_j \mid m$ for at least one $1 \leq j \leq l$. This can only happen if at least one of the primes p_i divides m_j. Since the degree of $U_t(x)/(x(x+1))$ is $t-3$, the condition that $p_i > t-3$ for every $1 \leq i \leq s$ implies that p_i is greater than m_j for every $1 \leq j \leq l$, since $t-3 \geq m_j$. This implies no prime factor of m divides any of the degrees of the irreducible factors and thus no degree divides m. Therefore, there are no codewords with weight 3. ■

For instance, let $q = 2^{31}$ and $t = 33$. Since $t < 31 + 3 = 34$, Corollary 4.5 implies that there are no codewords with weight 3 in the code.

We remark that Corollary 4.5 is a generalization of a similar result where m is a prime number, which can be seen in [20, Proposition 3].

Thus the number of codewords of weight 3 in this polynomial is related to the number of roots of the irreducible polynomials in $\mathbb{F}_2[x]$. But how can we count the number of codewords of weight 3 from the number of solutions in $\mathbb{F}_q \setminus \{0,1\}$ of $U_t(x)$? As we have seen, each root $\beta = \gamma^i$ has another corresponding root $\beta' = \gamma^j$ such that the codeword

$$c(x;i,j) = 1 + x^i + x^j$$

is in the code. Notice that we can swap β and β' and still obtain the same codeword, so the number of codewords of the form $c(x;i,j)$ in the code is half the number of roots of $U_t(x)/(x(x+1))$ in $\mathbb{F}_q$. Then all codewords of weight 3 can be obtained by shifting these codewords $n = q-1$ times. However, doing this will count every codeword three times, so the total number of codewords of weight 3 is given by the following corollary:

Corollary 4.6. *The number of codewords with weight* 3 *in* $C_{1,t}$ *is*

$$A_3 = \frac{q-1}{6} \cdot \left(\deg(\gcd(U_t(x), x^q + x)) - 2\right).$$

Proof. According to Corollary 4.4, every root of $U_t(x)$ that is in $\mathbb{F}_q$ is in $\gcd(U_t(x), x^q + x)$. By the discussion above we know that the total number of codewords is $(q-1)/6$ times the total number of roots excluding 0 and 1. ■

For instance, let $q = 2^4$, $t = 7$ and consider the binary cyclic code $C_{1,t}$ over $\mathbb{F}_q$, with primitive element γ that satisfies $\gamma^4 + x + 1 = 0$. Since $\gcd(U_7(x), x^{16} + x) = x^4 + x$, the

Corollary (4.6) implies the code $C_{1,7}$ of length $n = 15$ has

$$A_3 = \frac{16-1}{6} \cdot (4-2) = 5$$

codewords with weight 3. In fact, the roots of $U_7(x)$ in $\mathbb{F}_q \setminus \{0,1\}$ are γ^5 and γ^{10}, which correspond to the codeword

$$(1,0,0,0,0,1,0,0,0,0,1,0,0,0,0),$$

and the codewords obtained by shifting it, totaling 5 such codewords.

4.3 Low weight codewords in binary cyclic codes

Let $q = 2^m$, γ be a primitive element of $\mathbb{F}_q$, and $C_{t_1,\ldots,t_s}$ be the code of length $n = q-1$ generated by the polynomial $g(x) = g_{t_1}(x)g_{t_2}(x)\cdots g_{t_s}(x)$ as defined previously and let $A_0, A_1, A_2, \ldots, A_n$ be its weight distribution. We will show how to relate A_w to N_w, the number of solutions in $(\mathbb{F}_q)^w$ of the system:

$$\begin{cases} x_1^{t_1} + \cdots + x_w^{t_1} = 0, \\ \vdots \\ x_1^{t_s} + \cdots + x_w^{t_s} = 0. \end{cases} \tag{4.5}$$

In polynomial notation, codewords with weight w are of the form

$$c(i_1, \ldots, i_w) := x^{i_1} + \cdots + x^{i_w},$$

where $0 \le i_1 < i_2 < \cdots < i_w \le q-2$ are the ordered indices. Multiplying any of these codewords by the parity check matrix (1.16), we conclude that the codeword is in $C_{t_1,\ldots,t_s}$ if and only if

$$\begin{cases} \gamma^{i_1 t_1} + \cdots + \gamma^{i_w t_1} = 0, \\ \vdots \\ \gamma^{i_1 t_s} + \cdots + \gamma^{i_w t_s} = 0. \end{cases}$$

Since γ is a primitive element and the indices run from 0 to $q-2$, each γ^{i_j} corresponds to a $x_j \in \mathbb{F}_q^*$. Thus, A_w is the number of solutions over $\mathbb{F}_q^*$ of the system (4.5), where the indices are ordered by their discrete logarithms, i.e, $\log_\gamma x_1 < \log_\gamma x_2 < \cdots < \log_\gamma x_w$. We remark that these solutions have no repetition of variables, unlike N_w, where the solutions can include zeros, be unordered, and may have repetitions.

We will introduce notation for orderings and repetitions within tuples. A tuple with elements in $\mathbb{F}_q$ is said to be ordered if its coordinates are arranged in ascending order according to the discrete logarithm with respect to γ, with $0 \in \mathbb{F}_q$ placed in the first position if present. Ordered tuples with integer elements are ordered in the usual ascending order. A subtuple of an ordered tuple is a tuple formed by selecting a subset of the elements from the original tuple while preserving their relative order. For instance, $(2,4)$ is a subtuple of $(1,2,3,4)$.

We denote the sum of coordinates in a tuple v as $|v|$, and the number of coordinates as $\#v$. We call a tuple w-sum if the sum of its coordinates is w. For instance, for $v = (2,2,4,4,4,8)$ we have $|v| = 24$ and $\#v = 6$.

Let $v = (v_1, \ldots, v_w)$ be a tuple, and let k denote the number of distinct values appearing in its coordinates. The partition $P(v)$ is defined as the ordered integer tuple $P(v) = (a_1, \ldots, a_k)$, where each $a_i \in \mathbb{Z}_{>0}$ represents the number of repetitions of one of the distinct values. For instance, the partition for the tuple $v = (4,3,5,4,3,0,4)$ is $P(v) = (1,1,2,3)$, since 0 and 5 appear once each, 3 repeats twice and 4 repeats three times.

We will also refer to ordered tuples with positive integer coordinates as partitions, since they can be obtained as partitions of other tuples. For a partition tuple v, we define $\varepsilon(v)$ and $\theta(v)$ as the subtuples of v consisting only of the even and odd coordinates in v, respectively.

For a positive integer w, its assembly $S(w)$ is defined as the set of all ordered positive w-sum partitions, i.e.,

$$S(w) = \left\{ (v_1, \ldots, v_k) : 1 \le v_1 \le \cdots \le v_k \le n; \sum_{i=1}^{k} v_i = w \right\}.$$

For instance, $S(4) = \{(1,1,1,1),(1,1,2),(1,3),(2,2),(4)\}$.

For a tuple $v = (a_1, \ldots, a_k)$ with sum $|v| = w$, we define its factorial as $v! = a_1! \cdots a_k!$, and its binomial as

$$\binom{w}{v} := \binom{w}{a_1, a_2, \ldots, a_k} = \frac{w!}{a_1! a_2! \cdots a_k!}.$$

We can now state our main result. The following theorem relates the weight distribution of a cyclic code to the number of solutions of a system of diagonal equations.

Theorem 4.7. *Consider the binary cyclic code $C_{t_1,\ldots,t_s}$ of length $n = q - 1$ with generator polynomial $g(x) = g_{t_1}(x) \cdots g_{t_s}(x)$. Let $A_0, A_1, A_2, \ldots$ be the weight distribution of the code and, for any non-negative integer w, let N_w be defined as in (4.5). The following*

relationship between N_w and the weight distribution of $C_{t_1,\ldots,t_s}$ holds:

$$N_w = \sum_{v \in S(w)} (A_{\#\theta(v)} + A_{\#\theta(v)-1}) \frac{\#\theta(v)!}{P(\theta(v))!} \cdot \binom{q - \#\theta(v)}{\#\varepsilon(v)} \frac{\#\varepsilon(v)!}{P(\varepsilon(v))!} \cdot \binom{w}{v}. \tag{4.6}$$

Proof. Let N_w be the set of solutions to (4.5) over $\mathbb{F}_q$, with no restrictions. We will first divide this set into the union of subsets $N_w(v)$, where each subset contains solutions that are tuples of length n with a partition tuple equal to v, for each partition v with sum n. We will show how to relate the values of $N_w(v)$ to the weight distributions A_k where $k \leq w$.

Let us fix a particular partition $v \in S(w)$ and generate every solution that has this partition. We will determine what values can be put into the coordinates to form a solution, how many repetitions they have, and in how many ways they can be arranged.

Let $\theta(v)$ and $\varepsilon(v)$ be the ordered subtuples containing the odd and even values in v, respectively. The values with even repetition vanish in (4.5), while the values with odd repetition reduce to only one summand in (4.5) and thus must be solutions to

$$\begin{cases} y_1^{t_1} + \cdots + y_{\#\theta(v)}^{t_1} = 0, \\ \vdots \\ y_1^{t_s} + \cdots + y_{\#\theta(v)}^{t_s} = 0, \end{cases} \tag{4.7}$$

with the added restriction that they must be distinct. There are then two possibilities: if one of the values is 0, the remaining $\#\theta(v) - 1$ values correspond to codewords with weight $\#\theta(v) - 1$, giving us $A_{\#\theta(v)-1}$ choices; if none of the values is 0, then there are $A_{\#\theta(v)}$ choices for the values. Thus, the total number of ordered tuples of values that are solutions to (4.7) is $(A_{\#\theta(v)-1} + A_{\#\theta(v)})$. Next, we need to choose how many repetitions each value has. We can select one of the odd repetition values in $\#\theta(v)$ for each ordered tuple of values with odd repetition, which can be done in $\#\theta(v)!/P(\theta(v))!$ ways.

The values with even repetition vanish in (4.5) so for any choice of values with odd repetition the only restriction is that we cannot choose values that have already been chosen. Thus, we have

$$\binom{q - \#\theta(v)}{\#\varepsilon(v)}$$

tuples of values with even repetition. Similarly to the values with odd repetition, the number of repetitions of the values with even repetition can be chosen $\#\varepsilon(v)!/P(\varepsilon(v))!$ ways.

Finally, to form a solution tuple $(x_1, \ldots, x_w)$ of (4.5), we just need to distribute these coordinates according to the partition, which can be done $\binom{w}{v}$ ways.

The total number of solutions with the partition v is then obtained multiplying every independent choice, giving us

$$N_w(v) = (A_{\#\theta(v)} + A_{\#\theta(v)-1}) \frac{\#\theta(v)!}{P(\theta(v))!} \cdot \binom{q - \#\theta(v)}{\#\varepsilon(v)} \frac{\#\varepsilon(v)!}{P(\varepsilon(v))!} \cdot \binom{w}{v}.$$

Summing it over every w-sum partition gives us the value expression (4.6). ■

In the next section we will see some applications of this result.

For instance, a BCH code of designed distance δ is a cyclic code with generator polynomial $g(x) = \operatorname{lcm}(g_b(x), g_{b+1}(x), \ldots, g_{b+\delta-2}(x))$ where b is a positive integer. A classic result about such BCH codes is that their minimum distance d is greater or equal to δ [21, Theorem 5.1.1].

Let b and δ be parameters for a binary BCH code, $0 < w < d$ an odd integer and N_w be the number of solutions of the system

$$\begin{cases} x_1^b + \cdots + x_w^b = 0, \\ x_1^{b+1} + \cdots + x_w^{b+1} = 0, \\ \vdots \\ x_1^{b+\delta-2} + \cdots + x_w^{b+\delta-2} = 0. \end{cases}$$

Since $\#\theta(v) < w$ for every partition and $A_u = 0$ for $0 < u < d$, the term $(A_{\#\theta(v)} + A_{\#\theta(v)-1})$ will vanish whenever $\#\theta(v) \notin \{0, 1\}$, significantly reducing the number of terms in (4.6). When w is odd, there are no partitions of w without odd repetitions, so every summand that does not vanish must have $\#\theta(v) = 1$, further reducing the number of terms. In this case, every summand will be of the form

$$\binom{q-1}{\#\varepsilon(v)} \frac{\#\varepsilon(v)!}{P(\varepsilon(v))!} \cdot \binom{w}{v}.$$

An example of a binary BCH code is the code with length $n = 15$, designed distance $\delta = 5$, and generator polynomial $g(x) = \operatorname{lcm}(g_1(x), g_2(x), g_3(x), g_4(x)) = g_1(x)g_3(x)$. Using the general example in this instance we will determine the number N_3 of solutions to the system

$$\begin{cases} x_1 + x_2 + x_3 = 0, \\ x_1^2 + x_2^2 + x_3^2 = 0, \\ x_1^3 + x_2^3 + x_3^3 = 0, \\ x_1^4 + x_2^4 + x_3^4 = 0. \end{cases}$$

The partition $S(3) = \{(1,1,1),(1,2),(3)\}$ is such that the only tuples that have $\#\theta(v) = 1$ are $(1,2)$ and (3), giving us only two terms we will need to sum.

$$\begin{aligned} N_3 &= \binom{15}{1}\frac{1!}{1!}\cdot\binom{3}{1,2} + \binom{15}{0}\binom{3}{3} \\ &= 46. \end{aligned}$$

In the next section, we will demonstrate an application where we determine the number of codewords with weights 3, 4, and 5 in a cyclic code.

4.4 An application

We will first introduce an example where the number of solutions of a system of diagonal equations of the form (4.5) can be computed.

Let r be a positive integer and $m = 2r$. Let $q = 2^m$, such that $q^{1/2} = 2^r$, and let us define $t = q^{1/2} + 1$. This choice of t is significant later as it is such that the norm function $N : \mathbb{F}_{2^m} \to \mathbb{F}_{2^r}$ is defined as

$$N(x) = x^{(2^m-1)/(2^r-1)} = x^t.$$

For this code, N_w is the number of solutions $(x_1,\ldots,x_w)$ over $\mathbb{F}_q$ of the system

$$\begin{cases} x_1 + x_2 + \cdots + x_w = 0, \\ x_1^t + x_2^t + \cdots + x_w^t = 0. \end{cases} \tag{4.8}$$

As seen in Theorem 4.7, this value is related to the number of codewords with weight w in the code.

Let us denote trace functions as

$$\begin{aligned} \mathrm{Tr}_{a|b} : \mathbb{F}_{2^a} &\to \mathbb{F}_{2^b} \\ x &\to \sum_{i=0}^{a/b-1} x^{2^{bi}}, \end{aligned}$$

and

$$\psi_1 := (-1)^{\mathrm{Tr}_{r|1}(x)}$$

be the canonical additive character from $\mathbb{F}_{2^r}$ to $\mathbb{C}$ and $\psi = \psi_1 \circ \mathrm{Tr}_{m|r}(x)$ be the canonical additive character from $\mathbb{F}_q$ to $\mathbb{C}$. For $\alpha, \beta \in \mathbb{F}_q$ let us define the sum

$$S(\alpha,\beta) = \sum_{x\in\mathbb{F}_q} \psi(\alpha x + \beta x^t). \tag{4.9}$$

Let $C_{1,t}$ be the cyclic code of length $n = q-1$ and generator $g(x) = g_1(x)g_t(x)$.

Theorem 4.8. *Let $C_{1,t}$ as defined above we have*

$$N_w = \frac{1}{q^2} \sum_{\alpha,\beta \in \mathbb{F}_q} \big(S(\alpha,\beta)\big)^w .$$

Proof. Corollary 1.5 allows us to use character sums over ψ as indicator functions. For each vector $(x_1,\ldots,x_w)$ we have that

$$\left(\frac{1}{q}\sum_{\alpha\in\mathbb{F}_q} \psi(\alpha(\sum_{i=1}^{n} x_i))\right)\left(\frac{1}{q}\sum_{\beta\in\mathbb{F}_q} \psi(\beta(\sum_{i=1}^{w} x_i^t))\right) = \begin{cases} 1 & \text{if it is a solution to system (4.8),} \\ 0 & \text{otherwise.} \end{cases}$$

Thus,

$$N_w = \sum_{x_1\in\mathbb{F}_q} \cdots \sum_{x_w\in\mathbb{F}_q} \left(\frac{1}{q}\sum_{\alpha\in\mathbb{F}_q} \psi(\alpha(\sum_{i=1}^{w} x_i)\right)\left(\frac{1}{q}\sum_{\beta\in\mathbb{F}_q} \psi(\beta(\sum_{i=1}^{w} x_i^t))\right),$$

which can be rewritten as

$$N_w = \frac{1}{q^2} \sum_{\alpha,\beta \in \mathbb{F}_q} \big(\psi(\alpha x + \beta x^t)\big)^w . \tag{4.10}$$

■

Hence being able to compute the distribution of values of $S(\alpha,\beta)$ should be enough to compute N_w. Let us define the l-dimensional Kloosterman sum over $\mathbb{F}_{2^r}$ with $u \in \mathbb{F}_{2^r}^*$ as

$$k_l(u) = \sum_{x_1,\ldots,x_l \in \mathbb{F}_{2^r}^*} \psi(x_1 + \cdots + x_l + u x_1^{-1}\cdots x_l^{-1}).$$

The following result links exponential character sums to this kind of sum.

Lemma 4.9. *Let $q = 2^{ar}$, $t = (q-1)/(2^r-1)$, and $\alpha \in \mathbb{F}_q^*$, $\beta \neq 0$ and $a > 1$. Then*

$$\sum_{x\in\mathbb{F}_q^*} \psi(\alpha x^{2^r-1}) = (-1)^{a-1}(2^r-1)k_{a-1}(N(\alpha)).$$

Proof. See Theorem 3 in [22]. ■

Theorem 4.10. *Let $q = 2^{2r}$, $t = (q-1)/(2^r-1)$, and $\alpha,\beta \in \mathbb{F}_q$. Then*

$$\sum_{x\in\mathbb{F}_q} \psi(\alpha x^t + \beta x) = \begin{cases} q & \text{if } \beta = 0 \text{ and } Tr(\alpha) = 0, \\ -q^{1/2} & \text{if } \beta = 0 \text{ and } Tr(\alpha) \neq 0. \\ 0 & \text{if } \beta \neq 0 \text{ and } Tr(\alpha) = 0, \\ q^{1/2}\psi_1(-Tr(\alpha)^{-1}\beta^t) & \text{if } \beta \neq 0 \text{ and } Tr(\alpha) \neq 0, \end{cases} \tag{4.11}$$

Giving $S(\alpha,\beta)$ the frequency distribution in Table 4.1.

Proof. Let us suppose $\beta = 0$. In this case,

$$\begin{aligned} S(\alpha,0) &= \sum_{x\in\mathbb{F}_q} \psi(\alpha x^t) \\ &= 1 + \sum_{x\in\mathbb{F}_q^*} \psi(\alpha x^t) \\ &= 1 + (2^r+1) \sum_{y\in\mathbb{F}_{2^r}^*} \psi(\alpha y) \\ &= 1 + (2^r+1) \sum_{y\in\mathbb{F}_{2^r}^*} \psi_1(y\mathrm{Tr}_{m|r}(\alpha)) \\ &= \begin{cases} 1+(2^r+1)(2^r-1) & \text{if } \mathrm{Tr}_{m|r}(\alpha)=0, \\ 1+(2^r+1)(-1) & \text{otherwise.} \end{cases} \\ &= \begin{cases} q & \text{if } \mathrm{Tr}_{r|1}(\alpha)=0, \\ -q^{1/2} & \text{otherwise.} \end{cases} \end{aligned} \tag{4.12}$$

Since $Tr_{r|1} : \mathbb{F}_q \to \mathbb{F}_{2^r}$ is uniform, then $\mathrm{Tr}_{r|1}(\alpha) = 0$ for 2^r values of α and $\mathrm{Tr}_{r|1}(\alpha) \neq 0$ for the other $q - 2^r$ of them.

Now let us suppose $\beta \neq 0$. Denote the canonical additive character from $\mathbb{F}_{2^r}$ to $\mathbb{C}$ as ψ_1, and let γ be a primitive element of $\mathbb{F}_q^*$. Then we can rewrite

$$\begin{aligned} S(\alpha,\beta) &= \sum_{x\in\mathbb{F}_q} \psi(\alpha x^t + \beta x) \\ &= 1 + \sum_{k=0}^{q-2} \psi(\alpha\gamma^{kt} + \beta\gamma^k). \end{aligned}$$

Using euclidean division we can rewrite the indices as $k = i + j(2^r-1)$ where $0 \le i \le 2^r-2$ and $0 \le j \le 2^r$. We obtain

$$\begin{aligned} S(\alpha,\beta) &= 1 + \sum_{i=0}^{2^r-2} \sum_{j=0}^{2^r} \psi(\alpha\gamma^{i(2^r+1)}\gamma^{j(2^r+1)(2^r-1)} + \beta\gamma^i\gamma^{j(2^r-1)}) \\ &= 1 + \sum_{i=0}^{2^r-2} \psi(\alpha\gamma^{i(2^r+1)}) \sum_{j=0}^{2^r} \psi(\beta\gamma^i\gamma^{j(2^r-1)}) \\ &= 1 + \sum_{i=0}^{2^r-2} \psi(\alpha\gamma^{i(2^r+1)}) \cdot \frac{1}{2^r-1} \sum_{x\in\mathbb{F}_q^*} \psi(\beta\gamma^i x^{2^r-1}). \end{aligned}$$

Lemma 4.9 implies that $\sum_{x\in\mathbb{F}_q^*} \psi(\beta\gamma^i x^{2^r-1}) = -(2^r-1)\cdot k_1(N(\beta\gamma^i))$, thus we can rewrite

$S(\alpha,\beta)$ as

$$\begin{aligned}
S(\alpha,\beta) &= 1-\sum_{i=0}^{2^r-2}\psi(\alpha\gamma^{i(2^r+1)})\sum_{y\in\mathbb{F}_{2^r}^*}\psi_1(y+N(\beta\gamma^i)y^{-1})\\
&= 1-\sum_{i=0}^{2^r-2}\psi(\alpha\gamma^{it})\sum_{y\in\mathbb{F}_{2^r}^*}\psi_1(y+\beta^t\gamma^{it}y^{-1})\\
&= 1-\sum_{x\in\mathbb{F}_{2^r}^*}\psi(\alpha x)\sum_{y\in\mathbb{F}_{2^r}^*}\psi_1(y+\beta^t xy^{-1})\\
&= 1-\sum_{x\in\mathbb{F}_{2^r}^*}\psi_1(x\mathrm{Tr}_{m|r}(\alpha))\sum_{y\in\mathbb{F}_{2^r}^*}\psi_1(y+\beta^t xy^{-1})\\
&= 1+\sum_{y\in\mathbb{F}_{2^r}^*}\psi_1(y)-\sum_{x\in\mathbb{F}_{2^r}}\sum_{y\in\mathbb{F}_{2^r}^*}\psi_1(x\mathrm{Tr}_{m|r}(\alpha)+y+\beta^t xy^{-1})\\
&= -\sum_{y\in\mathbb{F}_{2^r}^*}\psi_1(y)\sum_{x\in\mathbb{F}_{2^r}}\psi_1(x(\mathrm{Tr}_{m|r}(\alpha)+\beta^t y^{-1}))
\end{aligned}$$

Now we have two possibilities. If $\mathrm{Tr}_{m|r}(\alpha)=0$, then $\sum_{x\in\mathbb{F}_{2^r}}\psi_1(x\beta^t y^{-1})=0$ for every $y\in\mathbb{F}_{2^r}^*$. Thus for $\beta\neq 0$ and $\mathrm{Tr}_{m|r}(\alpha)=0$ we have

$$S(\alpha,\beta)=0, \tag{4.13}$$

with frequency $q^{3/2}-q^{1/2}$.

If $\mathrm{Tr}_{m|r}(\alpha)\neq 0$, then

$$\sum_{x\in\mathbb{F}_{2^r}}\psi_1(x(\mathrm{Tr}_{m|r}(\alpha)+\beta^t y^{-1}))=\begin{cases}q^{1/2} & \text{if } y=-\mathrm{Tr}_{m|r}(\alpha)^{-1}\beta^t\\ 0 & \text{otherwise.}\end{cases}$$

Hence, in this case we have

$$S(\alpha,\beta)=-q^{1/2}\psi_1(-\mathrm{Tr}_{m|r}(\alpha)^{-1}\beta^t). \tag{4.14}$$

Combining (4.12), (4.13) and (4.14) we obtain (4.11). To compute the explicit values and frequencies of the case where $\mathrm{Tr}_{m|r}(\alpha)\neq 0$ and $\beta\neq 0$, we rewrite (4.14) as:

$$-q^{1/2}\psi_1(-\mathrm{Tr}_{m|r}(\alpha)^{-1}\beta^t)=\begin{cases}-q^{1/2}, & \text{if } \mathrm{Tr}_{r|1}(-\mathrm{Tr}_{m|r}(\alpha)^{-1}\beta^t)=0,\\ q^{1/2}, & \text{if } \mathrm{Tr}_{r|1}(-\mathrm{Tr}_{m|r}(\alpha)^{-1}\beta^t)=1.\end{cases}$$

We have that $\mathrm{Tr}_{m|r}:\mathbb{F}_q\to\mathbb{F}_{2^r}$ maps the values such that $\mathrm{Tr}_{m|r}(\alpha)\neq 0$ uniformly into $\mathbb{F}_{2^r}^*$, while $N:\mathbb{F}_q\to\mathbb{F}_{2^r}$ maps the values $\beta\neq 0$ uniformly into $\mathbb{F}_{2^r}^*$. Thus as α and β run through,

$S(\alpha,\beta)$	Frequency
0	$q^{3/2}-q^{1/2}$
$-q^{1/2}$	$\frac{q^2+q}{2}-q^{3/2}$
$q^{1/2}$	$\frac{q^2-q}{2}$
q	$q^{1/2}$

Table 4.1: Frequency distribution of $S(\alpha,\beta)$ when $q=2^{2r}$ and $t=2^r+1$

$-\mathrm{Tr}_{m|r}(\alpha)^{-1}\beta^t$ evaluates to every value in $\mathbb{F}_{2^r}^*$ with $(q-2^r)(q-1)/(2^r-1)=q^{3/2}-q^{1/2}$ repetitions. $\mathrm{Tr}_{r|1}$ maps $2^{r-1}-1$ values in $\mathbb{F}_{2^r}^*$ to 0 and 2^{r-1} values to 1. Multiplying this by the repetitions, we have the following frequencies for values of $S(\alpha,\beta)$ when $\mathrm{Tr}_{m|r}(\alpha)\neq 0$ and $\beta\neq 0$.

$$S(\alpha,\beta)=\begin{cases}-q^{1/2}, & (q^{3/2}-q^{1/2})(\frac{q^{1/2}}{2}-1) \text{ times},\\ q^{1/2}, & (q^{3/2}-q^{1/2})\frac{q^{1/2}}{2} \text{ times}.\end{cases} \tag{4.15}$$

We add up all the values and frequencies for each case to obtain the complete Table 4.1. ■

Theorem 4.11. *Under the same assumptions as Theorem 4.10, we have*

$$N_w=\begin{cases}q^{w-3/2}+q^{(w-1)/2}-q^{(w-2)/2}, & \text{if } w \text{ is odd},\\ q^{w-3/2}+q^{w/2}-q^{(w-1)/2}, & \text{if } w \text{ is even}.\end{cases} \tag{4.16}$$

Proof. We will use the values and frequencies in Table 4.1 to compute N_w.

$$\begin{aligned}
N_w &= \frac{1}{q^2}\sum_{\alpha,\beta\in\mathbb{F}_q} S(\alpha,\beta)^w\\
&= \frac{1}{q^2}\left(\begin{array}{c} 0^n\cdot(q^{3/2}-q^{1/2})+(-q^{1/2})^w\cdot\left(\frac{q^2+q}{2}-q^{3/2}\right)\\ +(q^{1/2})^w\cdot\left(\frac{q^2-q}{2}\right)+(q^n)\cdot q^{1/2}\end{array}\right)\\
&= \frac{1}{q^2}\left(\begin{array}{c}(-1)^w\left(\frac{q^{w/2+2}+q^{w/2+1}}{2}-q^{(w+3)/2}\right)\\ +\frac{q^{w/2+2}-q^{w/2+1}}{2}+q^{w+1/2}\end{array}\right)\\
&= (-1)^w\left(\frac{q^{w/2}+q^{w/2-1}}{2}-q^{(w-1)/2}\right)+\frac{q^{w/2}-q^{w/2-1}}{2}+q^{w-3/2}\\
&= \begin{cases}q^{w-3/2}+q^{(w-1)/2}-q^{(w-2)/2}, & \text{if w is odd},\\ q^{w-3/2}+q^{w/2}-q^{(w)/2}, & \text{if w is even}.\end{cases}
\end{aligned}$$

■

Corollary 4.12. *Let $C_{1,t}$ be as described previously. Then*

$$A_3 = \frac{q^{3/2} - 2q - q^{1/2} + 2}{6};$$
$$A_4 = \frac{q^{5/2} - 2q^2 - 5q^{3/2} + 10q + 4q^{1/2} - 8}{24};$$
$$A_5 = \frac{q^{7/2} - 15q^{5/2} + 16q^2 + 54q^{3/2} - 80q - 40q^{1/2} + 64}{120};$$

Proof. It can be verified that $A_0 = 1$, $A_1 = 0$. Theorem 4.1 implies that $A_2 = 0$, and by our convention $A_{-1} = 0$. We will use Theorem 4.7 for the recursive relation. For $w = 3$ we have

$$P(3) = \{(1,1,1),(1,2),(3)\},$$

and thus,

$$\begin{aligned} N_3 = &(A_3 + A_2)\frac{3!}{3!}\binom{q-3}{0}\frac{0!}{0!}\binom{3}{1,1,1} \\ &+ (A_1 + A_0)\frac{1!}{1!}\binom{q-1}{1}\frac{1!}{1!}\binom{3}{1,2} \\ &+ (A_1 + A_0)\frac{1!}{1!}\binom{q}{0}\frac{0!}{0!}\binom{3}{3} \\ = &\, 3!A_3 + 3(q-1) + 1. \end{aligned}$$

As such,

$$A_3 = \frac{N_3 - 3(q-1) - 1}{3!}.$$

Thus the value of N_3 is tied to the value of A_3. According to Theorem 4.11, for this particular code $C_{1,t}$ we have

$$N_3 = q^{3/2} + q - q^{1/2},$$

and thus

$$A_3 = \frac{q^{3/2} - 2q - q^{1/2} + 2}{6}.$$

For $w = 4$, we have

$$P(4) = \{(1,1,1,1),(1,1,2),(1,3),(2,2),(4)\},$$

hence,

$$\begin{aligned}
N_4 = (A_4 + A_3)\frac{4!}{4!}\binom{4}{1,1,1,1} \\
&+ (A_2 + A_1)\frac{2!}{2!}\binom{q-2}{1}\frac{1!}{1!}\binom{4}{1,1,2} \\
&+ (A_2 + A_1)\frac{2!}{1!1!}\binom{4}{1,3} \\
&+ \binom{q}{2}\frac{2!}{2!}\binom{4}{2,2} \\
&+ \binom{q}{1}\frac{1!}{1!}\binom{4}{4} \\
&= 4!(A_4 + A_3) + 3q(q-1) + q,
\end{aligned}$$

and thus

$$A_4 = \frac{N_4 - 3q(q-1) - q}{4!} - A_3.$$

According to Theorem 4.11, we have

$$N_4 = q^{5/2} + q^2 - q^{3/2},$$

which can be substituted in the previous expression to obtain

$$A_4 = \frac{q^{5/2} - 2q^2 - 5q^{3/2} + 10q + 4q^{1/2} - 8}{24}.$$

For $w = 5$, we have

$$P(4) = \{(1,1,1,1,1),(1,1,1,2),(1,1,3),(1,4)(1,2,2),(2,3),(4)\},$$

hence,

$$\begin{aligned}
N_5 =& (A_5 + A_4)\frac{5!}{5!}\binom{5}{1,1,1,1,1} \\
&+ (A_3 + A_2)\frac{3!}{3!}\binom{q-3}{1}\frac{1!}{1!}\binom{5}{1,1,1,2} \\
&+ (A_3 + A_2)\frac{3!}{1!2!}\binom{5}{1,1,3} \\
&+ (A_1 + A_0)\frac{1!}{1!}\binom{q-1}{1}\frac{1!}{1!}\binom{5}{1,4} \\
&+ (A_1 + A_0)\frac{1!}{1!}\binom{q-1}{2}\frac{2!}{2!}\binom{5}{1,2,2} \\
&+ (A_1 + A_0)\frac{1!}{1!}\binom{q-1}{1}\frac{1!}{1!}\binom{5}{2,3} \\
&+ (A_1 + A_0)\frac{1!}{1!}\binom{5}{5} \\
=& 5!(A_5 + A_4) + 60(q-3)A_3 + 60A_3 + 5(q-1) + 15(q-1)(q-2) + 10(q-1) + 1 \\
=& 5!(A_5 + A_4) + 60(q-2)A_3 + 15(q-1)^2 + 1.
\end{aligned}$$

Thus,

$$A_5 = \frac{N_5 - 60(q-2)A_3 - 15(q-1)^2 - 1}{120} - A_4.$$

By Theorem 4.11 we have

$$N_5 = q^{7/2} + q^3 - q^{5/2},$$

which can be substituted in the previous expression to obtain

$$A_5 = \frac{q^{7/2} - 15q^{5/2} + 16q^2 + 54q^{3/2} - 80q - 40q^{1/2} + 64}{120}.$$

■

We remark that the code $C_{1,t}$ is the dual of the code with parity check polynomial $g_1(x)g_t(x)$, which is a Hamming code. The weight distribution of this dual code was determined in [25], and a summary can be found in Theorem 4.1 and Table 2 of [26]. Using these distributions we can also use the MacWilliams equations (see section 7.1 in [21]) to determine the weight distribution of $C_{1,t}$. We remark that using MacWilliams equations is also computationally intensive, since the equations are recursive and become more complicated as w grows.

Let us conclude this chapter with some examples. Let $q = 2^4 = 16$, and $t = 2^2 + 1 = 5$, and let us consider the binary cyclic code $C_{1,5}$ with length $n = 15$. This is a code with minimum distance $d = 3$, and the weight distribution of the code is, using the formulas in Theorem 4.12,

$$A_3 = 5, \quad A_4 = 15, \quad A_5 = 60.$$

For another example, let

$$q = 2^8 = 1024, \quad t = 2^4 + 1 = 17,$$

and consider the binary cyclic code $C_{1,17}$ with length $n = 1023$. We have $A_2 = 0$, and according to Theorem 4.12, we have

$$A_3 = 5115, \quad A_4 = 1304325, \quad A_5 = 282290712.$$

BIBLIOGRAPHY

[1] José G. Coelho, F. E. Brochero Martínez. *Counting roots of fully triangular polynomials over finite fields.* Finite Fields and Their Applications, vol. 94, 102345, 2024.

[2] José G. Coelho. *Solutions of full equations related to diagonal equations.* https://arxiv.org/abs/2403.13133, 2024.

[3] José G. Coelho and F. E. Brochero Martínez. *Low-weight codewords in cyclic codes.* https://arxiv.org/abs/2407.18398, 2024.

[4] Kun Jiang, Wei Gao, Wei Cao. *Counting solutions to generalized Markoff-Hurwitz-type equations in finite fields.* Finite Fields and Their Applications, vol. 62, February 2020.

[5] Sun, Q. *On the Formula of the number of solutions of some equations over a finite field.* Chinese Annals of Mathematics Series A 18, pgs. 403-408, 1997.

[6] Wei Cao. *On generalized Markoff-Hurwitz-type Equations over finite fields.* Acta Applicandae Mathematicae, vol. 112, pgs. 275â€“281, 2010.

[7] Ruyun Wang, Binbin Wen, Wei Cao. *Degree matrices and enumeration of rational points of some hypersurfaces over finite fields.* Journal of Number Theory, vol. 177, pgs. 91-99, 2017.

[8] Ioulia Baoulina. *On the number of solutions of the equation* $a_1x_1^{m_1} + ... + a_nx_n^{m_n} = bx_1...x_n$ *in a finite field*. Acta Applicandae Mathematicae, vol. 85, pgs. 35-39, 2005.

[9] Leonard Carlitz. *Certain special equations in a finite field.* Monatshefte fÃ¼r Mathematik, vol. 58, pgs. 5-12, 1954.

[10] Lei Fu, Danqing Wan. *Mirror congruence for rational points on Calabi-Yau varieties.* Asian Journal of Mathematics, vol. 10, pgs. 1-10, 2006.

[11] Antonio Rojas-Leon, Daqing Wan. *Moment zeta functions for toric Calabi-Yau hypersurfaces.* Communications in Number Theory and Physics, vol. 1, pgs. 539â€“578, 2007.

[12] Bruce C. Berndt, Ronald J. Evans, Kenneth S. Williams. *Gauss and Jacobi Sums.* Monographies et Ã‰tudes de la Société Mathématique du Canada, (1998).

[13] Henri Cohen. *Number Theory Volume I: Tools and Diophantine Equations.* Graduate Texts in Mathematics, Vol. 239. Springer, (2007).

[14] Ronald Graham, Donald Knuth, Oren Patashnik. *Concrete Mathematics.* 2nd ed., Addison-Wesley, (1994).

[15] Rudolf Lidl, Harald Niederreiter. *Finite Fields.* Encyclopedia of Mathematics and its Applications, Cambridge University Press, 2nd ed., (1997).

[16] Wei Cao, Qi Sun. *On a class of equations with special degrees over finite fields.* Acta Arithmetica 130.2, 2007.

[17] Ruyun Wang, Binbin Wen, Wei Cao. *Degree matrices and enumeration of rational points of some hypersurfaces over finite fields.* Journal of Number Theory, Volume 177, pgs. 91-99, 2017.

[18] José Alves Oliveira, *On diagonal equations over finite fields.* Finite Fields and Their Applications, vol. 76, 101927, 2021.

[19] Ronald J. Evans. *Pure Gauss sums over finite fields.* Mathematika, vol. 28(2), pgs. 239-248, 1981.

[20] P. Charpin, A. Tietavainen, Z. Zinoviev. *On binary cyclic codes with minimum distance $d = 3$.* Probl. Peredachi Inf, vol. 33(4), pgs. 287-296, 1997.

[21] Huffman WC, Pless V. *Fundamentals of Error-Correcting Codes.* Cambridge University Press, (2003).

[22] Marko Moisio. *On the number of rational points on some families of Fermat curves over finite fields.* Finite Fields and Their Applications, vol. 13(3), pgs. 546-562, 2007.

[23] Marko Moisio, Kalle Hanto. *Kloosterman sum identities and low-weight codewords in a cyclic code with two zeros.* Finite Fields and Their Applications, vol. 13, pgs. 922â€“935, 2007.

[24] Charpin, Pascale. *Open problems on cyclic codes.* Course notes, https://api.semanticscholar.org/CorpusID:116345960, 2009.

[25] K. Tadao. *Weight distributions of bose-chaudhuri-hocquenghem codes.* Coordinated Science Laboratory Report no. R-317, 1966.

[26] Cunsheng Ding, Chunlei Li, Nian Li, Zhengchun Zhou. *Three-weight cyclic codes and their weight distributions.* Discrete Mathematics, vol. 339(2), pgs. 415-427, 2016.

[27] Hai Q. Dinh, Chengju Li, Qin Yue. *Recent progress on weight distributions of cyclic codes over finite fields.* J. Algebra Comb. Discrete Appl., vol. 2(1), pgs. 39-63, 2015.

ANNEX

In this annex, we present SageMath commands used to test and create examples of the formulas and results discussed. We began by defining functions to count roots of a polynomial or system of polynomials using the brute force approach.

```
from itertools import product

'''
This function counts the number of roots of a given polynomial.

Inputs:
- polynomial: The polynomial whose roots are to be counted.
- poly_ring: The polynomial ring in which the polynomial is defined.
- star: A boolean indicating whether to count over (F_q^*)^n instead of F_q^n.
- show: A boolean indicating whether to print the points that are roots.

Output: the number of roots of the polynomial.
'''
def count_points(polynomial, poly_ring, star=False, show=False):
    variables = poly_ring.gens()
    n = len(poly_ring.gens())
    base_field = poly_ring.base_ring()
    elements_in_coordinates = list(base_field)

    if star:
        elements_in_coordinates = elements_in_coordinates[1:]
    values_in_domain = product(elements_in_coordinates, repeat=n)

    count = 0
    for value in values_in_domain:
        if polynomial(value) == 0:
            count += 1
            if show:
                print(value)
    return count

# Example usage:
q = 5
R.<x,y> = PolynomialRing(GF(q))
f = x^2*y^3 + x*y^2
print(count_points(f, R, star=True, show=True))
```

```
'''
Inputs:
- polynomials: A list of polynomials.
- poly_ring: The polynomial ring in which the polynomials are defined.
- star: A boolean indicating whether to count over (F_q^*)^n instead of F_q^n.
- show: A boolean indicating whether to print the points that are roots.

Output: the number of roots that vanish in all the polynomials.
'''

def count_common_roots(polynomials, poly_ring, star=False, show=False):
    variables = poly_ring.gens()
    n = len(variables)
    base_field = poly_ring.base_ring()

    elements_in_coordinates = list(base_field)
    if star:
        elements_in_coordinates = elements_in_coordinates[1:]
    values_in_domain = product(elements_in_coordinates, repeat=n)

    count = 0
    for value in values_in_domain:
        if all(poly(value) == 0 for poly in polynomials):
            count += 1
            if show:
                print(value)
    return count

# Example usage:
q = 5
R.<x,y> = PolynomialRing(GF(q))
f = x^2*y^3 + x*y^2
g = x*y + x^3*y^2
print(count_common_roots([f, g], R, star=True, show=True))
```

For each of the formulas in Chapter 2, we tested the examples by creating functions that implemented the formulas.

```
'''
This function counts the number of roots over GF(q)^n for a fully triangular polynomial
of the form
f(x_1, ..., x_n) = a_1 x_1^(d_{1,1}) + ... + a_n x_1^(d_{1,n}) x_2^(d_{2,n})... x_n^(d_{n,n}),
with coefficients in GF(q), that is *-equivalent to a linear diagonal polynomial of the form
g(x_1, ..., x_n) = a_1 x_1 + ... + a_n x_n.

Inputs:
- q: The number of elements in the field GF(q).
- n: The number of variables in the polynomial.

Output:
- The number of roots over GF(q)^n.
'''
```

```
def roots_linear_b_zero(q, n):
    return (2*q^n + (-1)^n*(q - 1))/(q + 1)

'''
This function counts the number of roots over (GF(q)^*)^n for a fully triangular polynomial
of the form
f(x_1, ..., x_n) = a_1 x_1^(d_{1,1}) + ... + a_n x_1^(d_{1,n}) x_2^(d_{2,n})... x_n^(d_{n,n}),
with coefficients in GF(q), that is *-equivalent to a linear diagonal polynomial of the form
g(x_1, ..., x_n) = a_1 x_1 + ... + a_n x_n.

Inputs:
- q: The number of elements in the field GF(q).
- n: The number of variables in the polynomial.

Output:
- The number of roots over (GF(q)^*)^n.
'''
def roots_linear_star_b_zero(q, n):
    return ((q - 1)^n)/q - ((-1)^n)/q + (-1)^n


# Example usage:
q = 7**2
R.<x_1, x_2, x_3> = PolynomialRing(GF(q))
f = x_1^5 + x_1^3 * x_2^1 + x_1^3 * x_2^3 * x_3^5
print(count_points(f, R))
print(roots_linear_b_zero(q, 3))
print(count_points(f, R, star=True))
print(roots_linear_star_b_zero(q, 3))

'''
This function counts the number of roots over GF(q)^n for a fully triangular polynomial
of the form
f(x_1, ..., x_n) = a_1 x_1^(d_{1,1}) + ... + a_n x_1^(d_{1,n}) x_2^(d_{2,n})... x_n^(d_{n,n}) - b,
with coefficients in GF(q), that is *-equivalent to a linear diagonal polynomial of the form
g(x_1, ..., x_n) = a_1 x_1 + ... + a_n x_n - b.

Inputs:
- q: The number of elements in the field GF(q).
- n: The number of variables in the polynomial.

Output:
- The number of roots over GF(q)^n.
'''
def roots_linear_with_b(q, n):
    total = (q^n - (-1)^n)/(q + 1)
    return total

'''
This function counts the number of roots over (GF(q)^*)^n for a fully triangular polynomial
of the form
f(x_1, ..., x_n) = a_1 x_1^(d_{1,1}) + ... + a_n x_1^(d_{1,n}) x_2^(d_{2,n})... x_n^(d_{n,n}) - b,
with coefficients in GF(q), that is *-equivalent to a linear diagonal polynomial of the form
g(x_1, ..., x_n) = a_1 x_1 + ... + a_n x_n - b.
```

```
Inputs:
- q: The number of elements in the field GF(q).
- n: The number of variables in the polynomial.

Output:
- The number of roots over (GF(q)^*)^n.
'''
def roots_linear_star_with_b(q, n):
    return ((q - 1)^n)/q - ((-1)^n)/q


'''
This function returns the multiplicative character of order 2 over the base field GF(q).

Inputs:
- number: The element of the base field GF(q) to evaluate.
- base_field: The field GF(q) in which the number resides.

Output:
- The value of the multiplicative character at the given number.
'''
def eta(number, base_field):
    generator = base_field.multiplicative_generator()
    exponent = base_field(number).log(generator)

    result = exp(2*pi*I*exponent/2)
    return result


'''
This function returns the constants zeta_1 and zeta_2 over the input field GF(q),
which are used to count solutions when there is *-equivalence to a quadratic diagonal polynomial.

Inputs:
- base_field: The field GF(q) for which the constants are computed.

Output:
- A tuple (zeta_1, zeta_2) representing the constants used in solution counting.
'''
def get_zeta_constants(base_field):
    q = base_field.order()

    zeta_1 = (eta(-1, base_field)*q)^(1/2) - 1
    zeta_2 = -(eta(-1, base_field)*q)^(1/2) - 1
    return zeta_1, zeta_2


'''
The following two functions take the coefficient vector (a_1, ..., a_n) and
return how many of these coefficients are squares or non-squares over the base field GF(q).

Inputs (for both functions):
- coeff_vector: A list of coefficients (a_1, ..., a_n) over GF(q).
```

```
- base_field: The field GF(q) in which the coefficients reside.

Outputs:
- The number of square or non-square elements in the coefficient vector.
'''
def r(coeff_vector, base_field): # how many squares?
    k = len(coeff_vector)
    r = 0
    for i in range(k):
        if 1 == eta(coeff_vector[i], base_field):
            r += 1
    return r

def s(coeff_vector, base_field): # how many non-squares?
    k = len(coeff_vector)
    return k - r(coeff_vector, base_field)


'''
This function counts the number of roots over (GF(q)^*)^n for a fully triangular polynomial
of the form
f(x_1, ..., x_n) = a_1 x_1^(d_{1,1}) + ... + a_n x_1^(d_{1,n}) x_2^(d_{2,n})... x_n^(d_{n,n}),
with coefficients in GF(q), that is *-equivalent to a quadratic diagonal polynomial of the form
g(x_1, ..., x_n) = a_1 x_1^2 + ... + a_n x_n^2.

Inputs:
- coeff_vector: A list of coefficients (a_1, ..., a_n) for the polynomial f.
- R: The polynomial ring R = GF(q)[x_1, ..., x_n] where the polynomial f is defined.

Output:
- The number of roots over (GF(q)^*)^n.
'''
def roots_quadratic_star_b_zero(coeff_vector, R):
    base_field = R.base_ring()
    n = len(coeff_vector)
    rn = r(coeff_vector, base_field)
    q = base_field.order()
    sn = n - rn

    zeta_1, zeta_2 = get_zeta_constants(base_field)
    total = ((q - 1)^(rn+sn))/q + ((q-1)/(2*q)) * (zeta_1^rn*zeta_2^sn + zeta_2^rn*zeta_1^sn)
    return numerical_approx(total)


'''
This function counts the number of roots over GF(q)^n for a fully triangular polynomial
of the form
f(x_1, ..., x_n) = a_1 x_1^(d_{1,1}) + ... + a_n x_1^(d_{1,n}) x_2^(d_{2,n})... x_n^(d_{n,n}),
with coefficients in GF(q), that is *-equivalent to a quadratic diagonal polynomial of the form
g(x_1, ..., x_n) = a_1 x_1^2 + ... + a_n x_n^2.

Inputs:
- coeff_vector: A list of coefficients (a_1, ..., a_n) for the polynomial f.
```

```
- R: The polynomial ring R = GF(q)[x_1, ..., x_n] where the polynomial f is defined.

Output:
- The number of roots over GF(q)^n.
'''
def roots_quadratic_b_zero(coeff_vector, R):
    base_field = R.base_ring()

    q = base_field.order()
    n = len(coeff_vector)

    zeta_1, zeta_2 = get_zeta_constants(base_field)
    total = q**(n-1) + roots_quadratic_star_b_zero(
            coeff_vector, R
    )

    for k in range(1, n):
        N_k = roots_quadratic_star_b_zero(
            coeff_vector[:k], R
        )
        total += N_k * (q**(n - k - 1))
    return numerical_approx(total)


# Example usage:
print(roots_quadratic_star_b_zero((1,R(2)), R))
print(count_points(f, R, star=True))
print(roots_quadratic_b_zero((1, R(2)), R))
print(count_points(f, R))

'''
This function takes as input q, a list of coefficients
(a_1, ..., a_n), and a constant b of a fully triangular
polynomial of the form
f(x_1, ..., x_n) = a_1 x_1^(d_{1,1}) + ... + a_n x_1^(d_{1,n}) x_2^(d_{2,n})... x_n^(d_{n,n}) - b,
with coefficients in GF(q), that is *-equivalent to a
quadratic diagonal polynomial of the form
g(x_1, ..., x_n) = a_1 x_1^2 + ... + a_n x_n^2 - b.
It returns the number of roots over (GF(q)^*)^n of f.

Inputs:
- coeff_vector: A list of coefficients (a_1, ..., a_n) for the polynomial f.
- b: The constant term subtracted from the polynomial.
- R: The polynomial ring R = GF(q)[x_1, ..., x_n] where the polynomial f is defined.

Output:
- The number of roots over (GF(q)^*)^n.
'''
def roots_quadratic_star_with_b(coeff_vector, b, R):
    base_field = R.base_ring()
    n = len(coeff_vector)

    rn = r(coeff_vector, base_field)
    q = base_field.order()
```

```
    sn = n - rn

    zeta_1, zeta_2 = get_zeta_constants(base_field)
    total = ((q - 1) ^ n) / q
    total += -(1 / (2 * q)) * (
        zeta_1 ^ rn * zeta_2 ^ sn + zeta_2 ^ rn * zeta_1 ^ sn
    )
    total += (
        eta(b, base_field) /
        (2 * (q * eta(R(-1), base_field)) ^ (1 / 2))
    ) * (
        zeta_1 ^ rn * zeta_2 ^ sn - zeta_2 ^ rn * zeta_1 ^ sn
    )

    return numerical_approx(total)


'''
This function takes as input q, a list of coefficients
(a_1, ..., a_n), and a constant b of a fully triangular
polynomial of the form
f(x_1, ..., x_n) = a_1 x_1^(d_{1,1}) + ... + a_n x_1^(d_{1,n}) x_2^(d_{2,n})... x_n^(d_{n,n}) - b,
with coefficients in GF(q), that is *-equivalent to a
quadratic diagonal polynomial of the form
g(x_1, ..., x_n) = a_1 x_1^2 + ... + a_n x_n^2 - b.
It returns the number of roots over GF(q)^n of f.

Inputs:
- coeff_vector: A list of coefficients (a_1, ..., a_n) for the polynomial f.
- b: The constant term subtracted from the polynomial.
- R: The polynomial ring R = GF(q)[x_1, ..., x_n] where the polynomial f is defined.

Output:
- The number of roots over GF(q)^n.
'''
def roots_quadratic_with_b(coeff_vector, b, R):
    base_field = R.base_ring()

    q = base_field.order()
    n = len(coeff_vector)

    zeta_1, zeta_2 = get_zeta_constants(base_field)
    total = roots_quadratic_star_with_b(
        coeff_vector, b, R
    )

    for k in range(1, n):
        N_k = roots_quadratic_star_with_b(
            coeff_vector[:k], b, R
        )
        total += N_k * (q ** (n - k - 1))

    return numerical_approx(total)
```

```
# Example usage:
f = x_1^2 + x_1^4 * x_2^2 + x_1^4 * x_2^2 * x_3^2 - R(1)
print(roots_quadratic_star_with_b((1, 1, 1), 1, R))
print(count_points(f, R, star=True))
print(roots_quadratic_with_b((1, 1, 1), 1, R))
print(count_points(f, R))
```

In Chapter 3, we presented a formula with three cases for the number of solutions of full polynomials that satisfy certain conditions. We implemented it as a single function.

```
'''
Computes the number of roots of a full polynomial of the form
f(x_1, ..., x_n) =  a_1 x_1^{d_{1,1}}x_2^{d_{2,1}}... x_n^{d_{n, 1}}
                    + ... + a_n x_1^{d_{1,n}}x_2^{d_{2,1}}... x_n^{d_{n, n}}  - b,
which is *-equivalent to a diagonal polynomial with s variables of the form:
g(x_1, ..., x_n) = a_1 x_1^d + ... + a_s x_s^d - b,
where n >= s, and d >= 3 is (p, r)-admissible for some positive integer r such that 2r | m,
and eta_{d}(a_1) = cdots = eta_{d}(a_s).

Inputs:
- a_1: Coefficient in the polynomial.
- b: Constant term in the polynomial.
- base_field: The finite field GF(q) where the coefficients are defined.
- n: Number of variables in the polynomial.
- s: Number of variables in the diagonal polynomial.
- d: Parameter related to the degree of the polynomial.

Outputs:
- The number of roots over GF(q)^n of the polynomial.
'''
def roots_full_poly(a_1, b, base_field, n, s, d):
    q = base_field.cardinality()
    p = base_field.characteristic()
    m = base_field.degree()

    r = 1
    while not (p**r + 1) % d == 0:
        r += 1
    h = m / (2 * r)

    A_d = -1 - (-1)**h * (d - 1) * q**0.5
    B_d = -1 + (-1)**h * q**0.5

    def eta_d(number):
        generator = base_field.multiplicative_generator()
        exponent = base_field(number).log(generator)
        return exp(2*pi*I*exponent/d)

    term_1 = q**n - (q - 1)**n
    term_2 = (q - 1)**(n - s) / q
    if b == 0:
```

```
        return term_1 + term_2 * (
            (q - 1)**s + ((q - 1) / d) * A_d**s
            + ((q - 1) * (d - 1) / d) * B_d**s
        )

    eta_value = eta_d(a_1/b)
    if eta_value == 1:
        return term_2 * (
            (q - 1)**s - B_d**s
            + (A_d * (A_d**s - B_d**s)) / d
        )
    return term_2 * (
        (q - 1)**s - B_d**s
        + (B_d * (A_d**s - B_d**s)) / d
    )


# Example usage:
q = 16
print(roots_full_poly(1, 0, GF(q), 3, 2, 5))
```

Lastly, in Chapter 4 we did not use many computational resources to compute the examples. The examples that required it were obtained by using default SageMath commands for cyclic codes and manually examining the generator matrices and parity check matrices. When possible we also checked the minimum distance; however the default SageMath command takes forever to evaluate when the parameters are large. Here we present some sample commands.

```
Cc = codes.CyclicCode(length=15, field=GF(2), D=[1,7])
print(Cc.parity_check_matrix())
print("") # to make sure the matrices do not stack on the output
print(Cc.generator_matrix())

# the following line takes some time to evaluate
# and becomes unusable for codes where the parameters
# q or n are large
print(Cc.minimum_distance())

# this length is enough to make my personal computer cry
Cc = codes.CyclicCode(length=63, field=GF(2), D=[1,7])
print(Cc.minimum_distance())
```