UNIVERSIDADE FEDERAL DO MARANHÃO
Fundação Instituída nos termos da Lei 5.152 de 21/10/1966 - São Luís - MA
Centro de Ciências Exatas e Tecnologia
Curso de Matemática – Bacharelado

Isabelle Cristina Reis Castro

Códigos Lineares: A Métrica de Hamming na Correção de erros

São Luís - MA
2025

Isabelle Cristina Reis Castro

Códigos Lineares: A Métrica de Hamming na Correção de erros

Monografia (Trabalho de Conclusão de Curso) apresentada à Coordenadoria dos cursos de Matemática, da Universidade Federal do Maranhão, como requisito parcial para obtenção do grau de Bacharela em Matemática.

Curso de Matemática – Bacharelado

Universidade Federal do Maranhão

Orientador: Prof. Dr. Gustavo Silvestre do Amaral Costa

São Luís - MA

2025

Ficha gerada por meio do SIGAA/Biblioteca com dados fornecidos pelo(a) autor(a).
Diretoria Integrada de Bibliotecas/UFMA

Reis Castro, Isabelle Cristina.
Códigos Lineares : a métrica de Hamming na Correção de Erros / Isabelle Cristina Reis Castro. - 2025.
66 p.

Orientador(a): Gustavo Silvestre do Amaral Costa.
Curso de Matemática, Universidade Federal do Maranhão, São Luís, 2025.

1. Códigos Lineares. 2. Distância de Hamming. 3. Correção de Erros. I. do Amaral Costa, Gustavo Silvestre. II. Título.

Isabelle Cristina Reis Castro

Códigos Lineares: A Métrica de Hamming na Correção de erros

Monografia (Trabalho de Conclusão de Curso) apresentada à Coordenadoria dos cursos de Matemática, da Universidade Federal do Maranhão, como requisito parcial para obtenção do grau de Bacharela em Matemática.

Trabalho **APROVADO**. São Luís - MA, 10/03/2025

Prof. Dr. Gustavo Silvestre do Amaral Costa
DEMAT/UFMA
Orientador

Prof. Dr. Ermerson Rocha Araujo
DEMAT/UFMA
Primeiro Examinador

Prof. Dr. Jadevilson Cruz Ribeiro
BICT/UFMA
Segundo Examinador

À minha mãe.

Agradecimentos

Agradeço ao meu orientador, Professor Gustavo Silvestre do Amaral Costa, pela paciência, pelo apoio e pela confiança depositada em mim desde o início da nossa colaboração. Sou também profundamente grata ao Professor Cléber Cavalcante, cuja paciência e dedicação foram essenciais ao longo dos últimos anos. À minha mãe, minha base e maior inspiração, agradeço por todo o apoio, amor e incentivo ao longo da minha vida. E, por fim, agradeço a mim mesma por cada esforço e por nunca ter desistido, mesmo diante dos desafios.

"Cada fracasso nos ensina algo que necessitávamos aprender."

Charles Dickens (1812 – 1870)

Resumo

A necessidade de garantir a integridade dos dados durante a transmissão e o armazenamento é uma preocupação crítica em sistemas modernos de comunicação e computação. Com o aumento exponencial na quantidade de dados gerados e transmitidos diariamente, o impacto dos erros na qualidade e na precisão das informações nunca foi tão significativo. Erros de transmissão podem ocorrer devido a diversos fatores, como ruído, interferência e degradação do meio de comunicação, e podem comprometer a funcionalidade e a confiabilidade dos sistemas.

Neste cenário, as técnicas de detecção e correção de erros desempenham um papel crucial na manutenção da qualidade dos dados.

A métrica de Hamming, como uma ferramenta fundamental na teoria dos códigos lineares, oferece uma abordagem eficaz para identificar e corrigir erros. O uso dessa métrica permite a construção de códigos que não apenas detectam a presença de erros, mas também corrigem esses erros sem a necessidade de retransmissão, o que é especialmente valioso em ambientes com alta taxa de erros e onde a retransmissão de dados é impraticável ou indesejável.

Este trabalho explora a métrica de Hamming, que mede a distância entre dois códigos como o número de posições em que eles diferem. Através dessa métrica, é possível identificar e corrigir erros que ocorrem durante a transmissão de dados em sistemas de comunicação. O trabalho analisa a estrutura dos códigos lineares e demonstra como a distância de Hamming entre palavras-código pode ser utilizada para projetar esquemas de correção de erros eficientes.

Palavras-chave: Códigos Lineares, Distância de Hamming, Correção de Erros.

Abstract

The need to ensure the integrity of data during transmission and storage is a critical concern in modern communication and computing systems. With the exponential increase in the amount of data generated and transmitted daily, the impact of errors on the quality and accuracy of information has never been more significant. Transmission errors can occur due to various factors, such as noise, interference, and degradation of the communication medium, and can compromise the functionality and reliability of the systems.

In this scenario, error detection and correction techniques play a crucial role in maintaining data quality.

Hamming metrics, as a fundamental tool in linear code theory, offer an effective approach to identifying and correcting errors. Using this metric allows you to build code that not only detects the presence of errors, but also corrects those errors without the need for retransmission, which is especially valuable in environments with a high error rate and where data relay is impractical or undesirable.

This work explores the Hamming metric, which measures the distance between two codes as the number of positions in which they differ. Through this metric, it is possible to identify and correct errors that occur during the transmission of data in communication systems. The work analyzes the structure of linear codes and demonstrates how the Hamming distance between codewords can be used to design efficient error correction schemes.

Keywords: linear codes, hamming distance, correction of errors

Lista de ilustrações

Lista de tabelas

Lista de símbolos

n	*comprimento das palavras de um codigo linear, 14*
w_H	*quantidade de bits não nulos em uma palavra, 15*
$\mathbb{F}$	*um corpo, geralmente finito, 16*
$\mathbb{F}_q$	*um corpo com q elementos, 16*
A	*um alfabeto, 16*
A^n	*conjunto com todas as palavras de comprimento n compostas por A elementos, 16*
$d_H(x, y)$	*distância de Hamming entre x e y, 18*
C	*um codigo linear, 22*
$d_{\min}$, d	*distância minima de um codigo linear, 22*
κ	*capacidade do código, 23*
$\lfloor x \rfloor$	*o maior inteiro menor ou igual que x, 23*
(n, k)	*parâmetros de um código linear com palavras de comprimento n e dimensão k, 24*
k	*dimensão de um codigo linear, 24*
M	*quantidade de palavras de um código linear, 25*
G, Γ	*matriz geradora de um código C, 25*
H, Δ	*matriz de verificação de um codigo C, 28*
$\Delta^\top$	*a transposta de Δ, 29*
e	*vetor erro, 29*
$S(x, r)$	*esfera de raio r centrada em x, 63*
$B(x, r)$	*bola de raio r centrada em x, 63*
$A_q(n, d)$	*quantidade de palavras de um código ótimo, 66*

Sumário

Introdução

> Códigos de correção de erros são usados para melhorar a confiabilidade de sistemas de comunicação. Podemos pensar em comunicação a grandes distâncias, como na comunicação com espaçonaves, ou basicamente em qualquer forma de transmissão de informações, incluindo a reprodução de uma peça musical que foi gravada anteriormente e armazenada em algum tipo de mídia, por exemplo. (...) O objetivo da Teoria da Codificação é fornecer métodos que melhorem a confiabilidade da comunicação através de um canal ruidoso. (BETTEN MICHAEL BRAUN; WASSERMANN, 2006).

Ou seja, em qualquer caso, o objetivo dos códigos corretores de erros é transmitir e reproduzir as informações com a maior precisão possível, mesmo sob circunstâncias desfavoráveis, como em um ambiente propenso a erros.

A métrica de Hamming, introduzida por Richard Hamming na década de 1950, é uma medida que quantifica a diferença entre duas cadeias de símbolos. Especificamente, ela conta o número de posições nas quais dois códigos diferem. Essa métrica é fundamental para a construção e a análise de códigos que podem detectar e corrigir erros. Em sistemas de comunicação, onde a transmissão de dados é suscetível a interferências e ruídos, a capacidade de detectar e corrigir erros é vital para garantir a precisão das informações recebidas.

Os códigos de Hamming são um tipo de código de correção de erros que utiliza a métrica de Hamming para identificar e corrigir erros. Esses códigos são projetados para maximizar a distância mínima entre os códigos válidos, o que permite a correção de um número determinado de erros. A aplicação prática desses códigos inclui diversas áreas, como redes de computadores, transmissão de dados sem fio, e armazenamento de dados digitais.

Ao longo deste trabalho, discutiremos os fundamentos básicos dos códigos lineares, e definiremos a métrica de Hamming. Também apresentaremos a estrutura dos códigos lineares, sua codificação e decodificação, bem como, suas relações algébricas no espaço vetorial. Por último, discutiremos seus limites de eficiência e desempenho.

1 Desenvolvimentos Preliminares

1.1 Definições

Primeiro, iremos definir alguns conceitos básicos que são importantes para a compreensão dos códigos corretores de erros. Esses conceitos incluem os de alfabeto, palavra, comprimento e peso de palavras, imprescindíveis para o entendimento de códigos lineares.

1. **Alfabeto** (A)

Um **alfabeto** A é um conjunto finito e não vazio de símbolos, que serve como base para a construção de palavras. Cada símbolo do alfabeto representa uma unidade básica de informação. O tamanho do alfabeto, denotado por $|A|$, é o número de elementos distintos no conjunto. Também podemos representar $|A| = \mathrm{q}$.

- $A = \{0, 1\}$, conhecido como o alfabeto binário, é amplamente utilizado em computação e telecomunicações, onde $|A| = 2$.
- $A = \{a, b, c, \ldots, z\}$, o alfabeto latino, é usado para representação de texto em linguagens naturais, onde $|A| = 26$.
- $A = \{0, 1, 2, 3\}$, um alfabeto com quatro símbolos, utilizado em sistemas numéricos quaternários, onde $|A| = 4$.

2. **Palavra-código**

Uma **palavra-código** sobre um alfabeto A é definida como uma sequência finita de símbolos pertencentes ao alfabeto. Mais formalmente, se A é um alfabeto, uma palavra-código w de comprimento n é uma função:

$$w : \{1, 2, \ldots, n\} \to A,$$

que associa a cada posição $i \in \{1, 2, \ldots, n\}$ um símbolo $w_i \in A$. As palavras-código são frequentemente representadas pela notação $w = (w_1, w_2, \ldots, w_n)$, e podem também ser chamadas apenas de **palavra**.

- Se $A = \{0, 1\}$, então $w = (0, 1, 1, 0)$ é uma palavra de A.

- Se $A = \{a, b, c\}$, uma palavra válida de A pode ser $w = (a, c, b)$.
- Para $A = \{0, 1, 2\}$, $w = (2, 0, 1, 2)$ é um exemplo de palavra de A.

3. **Comprimento de uma Palavra-código** (n)

O **comprimento** n de uma palavra-código w, também denotado por $|w|$, é definido como o número de símbolos que a palavra-código contém. O comprimento é uma medida importante, especialmente no contexto de códigos corretores de erros, pois determina o tamanho das mensagens ou blocos de dados.

- Para $w = (0, 1, 1, 0)$, onde $A = \{0, 1\}$, o comprimento é $n = 4$.
- Se $w = (a, b, c)$ e $A = \{a, b, c, d\}$, temos $n = 3$.
- Uma palavra-código vazia w possui comprimento $n = 0$.

4. **Peso de uma Palavra-código** (w_H)

Seja $\mathbf{x} = (x_1, x_2, \ldots, x_n) \in \mathbb{F}_q^n$ uma palavra de comprimento n sobre o corpo $\mathbb{F}_q$. O **peso de Hamming** de $\mathbf{x}$, denotado por $w_H(\mathbf{x})$, é definido como o número de coordenadas não nulas de $\mathbf{x}$. Formalmente:

$$w_H(\mathbf{x}) = |\{i \mid x_i \neq 0,\ 1 \leq i \leq n\}|.$$

Exemplo 1.1. *Considere* $\mathbf{x} = (1, 0, 0, 1, 0, 1) \in \mathbb{F}_2^6$. *O peso de Hamming de* $\mathbf{x}$ *é dado pelo número de posições onde os elementos de* $\mathbf{x}$ *são diferentes de zero. Aqui, temos:*

$$\mathbf{x}_1 = 1,\ \mathbf{x}_4 = 1,\ \mathbf{x}_6 = 1,$$

o que implica que:

$$w_H(\mathbf{x}) = 3.$$

Exemplo elaborado pela autora.

1.2 A^n: Produto Cartesiano do Alfabeto A

Dado um alfabeto A, definido como um conjunto finito e não vazio de símbolos, A^n é o conjunto de todas as sequências ordenadas de comprimento n, cujos elementos pertencem a A. Formalmente, temos:

$$A^n = \{(a_1, a_2, \ldots, a_n) \mid a_i \in A,\ \forall i \in \{1, 2, \ldots, n\}\}.$$

Propriedades de A^n

- Cada elemento de A^n é chamado de uma **palavra** de comprimento n sobre o alfabeto A.
- O tamanho de A^n é dado por $|A|^n = q^n$, onde $|A|$ é o número de elementos no alfabeto A. Ou seja, existem q^n palavras em A^n.

Exemplo 1.2. *Palavras em A^n*

- *Se $A = \{0, 1\}$, então $A^3 = \{(0, 0, 0), (0, 0, 1), \ldots, (1, 1, 1)\}$, contendo $2^3 = 8$ palavras.*
- *Para $A = \{a, b\}$, $A^2 = \{(a, a), (a, b), (b, a), (b, b)\}$, com $2^2 = 4$ palavras.*

Exemplo elaborado pela autora.

1.3 $\mathbb{F}_q^n$: Espaço Vetorial sobre um Corpo Finito

O conjunto $\mathbb{F}_q^n$ representa um espaço vetorial de dimensão n sobre $\mathbb{F}_q$, onde $\mathbb{F}$ é um corpo finito e q é a sua quantidade de elementos. Aqui, $\mathbb{F}_q$ é o conjunto de elementos do corpo finito com q elementos, equipado com operações de soma e multiplicação que satisfazem as propriedades de um corpo. Temos:

$$\mathbb{F}_q^n = \{(v_1, v_2, \ldots, v_n) \mid v_i \in \mathbb{F}_q, \forall i \in \{1, 2, \ldots, n\}\}.$$

Propriedades de $\mathbb{F}_q^n$

- $\mathbb{F}_q^n$ é um espaço vetorial, satisfazendo:
  - A soma de dois vetores em $\mathbb{F}_q^n$ resulta em outro vetor em $\mathbb{F}_q^n$.
  - A multiplicação escalar de um vetor em $\mathbb{F}_q^n$ por um elemento de $\mathbb{F}_q$ resulta em outro vetor em $\mathbb{F}_q^n$.
- O número total de elementos em $\mathbb{F}_q^n$ é q^n, pois há q escolhas possíveis para cada coordenada.
  - Seja $\mathbb{F}_q$ um corpo finito com q elementos. O espaço $\mathbb{F}_q^n$ é o conjunto de todas as n-uplas $(v_1, v_2, \ldots, v_n)$, onde cada $v_i \in \mathbb{F}_q$. Para cada coordenada v_i, existem q escolhas possíveis, pois $\mathbb{F}_q$ contém exatamente q elementos.

    Assim, o número total de n-uplas $(v_1, v_2, \ldots, v_n)$ é dado pelo produto das q possibilidades para cada uma das n coordenadas. Portanto:

$$|\mathbb{F}_q^n| = q \cdot q \cdot \cdots \cdot q = q^n.$$

Exemplo 1.3. *Seja* $\mathbb{F}_2 = \{0, 1\}$. *O espaço* $\mathbb{F}_2^3$ *consiste em todas as* 3*-uplas de* $\mathbb{F}_2$*:*

$$\mathbb{F}_2^3 = \{(0,0,0), (0,0,1), (0,1,0), (0,1,1), (1,0,0), (1,0,1), (1,1,0), (1,1,1)\}.$$

Neste caso, $|\mathbb{F}_2^3| = 2^3 = 8$*, o que confirma o resultado geral.*

- Para $q = 2$ e $n = 3$, temos $\mathbb{F}_2 = \{0, 1\}$ e $\mathbb{F}_2^3 = \{(0,0,0), (0,0,1), (0,1,0), \ldots, (1,1,1)\}$, com $2^3 = 8$ vetores.
- Para $q = 3$ e $n = 2$, $\mathbb{F}_3 = \{0, 1, 2\}$ e $\mathbb{F}_3^2$ contém $3^2 = 9$ vetores, como: $(0,0), (0,1), (0,2)$, $\ldots, (2,2)$.

Exemplo elaborado pela autora.

1.3.1 Operações em $\mathbb{F}_q^n$

No espaço vetorial $\mathbb{F}_q^n$, definem-se a **soma de vetores** e a **multiplicação de um vetor por um escalar**, ambas realizadas coordenada a coordenada. Essas operações seguem as propriedades do corpo finito $\mathbb{F}_q$, o que assegura que $\mathbb{F}_q^n$ seja um espaço vetorial.

A tabela a seguir apresenta as definições dessas operações:

Operação	**Definição Matemática**
Soma de vetores	$\mathbf{v} + \mathbf{w} = (v_1 + w_1, v_2 + w_2, \ldots, v_n + w_n)$
Multiplicação por escalar	$c \cdot \mathbf{v} = (c \cdot v_1, c \cdot v_2, \ldots, c \cdot v_n)$

Tabela 1.1 – Operações no espaço vetorial $\mathbb{F}_q^n$.

Explicação das Operações

1. **Soma de Vetores:** A soma de dois vetores $\mathbf{v}, \mathbf{w} \in \mathbb{F}_q^n$ é realizada somando-se as coordenadas correspondentes módulo q. Formalmente:

$$\mathbf{v} + \mathbf{w} = (v_1 + w_1, v_2 + w_2, \ldots, v_n + w_n),$$

onde $+$ representa a operação de adição em $\mathbb{F}_q$.

2. **Multiplicação por Escalar:** A multiplicação de um vetor $\mathbf{v} \in \mathbb{F}_q^n$ por um escalar $c \in \mathbb{F}_q$ é realizada multiplicando-se cada coordenada do vetor por c módulo q. Formalmente:

$$c \cdot \mathbf{v} = (c \cdot v_1, c \cdot v_2, \ldots, c \cdot v_n),$$

onde $\cdot$ representa a operação de multiplicação em $\mathbb{F}_q$.

Essas operações são aplicadas na definição de códigos lineares, onde $\mathbb{F}_q^n$ é utilizado para representar palavras-código e mensagens.

Usaremos como operações sobre o corpo $\mathbb{F}_2^n$,a soma e a multiplicação módulo 2, descritas a seguir:

Operação	**Entrada**	**Resultado**
Soma	$1+1$	0
Soma	$1+0$	1
Soma	$0+1$	1
Soma	$0+0$	0
Multiplicação	1×1	1
Multiplicação	1×0	0
Multiplicação	0×1	0
Multiplicação	0×0	0

Tabela 1.2 – Operações de soma e multiplicação módulo 2

1.3.2 Código Linear

Um **código linear** é um conjunto de palavras-código cujos elementos seguem uma mesma estrutura algébrica. Para uma definição formal, consulte a **Definição 2.1 (Código linear)** na página 35.

1.4 Distância de Hamming entre Duas Palavras

Teorema 1.4. *(Métrica de Hamming) ((BETTEN MICHAEL BRAUN; WASSERMANN, 2006), p. 13).*

Seja $\mathbb{F}_q^n$ o espaço vetorial de dimensão n sobre o corpo finito $\mathbb{F}_q$. Defina a função:

$$d_H : \mathbb{F}_q^n \times \mathbb{F}_q^n \to \mathbb{N}, \quad (u,v) \mapsto |\{i \mid i \in \{1,\dots,n\}, u_i \neq v_i\}|,$$

onde $u = (u_1,\dots,u_n)$ e $v = (v_1,\dots,v_n)$. A função d_H é chamada de ***métrica de Hamming*** *e satisfaz as seguintes propriedades:*

1. *Não negatividade e identidade dos indiscerníveis:*

$$d_H(u,v) = 0 \iff u = v.$$

2. *Simetria:*

$$d_H(u,v) = d_H(v,u).$$

3. *Desigualdade triangular:*

$$d_H(u,v) \leq d_H(u,w) + d_H(w,v), \quad \text{para todos } u,v,w \in \mathbb{F}_q^n.$$

Além disso, a distância de Hamming possui as seguintes propriedades de invariância:

1. *Invariância por translação:*

$$d_H(u,v) = d_H(u+w, v+w), \quad \text{para todos } u,v,w \in \mathbb{F}_q^n.$$

2. Invariância por multiplicação escalar:

$$d_H(u,v) = d_H(\lambda u, \lambda v), \quad \textit{para todo } \lambda \in \mathbb{F}_q, \lambda \neq 0.$$

Para duas palavras $x = (x_1, x_2, \ldots, x_n)$ e $y = (y_1, y_2, \ldots, y_n)$:

$$d_H(x,y) = \sum_{i=1}^{n} \delta(x_i, y_i), \tag{1.1}$$

onde:

$$\delta(x_i, y_i) = \begin{cases} 0, & \text{se } x_i = y_i; \\ 1, & \text{se } x_i \neq y_i. \end{cases} \tag{1.2}$$

Propriedades da Distância de Hamming

- $d_H(u,v) = 0 \iff u = v$ (Não negatividade e identidade dos indiscerníveis)

A primeira propriedade, $d_H(u,v) = 0$ se e somente se u = v, afirma que a distância entre dois pontos u e v é zero se, e somente se, esses pontos forem iguais.

Demonstração: Sejam $u = (u_1, u_2, \ldots, u_n)$ e $v = (v_1, v_2, \ldots, v_n)$ duas palavras em $\mathbb{F}_q^n$. A distância de Hamming é definida como:

$$d_H(u,v) = \sum_{i=1}^{n} \delta(u_i, v_i),$$

onde:

$$\delta(u_i, v_i) = \begin{cases} 0, & \text{se } u_i = v_i, \\ 1, & \text{se } u_i \neq v_i. \end{cases}$$

Vamos provar que $d_H(u,v) = 0 \iff u = v$.

Prova da ida ($d_H(u,v) = 0 \implies u = v$)

Suponha que $d_H(u,v) = 0$. Pela definição da distância de Hamming:

$$d_H(u,v) = \sum_{i=1}^{n} \delta(u_i, v_i).$$

Como $d_H(u,v) = 0$, segue que cada termo $\delta(u_i, v_i) = 0$ para todo $i \in \{1, 2, \ldots, n\}$.

De acordo com a definição de $\delta(u_i, v_i)$, isso implica que:

$$u_i = v_i, \quad \text{para todo } i \in \{1, 2, \ldots, n\}.$$

Portanto, $u = v$.

Prova da volta ($u = v \implies d_H(u, v) = 0$)

Suponha que $u = v$. Isso significa que, para todo $i \in \{1, 2, \ldots, n\}$, temos $u_i = v_i$. Pela definição de $\delta(u_i, v_i)$, segue que:

$$\delta(u_i, v_i) = 0, \quad \text{para todo } i \in \{1, 2, \ldots, n\}.$$

Assim, a soma que define $d_H(u, v)$ é:

$$d_H(u, v) = \sum_{i=1}^{n} \delta(u_i, v_i) = \sum_{i=1}^{n} 0 = 0.$$

Concluímos que:

$$d_H(u, v) = 0 \iff u = v.$$

■

- $d_H(u, v) = d_H(v, u)$ (simetria da distância)

A segunda propriedade, $d_H(u, v) = d_H(v, u)$, destaca que a distância entre dois pontos u e v em um código linear é a mesma, independentemente da ordem em que esses pontos são comparados. Em outras palavras, a distância entre v e ué igual à distância entre u e v, refletindo a simetria da relação de distância.

Demonstração: Sejam $x = (x_1, x_2, \ldots, x_n)$ e $y = (y_1, y_2, \ldots, y_n)$ duas palavras em $\mathbb{F}_q^n$.

Pela definição de $d_H(x, y)$, temos:

$$d_H(x, y) = \sum_{i=1}^{n} \delta(x_i, y_i).$$

Sabemos que a função $\delta(x_i, y_i)$ verifica a propriedade:

$$\delta(x_i, y_i) = \delta(y_i, x_i).$$

Isso ocorre porque a definição de δ depende apenas da comparação $x_i \neq y_i$, que é simétrica: - Se $x_i = y_i$, então $\delta(x_i, y_i) = \delta(y_i, x_i) = 0$. - Se $x_i \neq y_i$, então $\delta(x_i, y_i) = \delta(y_i, x_i) = 1$.

Substituindo na soma:

$$d_H(x, y) = \sum_{i=1}^{n} \delta(x_i, y_i) = \sum_{i=1}^{n} \delta(y_i, x_i) = d_H(y, x).$$

Concluímos que:

$$d_H(x, y) = d_H(y, x),$$

e, portanto, a distância de Hamming é simétrica.

■

- $d_H(x, z) \leq d_H(x, y) + d_H(y, z)$ (desigualdade triangular).

A terceira propriedade, $d_H(x, z) \leq d_H(x, y) + d_H(y, z)$, é conhecida como desigualdade triangular e estabelece que a distância entre dois pontos x e z em um código linear é sempre menor ou igual à soma das distâncias entre x e y, e entre y e z, para qualquer ponto y no código. Essa propriedade reflete a relação triangular entre os pontos em um espaço métrico.

Demonstração: Sejam $u = (u_1, u_2, \ldots, u_n)$, $v = (v_1, v_2, \ldots, v_n)$, e $w = (w_1, w_2, \ldots, w_n)$ palavras em $\mathbb{F}_q^n$. Para cada índice i, definimos os seguintes conjuntos:

- $A = \{i \mid u_i \neq w_i\}$:
  O conjunto de índices onde u e w diferem. O tamanho de A é a distância de Hamming entre u e w, ou seja, $|A| = d_H(u, w)$.
- $B = \{i \mid u_i \neq v_i\}$:
  O conjunto de índices onde u e v diferem. O tamanho de B é a distância de Hamming entre u e v, ou seja, $|B| = d_H(u, v)$.
- $C = \{i \mid v_i \neq w_i\}$:
  O conjunto de índices onde v e w diferem. O tamanho de C é a distância de Hamming entre v e w, ou seja, $|C| = d_H(v, w)$.

Queremos mostrar que $A \subseteq B \cup C$, ou seja, qualquer índice i em A pertence à união dos conjuntos B e C.

Seja $i \in A$, ou seja, $u_i \neq w_i$. Precisamos mostrar que $i \in B$ ou $i \in C$.

Analisamos os dois possíveis casos:

1. Se $i \notin B$, ou seja, $u_i = v_i$:
   - Como $i \in A$, temos $u_i \neq w_i$.
   - Mas $u_i = v_i$, então substituindo u_i por v_i, obtemos $v_i \neq w_i$.
   - Logo, $i \in C$.
2. Se $i \notin C$, ou seja, $v_i = w_i$:

– Como $i \in A$, temos $u_i \neq w_i$.
– Mas $v_i = w_i$, então substituindo w_i por v_i, obtemos $u_i \neq v_i$.
– Logo, $i \in B$.

Em ambos os casos, $i \in B \cup C$, o que mostra que $A \subseteq B \cup C$.

Podemos concluir que o conjunto A = {i: $u_i \neq v_i$} é uma subcoleção da união dos conjuntos B e C.
Podemos escrever: $A \subseteq B \cup C$, portanto $|A| \leq |B \cup C|$, e isso nos dá a desigualdade $d_H(u, v) \leq d_H(u, w) + d_H(w, v)$.

■

Exemplo 1.5. *Considere as palavras* $x = (1, 0, 1, 1)$ *e* $y = (1, 1, 0, 1)$*, onde* $A = \{0, 1\}$*. A distância de Hamming é calculada como:*

$$d_H(x, y) = \delta(1, 1) + \delta(0, 1) + \delta(1, 0) + \delta(1, 1) = 0 + 1 + 1 + 0 = 2. \tag{1.3}$$

Isso indica que as palavras diferem em duas posições.

Outro exemplo: para $x = (a, b, c, d)$ *e* $y = (a, c, c, d)$*, com* $A = \{a, b, c, d\}$*, temos:*

$$d_H(x, y) = \delta(a, a) + \delta(b, c) + \delta(c, c) + \delta(d, d) = 0 + 1 + 0 + 0 = 1. \tag{1.4}$$

Exemplo elaborado pela autora.

Observação 1.6. *Seja* $q = 2$*, a distância entre dois vetores código* u *e* v *de um código linear é igual ao peso de seu vetor soma* $u + v$*, isto é,* $d(u, v) = w_H(u + v)$*. (NOLLI, 1985)*

1.5 Distância Mínima de Hamming entre Códigos

A **distância mínima de Hamming** de um código $\mathcal{C} \subseteq \mathbb{F}_q^n$, denotada por $d_{\min}$, é definida como a menor distância de Hamming entre quaisquer duas palavras-código distintas de $\mathcal{C}$. Mais formalmente:

$$d_{\min} = \min\{d_H(x, y) \mid x, y \in \mathcal{C}, x \neq y\},$$

onde $d_H(x, y)$ representa a distância de Hamming entre as palavras x e y.

1. **Código com $\mathbb{F}_2^3$:** Considere o código $\mathcal{C} = \{(0, 0, 0), (1, 1, 1)\} \subseteq \mathbb{F}_2^3$. As distâncias de Hamming entre as palavras-código são:

$$d_H((0, 0, 0), (1, 1, 1)) = 3.$$

2. **Código com $\mathbb{F}_3^4$:** Considere o código $\mathcal{C} = \{(0, 0, 0, 0), (1, 1, 1, 1), (2, 2, 2, 2)\} \subseteq \mathbb{F}_3^4$. As distâncias de Hamming entre as palavras-código são:

$$d_H((0,0,0,0),(1,1,1,1)) = 4,$$
$$d_H((0,0,0,0),(2,2,2,2)) = 4,$$
$$d_H((1,1,1,1),(2,2,2,2)) = 4.$$

Portanto, $d_{\min} = 4$.

3. **Código Linear $\mathcal{C} \subseteq \mathbb{F}_2^4$:** Considere o código linear gerado por $\mathcal{C} = \langle (1,0,0,1), (0,1,1,0) \rangle$.
As palavras-código são:

$$\mathcal{C} = \{(0,0,0,0),(1,0,0,1),(0,1,1,0),(1,1,1,1)\}.$$

As distâncias de Hamming entre as palavras-código distintas são:

$$d_H((0,0,0,0),(1,0,0,1)) = 2,$$
$$d_H((0,0,0,0),(0,1,1,0)) = 2,$$
$$d_H((0,0,0,0),(1,1,1,1)) = 4,$$
$$d_H((1,0,0,1),(0,1,1,0)) = 4,$$
$$d_H((1,0,0,1),(1,1,1,1)) = 2,$$
$$d_H((0,1,1,0),(1,1,1,1)) = 2.$$

Portanto, temos $d_{\min} = 2$.

A distância mínima de Hamming é uma característica muito importante dos códigos de Hamming, pois está diretamente relacionada à sua capacidade de detecção e correção de erros. A análise dessa distância permite projetar códigos com propriedades específicas de detecção e correção, adaptando-os a diferentes aplicações. Isso nos leva diretamente ao próximo tópico.

1.6 Capacidade de Correção de Erros de um Código

A **capacidade de correção de erros** κ de um código $\mathcal{C} \subseteq \mathbb{F}_q^n$ está relacionada à sua capacidade de corrigir erros introduzidos durante a transmissão de palavras-código. Um código de Hamming pode detectar até $d_{\min} - 1$ erros e corrigir até κ erros. Este é um Teorema que será devidamente enunciado e demonstrado mais a frente.

Definimos κ como:

$$\kappa = \left\lfloor \frac{d_{\min} - 1}{2} \right\rfloor,$$

onde $d_{\min}$ é a distância mínima do código $\mathcal{C}$, e $\lfloor x \rfloor$ é o maior inteiro menor ou igual a x.

Essa definição garante que o código possa corrigir até κ erros em qualquer palavra-código, pois t erros ou menos não deslocam a palavra recebida para fora da esfera de correção do código, como ainda veremos.

Exemplo 1.7. .

1. ***Código com*** $d_{min} = 3$***:*** *Para um código* $\mathcal{C}$ *com* $d_{min} = 3$*:*

$$\kappa = \left\lfloor \frac{3-1}{2} \right\rfloor = \lfloor 1 \rfloor = 1.$$

Este código pode corrigir até um erro.

2. ***Código com*** $d_{min} = 5$***:*** *Se* $d_{min} = 5$*, então:*

$$\kappa = \left\lfloor \frac{5-1}{2} \right\rfloor = \lfloor 2 \rfloor = 2.$$

Este código pode corrigir até dois erros.

3. ***Código com*** $d_{min} = 2$***:*** *Para* $d_{min} = 2$*, temos:*

$$\kappa = \left\lfloor \frac{2-1}{2} \right\rfloor = \lfloor 0.5 \rfloor = 0.$$

Este código não pode corrigir erros, mas pode detectar um erro.

Exemplo elaborado pela autora.

1.6.1 Relação entre κ e $d_{\min}$

A capacidade κ de correção de erros é diretamente proporcional à distância mínima $d_{\min}$ do código. Um $d_{\min}$ maior resulta em maior κ, o que aumenta a robustez do código contra erros. Por outro lado, aumentar $d_{\min}$ pode reduzir a capacidade de palavras do código.

1.7 Codificação, Decodificação e a Estrutura do Código

Nesta seção, abordaremos a estrutura de códigos lineares e o processo de codificação, com ênfase no código de Hamming $(7, 4)$, que é amplamente utilizado devido à sua eficiência na detecção e correção de erros.

> Codificação é o processo de mapeamento, ou seja, é uma conversão de uma dada sequência de dígitos (alfabeto fonte) em uma outra sequência de dígitos (alfabeto do código). (NICOLETTI, 2015)

Definição 1.8 ((BETTEN MICHAEL BRAUN; WASSERMANN, 2006), p.12)**.** .

Um (n, k)*-código é um código linear que possui palavras de comprimento* n*, sendo* k *os dígitos de informação e* $n - k$ *os dígitos de paridade, onde* $n > k$*.*

1.7.1 Código de Hamming $(7, 4)$

O código de Hamming $(7, M, 4)$, também comumente chamado apenas de código de Hamming $(7, 4)$ é um código linear onde:

- $n = 7$ representa o comprimento da palavra-código, ou seja, o número total de bits (informação e paridade);
- $k = 4$ é o número de bits de informação na palavra-código;
- Os $n - k = 3$ bits restantes são bits de paridade, usados para correção de erros.
- M é o número de palavras do código.

Os bits de paridade são formados como combinações lineares dos bits de informação, garantindo a capacidade de detectar e corrigir até um erro em cada palavra-código.

Definição dos Bits de Informação e Paridade

Considere os bits de informação a, b, c, d. Os bits de paridade são definidos como:

$$p_1 = a + b, \quad p_2 = a + c, \quad p_3 = a + d,$$

onde as operações são realizadas no corpo finito $\mathbb{F}_2$ (aritmética módulo 2).

Assim, uma palavra-código $\mathbf{c} \in \mathbb{F}_2^7$ pode ser representada como:

$$\mathbf{c} = (a, b, c, d, p_1, p_2, p_3) = (a, b, c, d, a + b, a + c, a + d).$$

Exemplo 1.9. *Código de Hamming (7, 4, 4)*

$$C = \{(0,0,0,0,0,0,0), (1,0,1,1,1,0,0), (0,1,1,1,1,1,1), (1,1,0,0,0,1,1)\}$$

Esse código usa a combinação acima para gerar os dígitos de paridade.

Exemplo elaborado pela autora.

1.7.2 Matriz Geradora

A matriz geradora G é utilizada para construir todas as palavras-código a partir de combinações lineares dos bits de informação. Ela é responsável por **codificar** a informação, adicionando a ela, bits de redundância. Para o código de Hamming $(7, 4)$, a matriz geradora tem a seguinte forma:

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

Nesta matriz:

- As quatro primeiras colunas formam a matriz identidade I_4, que corresponde aos bits de informação a, b, c, d;
- As três colunas restantes formam a matriz de paridade P, que define como os bits de paridade p_1, p_2, p_3 são calculados.

Definição 1.10 ((BETTEN MICHAEL BRAUN; WASSERMANN, 2006), p.12 e (MILIES, 2009), p.25)**.** .

Seja C um código linear de parâmetros (n, k) sobre $\mathbb{F}_q$, onde n é o comprimento das palavras-código e k é o número de dígitos de informação. A ***matriz geradora*** *G é uma matriz de dimensão $k \times n$ que define a estrutura do código C. Cada palavra-código $\mathbf{c} \in C$ pode ser obtida como uma combinação linear dos vetores linha de G, ou seja:*

$$\mathbf{c} = \mathbf{u} \cdot G,$$

onde $\mathbf{u} \in \mathbb{F}_q^k$ é uma palavra de informação.

Estrutura de G

A matriz geradora G possui as seguintes características:

- G tem k linhas, correspondendo ao número de símbolos de informação.
- G tem n colunas, correspondendo ao comprimento total das palavras-código.
- G pode ser escrita na forma padrão $G = [I_k \mid P]$, onde I_k é a matriz identidade de ordem k e P é uma submatriz associada às combinações lineares que geram os bits de paridade. Para construir P, deve-se escolher uma combinação linear que se queira usar no código em questão, e embuí-la na matriz.

Propriedades

- Cada linha de G representa uma palavra base do espaço C.
- O espaço gerado por G, denotado por $\mathrm{Im}(G)$, contém todas as palavras-código de C.

Construção de Palavras-Código

Dada uma palavra de informação $\mathbf{u} = (u_1, u_2, \ldots, u_k) \in \mathbb{F}_q^k$, a palavra-código correspondente $\mathbf{c}$ é obtida como:

$$\mathbf{c} = \mathbf{u} \cdot G.$$

Para um código linear com $k = 4$ e $n = 7$, considere a matriz geradora:

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

Aqui, I_k é a submatriz identidade $\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$, e P é a submatriz $\begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$.

Para esse código, a matriz P usa a combinação linear $(a, b, c, d, a + b + d, a + c + d, b + c + d)$. Para a palavra de informação $\mathbf{u} = (1, 0, 1, 1)$, a palavra-código é calculada como:

$$\mathbf{c} = (1, 0, 1, 1) \cdot G = (1, 0, 1, 1, 0, 1, 0).$$

Exemplo 1.11. *Considere a palavra de informação* $\mathbf{u} = (1, 0, 1, 1)$*. A palavra-código correspondente é calculada como:*

$$\mathbf{c} = \mathbf{u} \cdot G.$$

Substituindo os valores, temos:

$$\mathbf{c} = (1, 0, 0, 0) \cdot \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix} = (1, 0, 1, 1, 1, 1, 0)$$

Dessa forma, a palavra-código $\mathbf{c}$ *contém os bits de informação e os bits de paridade corretamente calculados.*

Exemplo elaborado pela autora.

Exemplo 1.12. *Código com* $n = 6$ *e* $k = 3$

Considere um código linear com $n = 6$ *e* $k = 3$*. Neste caso, os bits de informação são* a, b, c*, e os bits de paridade* p_1, p_2, p_3 *são definidos como:*

$$p_1 = a + b, \quad p_2 = b + c, \quad p_3 = a + b + c,$$

com as operações realizadas no corpo $\mathbb{F}_2$*.*

A matriz geradora G *para este código é dada por:*

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

Para codificar a palavra de informação $\mathbf{u} = (1, 0, 1)$*, calculamos:*

$$\mathbf{c} = \mathbf{u} \cdot G.$$

Substituindo os valores, temos:

$$\mathbf{c} = (1, 0, 1) \cdot \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix} = (1, 0, 1, 1+0, 0+1, 1+1)$$

O resultado é:

$$\mathbf{c} = (1, 0, 1, 1, 1, 0).$$

Exemplo elaborado pela autora.

Exemplo 1.13. *Neste exemplo, temos* $n = 5$ *e* $k = 2$*. Os bits de informação* a, b *e os bits de paridade* p_1, p_2, p_3 *são calculados como:*

$$p_1 = a, \quad p_2 = b, \quad p_3 = a + b.$$

A matriz geradora G *para este código é:*

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

Para codificar a palavra de informação $\mathbf{u} = (1, 1)$*, realizamos:*

$$\mathbf{c} = \mathbf{u} \cdot G.$$

Substituindo os valores, temos:

$$\mathbf{c} = (1, 1) \cdot \begin{bmatrix} 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 \end{bmatrix} = (1, 1, 1, 1, 1+1)$$

O resultado é:

$$\mathbf{c} = (1, 1, 1, 1, 0).$$

Exemplo elaborado pela autora.

1.7.3 Matriz de Verificação

Além da matriz geradora, que é usada para codificação, outra ferramenta importante nos códigos corretores de erros é a **matriz de verificação**, denotada por Δ ou por H. Essa matriz é utilizada para decodificar palavras-código recebidas, verificando se elas possuem

erros, isto é, se elas satisfazem as relações de paridade definidas pela matriz geradora. Para o código de Hamming (7,4), a matriz de verificação tem a seguinte forma:

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

Definição 1.14 ((BETTEN MICHAEL BRAUN; WASSERMANN, 2006), p.20)**.** .

Seja C um código linear de parâmetros (n, k) sobre $\mathbb{F}_q$, onde n é o comprimento das palavras-código e k é o número de símbolos de informação. A matriz de verificação Δ é uma matriz de dimensão $(n-k) \times n$ que satisfaz a propriedade:

$$\mathbf{c} \cdot \Delta^\top = \mathbf{0},$$

para toda palavra-código $\mathbf{c} \in C$.

A matriz Δ é construída a partir do complemento linear das linhas da matriz geradora G. Formalmente, se G é uma matriz de dimensão $k \times n$, Δ é uma matriz de dimensão $(n-k) \times n$ tal que:

$$G \cdot \Delta^\top = 0.$$

Estrutura de Δ

A matriz Δ possui as seguintes características:

- Cada linha de Δ representa uma combinação linear dos bits de paridade.
- A matriz tem $n-k$ linhas, correspondendo ao número de bits de paridade no código.
- A matriz pode ser escrita na forma $\Delta = [P^\top \mid I_{n-k}]$, onde P é uma submatriz associada às combinações lineares e I_{n-k} é a matriz identidade de ordem $n-k$.

Detecção de Erros

Ao receber uma palavra $\mathbf{r} \in \mathbb{F}_q^n$, a matriz Δ é utilizada para calcular o vetor síndrome:

$$\mathbf{s} = \mathbf{r} \cdot \Delta^\top.$$

Se $\mathbf{s} = \mathbf{0}$, então $\mathbf{r}$ é uma palavra-código válida. Caso contrário, o vetor $\mathbf{s}$ indica que erros ocorreram durante a transmissão.

O vetor diferença $\mathbf{e}$ entre um vetor recebido $\mathbf{y}$ e o vetor transmitido $\mathbf{x}$ chama-se o vetor erro, isto é,

$$\mathbf{e} = \mathbf{y} - \mathbf{x}.$$

Note que o peso do vetor erro corresponde, precisamente, ao número de erros ocorridos numa palavra recebida. (MILIES, 2009)

Propriedades

- Toda palavra-código $\mathbf{c} \in C$ satisfaz $\mathbf{c} \cdot \Delta^\top = \mathbf{0}$.
- A matriz Δ define o espaço ortogonal gerado pelas linhas de G.
- A verificação por meio de Δ é eficiente, pois envolve apenas multiplicações e somas em $\mathbb{F}_q$.

Relação com o Espaço Ortogonal

A matriz de verificação Δ está intimamente relacionada à matriz geradora G de um código linear C. Essa relação pode ser formalizada utilizando o conceito de espaços ortogonais em álgebra linear.

Seja $\mathbb{F}_q^n$ o espaço vetorial de dimensão n sobre o corpo finito $\mathbb{F}_q$. As imagens das matrizes G e Δ, denotadas por $\mathrm{Im}(G)$ e $\mathrm{Im}(\Delta)$, satisfazem a seguinte propriedade:

$$\mathrm{Im}(G) \perp \mathrm{Im}(\Delta),$$

onde $\perp$ indica que os subespaços são ortogonais, ou seja, para todo $\mathbf{u} \in \mathrm{Im}(G)$ e $\mathbf{v} \in \mathrm{Im}(\Delta)$, temos:

$$\mathbf{u} \cdot \mathbf{v} = 0.$$

A ortogonalidade implica que cada palavra-código $\mathbf{c} \in C$, gerada por G, satisfaz:

$$\mathbf{c} \cdot \Delta^\top = \mathbf{0}.$$

Consequentemente, o espaço gerado por Δ é o complemento ortogonal do espaço gerado por G, garantindo que:

$$\dim(\mathrm{Im}(G)) + \dim(\mathrm{Im}(\Delta)) = n.$$

A matriz Δ pode ser explicitamente construída para satisfazer essa relação. Se G é uma matriz $k \times n$, então Δ é uma matriz $(n-k) \times n$, e suas linhas são combinações lineares ortogonais às linhas de G.

Verificação da ortogonalidade

Considere um código linear C de parâmetros $(7, 4)$ sobre $\mathbb{F}_2$, com a matriz geradora:

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

A matriz de verificação $\Delta = [P^\top \mid I_{n-k}]$, pode ser escrita como:

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

Para verificar a ortogonalidade, calculamos o produto $G \cdot \Delta^\top$:

$$G \cdot \Delta^\top = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1+1 & 1+1 & 1+1 \\ 1+1 & 0 & 1+1 \\ 1+1 & 1+1 & 0 \\ 0 & 1+1 & 1+1 \end{bmatrix}.$$

O resultado é a matriz nula:

$$G \cdot \Delta^\top = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}.$$

Esse resultado confirma que $\mathrm{Im}(G)$ e $\mathrm{Im}(\Delta)$ são subespaços ortogonais de $\mathbb{F}_2^7$, e a matriz Δ pode ser usada para verificar a validade de palavras-código geradas por G.

Exemplo 1.15. *Considere um código linear C de parâmetros $(7, 4)$ sobre $\mathbb{F}_2$, com a seguinte matriz geradora:*

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

As combinações lineares para os bits de paridade são:

$$p_1 = a + b, \quad p_2 = b + c, \quad p_3 = c + d.$$

- Para os bits de informação $a = 1, b = 0, c = 1, d = 0$, os bits de paridade são:

$$p_1 = 1 + 0 = 1, \quad p_2 = 0 + 1 = 1, \quad p_3 = 1 + 0 = 1.$$

A palavra-código é:

$$\mathbf{c}_1 = (1, 0, 1, 0, 1, 1, 1).$$

- Para $a = 0, b = 1, c = 0, d = 1$, os bits de paridade são:

$$p_1 = 0 + 1 = 1, \quad p_2 = 1 + 0 = 1, \quad p_3 = 0 + 1 = 1.$$

A palavra-código é:

$$\mathbf{c}_2 = (0, 1, 0, 1, 1, 1, 1).$$

A matriz de verificação correspondente é:

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

Verificação das Palavras-Código:

1. Para $\mathbf{c}_1 = (1, 0, 1, 0, 1, 1, 1)$, calculamos:

$$\mathbf{c}_1 \cdot \Delta^\top = (1, 0, 1, 0, 1, 1, 1) \cdot \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}.$$

A palavra-código é válida.

2. Para $\mathbf{c}_2 = (0, 1, 0, 1, 1, 1, 1)$, calculamos:

$$\mathbf{c}_2 \cdot \Delta^\top = (0, 1, 0, 1, 1, 1, 1) \cdot \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}.$$

A palavra-código é válida.

3. Para $\mathbf{c}_3 = (1,1,1,0,1,0,1)$*, que contém um erro, temos:*

$$\mathbf{c}_3 \cdot \Delta^\top = (1,1,1,0,1,0,1) \cdot \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}.$$

Como o resultado não é $\mathbf{0}$*, a palavra contém erros.*

Exemplo elaborado pela autora.

Corrigindo a palavra

> Uma vez detectado um erro, adotamos um critério de correção que irá substituir o elemento y recebido, pelo elemento x do código que está mais próximo de y. Para que a correção seja possível, será necessário que não haja ambiguidades quanto à determinação de tal elemento.
>
> A decodificação pode ser feita utilizando-se a distância de Hamming, ou seja, decodificando pela palavra mais próxima. (NICOLETTI, 2015)

No caso da palavra $\mathbf{c}_3 = (1,1,1,0,1,0,1)$, o produto com a matriz $\Delta^\top$ resultou em:

$$\mathbf{c}_3 \cdot \Delta^\top = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}.$$

O síndrome $\begin{bmatrix} 1 & 0 & 0 \end{bmatrix}$ indica que houve uma inconsistência nos bits associados à primeira combinação linear (relacionada a p_1), mas não em p_2 e p_3.

Como p_1 é definido por:

$$p_1 = a + b,$$

o erro deve estar em um dos bits a, b, ou no próprio p_1. Os bits c, d, p_2 e p_3 não influenciam esse dígito de paridade, e por isso não são os responsáveis pela inconsistência.

Para corrigir a palavra $\mathbf{c}_3 = (1,1,1,0,1,0,1)$, utilizamos o síndrome obtido:

$$\mathbf{c}_3 \cdot \Delta^\top = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}.$$

O vetor síndrome $\begin{bmatrix} 1 & 0 & 0 \end{bmatrix}$ indica que o erro está associado ao primeiro bit de informação a, já que a combinação linear $p_1 = a + b$ apresenta inconsistência. Se o erro

estivesse em b o vetor síndrome seria $\begin{bmatrix}1 & 1 & 0\end{bmatrix}$, já que b também é usado para a criação de p_2. Verificando os bits de paridade e recalculando p_1, temos:

$$p_1 = a + b = 1 + 1 = 0,$$

mas o valor registrado em p_1 é 1, indicando que o erro está no bit a.

Corrigindo o bit a, a palavra $\mathbf{c}_3$ torna-se:

$$\mathbf{c}_3' = (0, 1, 1, 0, 1, 0, 1).$$

Verificando a palavra corrigida:

$$\mathbf{c}_3' \cdot \Delta^\top = (0, 1, 1, 0, 1, 0, 1) \cdot \begin{bmatrix}1 & 0 & 0\\ 1 & 1 & 0\\ 0 & 1 & 1\\ 0 & 0 & 1\\ 1 & 0 & 0\\ 0 & 1 & 0\\ 0 & 0 & 1\end{bmatrix} = \begin{bmatrix}0 & 0 & 0\end{bmatrix}.$$

Como o vetor síndrome é $\mathbf{0}$, a palavra corrigida $\mathbf{c}_3'$ é válida.

Podemos ainda encontrar o vetor erro e:

$$e = c_3 - c_3' = (1, 1, 1, 0, 1, 0, 1) - (0, 1, 1, 0, 1, 0, 1) = (1, 0, 0, 0, 0, 0, 0)$$

O vetor e confirma que o erro ocorreu apenas no primeiro bit.

2 Transformações, Equivalências e Dualidade em Códigos Lineares

Neste capítulo, exploraremos conceitos que conectam propriedades algébricas e geométricas dos códigos lineares. Iniciaremos com a definição de homomorfismos de espaços vetoriais e isomorfismos em códigos lineares. Em seguida, abordaremos a equivalência de códigos, discutindo como transformações preservam ou alteram características específicas. Por fim, apresentaremos a teoria de códigos duais, ortogonais e autoduais, enfatizando sua relação com as propriedades de dualidade e ortogonalidade em espaços vetoriais.

2.1 Códigos Lineares como subespaços vetoriais

Começaremos apresentando uma definição, que será necessária para o aprofundamento do tema do capítulo nos próximos tópicos: a de que um código linear é um subespaço vetorial. Provaremos essa afirmação a seguir.

Definição 2.1. *(Código linear)((BETTEN MICHAEL BRAUN; WASSERMANN, 2006), p.12).*

Seja $C \subseteq \mathbb{F}_q^n$ um código linear. Temos que, C é um conjunto de vetores de comprimento n sobre o corpo finito $\mathbb{F}_q$, gerado por uma matriz geradora Γ. Assim, C é o conjunto de todas as combinações lineares das linhas de Γ, ou seja:

$$C = \{v \cdot \Gamma \mid v \in \mathbb{F}_q^k\},$$

onde $\Gamma \in \mathbb{F}_q^{k \times n}$ é uma matriz de dimensão $k \times n$ com $k \leq n$.

Queremos provar que C é um subespaço vetorial de $\mathbb{F}_q^n$, o que exige que C satisfaça as seguintes propriedades:

1. C é fechado sob adição de vetores.
2. C é fechado sob multiplicação escalar.
3. C contém o vetor nulo.

Demonstração: Seja C um código linear em $\mathbb{F}_q^n$.

1. Fechamento sob adição de vetores

Sejam $c_1, c_2 \in C$. Por definição, existem $v_1, v_2 \in \mathbb{F}_q^k$ tais que:

$$c_1 = v_1 \cdot \Gamma \quad \text{e} \quad c_2 = v_2 \cdot \Gamma.$$

A soma $c_1 + c_2$ é dada por:

$$c_1 + c_2 = (v_1 \cdot \Gamma) + (v_2 \cdot \Gamma).$$

Usando a distributividade da multiplicação de matrizes sobre a adição, temos:

$$c_1 + c_2 = (v_1 + v_2) \cdot \Gamma.$$

Como $v_1 + v_2 \in \mathbb{F}_q^k$, segue que $c_1 + c_2 \in C$. Portanto, C é fechado sob adição.

2. Fechamento sob multiplicação escalar

Seja $c \in C$. Por definição, existe $v \in \mathbb{F}_q^k$ tal que:

$$c = v \cdot \Gamma.$$

Para qualquer escalar $\alpha \in \mathbb{F}_q$, a multiplicação escalar αc é dada por:

$$\alpha c = \alpha(v \cdot \Gamma).$$

Usando a propriedade associativa da multiplicação escalar em espaços vetoriais, temos:

$$\alpha c = (\alpha v) \cdot \Gamma.$$

Como $\alpha v \in \mathbb{F}_q^k$, segue que $\alpha c \in C$. Portanto, C é fechado sob multiplicação escalar.

3. Presença do vetor nulo

Para C conter o vetor nulo de $\mathbb{F}_q^n$, basta mostrar que existe um vetor $v \in \mathbb{F}_q^k$ tal que:

$$v \cdot \Gamma = [0, 0, \ldots, 0].$$

Escolhendo $v = [0, 0, \ldots, 0] \in \mathbb{F}_q^k$, temos:

$$v \cdot \Gamma = [0, 0, \ldots, 0],$$

que é o vetor nulo. Logo, C contém o vetor nulo.

Como C satisfaz as três propriedades necessárias (fechamento sob adição, fechamento sob multiplicação escalar e presença do vetor nulo), segue que C é um subespaço vetorial de $\mathbb{F}_q^n$.

■

2.2 Homomorfismos de Espaços Vetoriais

Definição 2.2. *((BETTEN MICHAEL BRAUN; WASSERMANN, 2006), p.33).*

Sejam V e W dois espaços vetoriais sobre o mesmo corpo $\mathbb{F}$. Uma função $T : V \to W$ é chamada de homomorfismo de espaços vetoriais se, para quaisquer $\mathbf{u}, \mathbf{v} \in V$ e $c \in \mathbb{F}$, as seguintes propriedades são satisfeitas:

1. ***Preservação da Adição**: $T(\mathbf{u} + \mathbf{v}) = T(\mathbf{u}) + T(\mathbf{v})$;*
2. ***Preservação da Multiplicação Escalar**: $T(c\mathbf{u}) = cT(\mathbf{u})$.*

Essas propriedades garantem que T respeita a estrutura vetorial de V ao transformar elementos em W. Em outras palavras, a imagem de T mantém as operações básicas de adição e multiplicação por escalares.

Relação com Transformações Lineares

Um homomorfismo de espaços vetoriais é, por definição, uma transformação linear. Isso ocorre porque a linearidade é caracterizada exatamente pelas duas propriedades descritas acima: preservação da adição e da multiplicação escalar. Assim, toda transformação linear é um homomorfismo de espaços vetoriais, e todo homomorfismo de espaços vetoriais é uma transformação linear.

Exemplo 2.3. *(Projeção no Espaço)*

Considere $V = \mathbb{R}^3$, $W = \mathbb{R}^2$ e a transformação $T : \mathbb{R}^3 \to \mathbb{R}^2$ definida por:

$$T(x, y, z) = (x, y).$$

Aqui, T "projeta"cada vetor $(x, y, z) \in \mathbb{R}^3$ em um vetor do plano $\mathbb{R}^2$, ignorando o componente z.

Vamos verificar que T é um homomorfismo:

- *Para a adição, sejam $\mathbf{u} = (x_1, y_1, z_1)$ e $\mathbf{v} = (x_2, y_2, z_2)$:*

$$T(\mathbf{u} + \mathbf{v}) = T(x_1 + x_2, y_1 + y_2, z_1 + z_2) = (x_1 + x_2, y_1 + y_2).$$

Por outro lado:

$$T(\mathbf{u}) + T(\mathbf{v}) = (x_1, y_1) + (x_2, y_2) = (x_1 + x_2, y_1 + y_2).$$

Assim, $T(\mathbf{u} + \mathbf{v}) = T(\mathbf{u}) + T(\mathbf{v})$.

- *Para a multiplicação por escalar, seja* $c \in \mathbb{R}$ *e* $\mathbf{u} = (x, y, z)$*:*

$$T(c\mathbf{u}) = T(cx, cy, cz) = (cx, cy).$$

Por outro lado:

$$cT(\mathbf{u}) = c(x, y) = (cx, cy).$$

Assim, $T(c\mathbf{u}) = cT(\mathbf{u})$.

Logo, T *é um homomorfismo (ou transformação linear).*

Exemplo elaborado pela autora.

Exemplo 2.4. *Considere* $V = \mathbb{R}^2$*,* $W = \mathbb{R}^2$*, e a transformação* $T : \mathbb{R}^2 \to \mathbb{R}^2$ *definida por:*

$$T(\mathbf{v}) = A\mathbf{v}, \quad \text{onde } A = \begin{bmatrix} 1 & 2 \\ 0 & 1 \end{bmatrix}.$$

Seja $\mathbf{v} = \begin{bmatrix} x \\ y \end{bmatrix}$*. O cálculo da transformação é:*

$$T(\mathbf{v}) = \begin{bmatrix} 1 & 2 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x + 2y \\ y \end{bmatrix}.$$

Vamos verificar que T *é um homomorfismo:*

- *Para a adição, sejam* $\mathbf{v}_1 = \begin{bmatrix} x_1 \\ y_1 \end{bmatrix}$ *e* $\mathbf{v}_2 = \begin{bmatrix} x_2 \\ y_2 \end{bmatrix}$*:*

$$T(\mathbf{v}_1 + \mathbf{v}_2) = T\left(\begin{bmatrix} x_1 + x_2 \\ y_1 + y_2 \end{bmatrix}\right) = \begin{bmatrix} x_1 + x_2 + 2(y_1 + y_2) \\ y_1 + y_2 \end{bmatrix}.$$

Por outro lado:

$$T(\mathbf{v}_1) + T(\mathbf{v}_2) = \begin{bmatrix} x_1 + 2y_1 \\ y_1 \end{bmatrix} + \begin{bmatrix} x_2 + 2y_2 \\ y_2 \end{bmatrix} = \begin{bmatrix} x_1 + x_2 + 2(y_1 + y_2) \\ y_1 + y_2 \end{bmatrix}.$$

Assim, $T(\mathbf{v}_1 + \mathbf{v}_2) = T(\mathbf{v}_1) + T(\mathbf{v}_2)$.

- *Para a multiplicação por escalar, seja* $c \in \mathbb{R}$*:*

$$T(c\mathbf{v}) = T\left(\begin{bmatrix} cx \\ cy \end{bmatrix}\right) = \begin{bmatrix} cx + 2cy \\ cy \end{bmatrix}.$$

Por outro lado:

$$cT(\mathbf{v}) = c\begin{bmatrix} x + 2y \\ y \end{bmatrix} = \begin{bmatrix} cx + 2cy \\ cy \end{bmatrix}.$$

Assim, $T(c\mathbf{v}) = cT(\mathbf{v})$.

Logo, T é um homomorfismo de $\mathbb{R}^2$ em $\mathbb{R}^2$.

Exemplo elaborado pela autora.

Esses exemplos demonstram como os homomorfismos preservam a estrutura vetorial e podem ser representados por diferentes transformações.

2.2.1 Homomorfismos em Códigos Lineares

Seja C um código linear de parâmetros $(7, 4)$ sobre $\mathbb{F}_2$ com matriz geradora:

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

Cada palavra-código $\mathbf{c} \in C$ é obtida pela multiplicação $\mathbf{c} = \mathbf{u} \cdot G$, onde $\mathbf{u} \in \mathbb{F}_2^4$ é uma palavra de informação.

Agora, consideremos uma transformação $T : C \to \mathbb{F}_2^3$, definida como:

$$T(\mathbf{c}) = (\mathbf{c}_5, \mathbf{c}_6, \mathbf{c}_7),$$

onde $\mathbf{c}_i$ denota o i-ésimo componente de $\mathbf{c}$.

Essa transformação extrai os bits de paridade p_1, p_2, p_3 da palavra-código $\mathbf{c}$. Vamos verificar que T é um homomorfismo.

Verificação da Linearidade

Sejam $\mathbf{c}_1, \mathbf{c}_2 \in C$. Como C é um subespaço vetorial, sabemos que $\mathbf{c}_1 + \mathbf{c}_2 \in C$. Vamos verificar as propriedades:

- **Preservação da Adição**:

$$T(\mathbf{c}_1 + \mathbf{c}_2) = \Big((\mathbf{c}_1 + \mathbf{c}_2)_5, (\mathbf{c}_1 + \mathbf{c}_2)_6, (\mathbf{c}_1 + \mathbf{c}_2)_7\Big).$$

Usando a propriedade distributiva em $\mathbb{F}_2$, temos:

$$T(\mathbf{c}_1 + \mathbf{c}_2) = \Big(\mathbf{c}_{1,5} + \mathbf{c}_{2,5}, \mathbf{c}_{1,6} + \mathbf{c}_{2,6}, \mathbf{c}_{1,7} + \mathbf{c}_{2,7}\Big).$$

Por outro lado:

$$T(\mathbf{c}_1) + T(\mathbf{c}_2) = (\mathbf{c}_{1,5}, \mathbf{c}_{1,6}, \mathbf{c}_{1,7}) + (\mathbf{c}_{2,5}, \mathbf{c}_{2,6}, \mathbf{c}_{2,7}) = \Big(\mathbf{c}_{1,5} + \mathbf{c}_{2,5}, \mathbf{c}_{1,6} + \mathbf{c}_{2,6}, \mathbf{c}_{1,7} + \mathbf{c}_{2,7}\Big).$$

Assim, $T(\mathbf{c}_1 + \mathbf{c}_2) = T(\mathbf{c}_1) + T(\mathbf{c}_2)$.

- **Preservação da Multiplicação Escalar**: Seja $\alpha \in \mathbb{F}_2$. Então:

$$T(\alpha\mathbf{c}) = \Big((\alpha\mathbf{c})_5, (\alpha\mathbf{c})_6, (\alpha\mathbf{c})_7\Big) = \Big(\alpha\mathbf{c}_5, \alpha\mathbf{c}_6, \alpha\mathbf{c}_7\Big).$$

  Por outro lado:

$$\alpha T(\mathbf{c}) = \alpha(\mathbf{c}_5, \mathbf{c}_6, \mathbf{c}_7) = (\alpha\mathbf{c}_5, \alpha\mathbf{c}_6, \alpha\mathbf{c}_7).$$

  Assim, $T(\alpha\mathbf{c}) = \alpha T(\mathbf{c})$.

Logo, T é um homomorfismo.

Exemplo 2.5. *Considere a palavra de informação* $\mathbf{u} = (1, 0, 1, 1)$. *A palavra-código correspondente é:*

$$\mathbf{c} = \mathbf{u} \cdot G = (1, 0, 1, 1, 0, 1, 1).$$

Aplicando T, *obtemos:*

$$T(\mathbf{c}) = (0, 1, 1),$$

que corresponde aos bits de paridade da palavra-código.

Exemplo elaborado pela autora.

2.3 Isomorfismos e Isometrias em Códigos Lineares

2.3.1 Isomorfismo

Definição 2.6. *((BETTEN MICHAEL BRAUN; WASSERMANN, 2006), p.32).*

Sejam C_1 *e* C_2 *dois códigos lineares sobre* $\mathbb{F}_q$ *com o mesmo comprimento* n. *Dizemos que* C_1 *e* C_2 *são* ***isomorfos*** *se existe uma transformação linear bijetiva* $\phi : \mathbb{F}_q^n \to \mathbb{F}_q^n$ *tal que:*

$$\phi(C_1) = C_2.$$

Isto significa que ϕ *preserva a soma vetorial e a multiplicação por escalar, ou seja, para todos* $\mathbf{u}, \mathbf{v} \in \mathbb{F}_q^n$ *e* $\alpha \in \mathbb{F}_q$, *temos:*

$$\phi(\mathbf{u} + \mathbf{v}) = \phi(\mathbf{u}) + \phi(\mathbf{v}), \quad \phi(\alpha\mathbf{u}) = \alpha\phi(\mathbf{u}).$$

Observação 2.7. *Um isomorfismo garante que os códigos* C_1 *e* C_2 *possuem a mesma estrutura vetorial, mas não necessariamente preservam a distância entre as palavras-código.*

Exemplo 2.8. *Considere os códigos lineares* C_1 *e* C_2, *ambos sobre* $\mathbb{F}_2$, *definidos por:*

$$C_1 = \{(0, 0, 0), (1, 0, 1), (0, 1, 1), (0, 1, 0)\},$$

$$C_2 = \{(0,0,0), (1,1,0), (1,1,1), (1,0,1)\}.$$

Definimos uma transformação linear $\phi : \mathbb{F}_2^3 \to \mathbb{F}_2^3$ *como:*

$$\phi(x_1, x_2, x_3) = (x_1 + x_2, x_3, x_2).$$

Aplicando ϕ *a cada palavra-código de* C_1*, obtemos:*

$$\phi(0,0,0) = (0,0,0), \quad \phi(1,0,1) = (1,1,0), \quad \phi(0,1,1) = (1,1,1), \quad \phi(0,1,0) = (1,0,1).$$

O conjunto resultante é exatamente C_2*. Assim,* ϕ *é um isomorfismo, pois preserva a estrutura vetorial. Note que a distância de Hamming entre as palavras não é necessariamente preservada; por exemplo:*

$$d_H((1,0,1),(0,1,1)) = 2 \quad \textit{em } C_1,$$

$$d_H((1,1,0),(1,1,1)) = 1 \quad \textit{em } C_2.$$

Exemplo elaborado pela autora.

2.3.2 Isometria

Definição 2.9. *((BETTEN MICHAEL BRAUN; WASSERMANN, 2006), p.30).*

Sejam C_1 *e* C_2 *dois códigos lineares sobre* $\mathbb{F}_q$ *com o mesmo comprimento* n*. Dizemos que* C_1 *e* C_2 *são* ***isométricos*** *se existe uma transformação linear* $\psi : \mathbb{F}_q^n \to \mathbb{F}_q^n$ *tal que:*

$$\psi(C_1) = C_2,$$

e ψ *preserva a distância de Hamming, ou seja, para todos* $\mathbf{u}, \mathbf{v} \in \mathbb{F}_q^n$*, temos:*

$$d_H(\psi(\mathbf{u}), \psi(\mathbf{v})) = d_H(\mathbf{u}, \mathbf{v}),$$

onde d_H *é a distância de Hamming.*

Observação 2.10. *Toda isometria é automaticamente um isomorfismo, mas o recíproco não é necessariamente verdadeiro. Isometrias garantem que a estrutura geométrica do código, em termos de distâncias entre palavras, seja preservada.*

Exemplo 2.11. *Considere novamente* C_1 *sobre* $\mathbb{F}_2$*, mas agora com:*

$$C_1 = \{(0,0,0), (1,0,1), (0,1,1), (1,1,0)\},$$

Definimos uma transformação linear $\psi : \mathbb{F}_2^3 \to \mathbb{F}_2^3$ *como:*

$$\psi(x_1, x_2, x_3) = (x_1, x_3, x_2).$$

Aplicando ψ a C_1, obtemos:

$$\psi(0,0,0) = (0,0,0), \quad \psi(1,0,1) = (1,1,0), \quad \psi(0,1,1) = (0,1,1), \quad \psi(1,1,0) = (1,0,1).$$

Temos então o código $C_1' = \psi(C_1)$:

$$C_1' = \{(0,0,0), (1,1,0), (0,1,1), (1,0,1)\}$$

Verificamos que a distância de Hamming é preservada:

$$d_H((1,0,1),(0,1,1)) = 2 \quad em\ C_1,$$

$$d_H((1,1,0),(0,1,1)) = 2 \quad em\ C_1'.$$

$$d_H((0,1,1),(1,1,0)) = 2 \quad em\ C_1,$$

$$d_H((0,1,1),(1,0,1)) = 2 \quad em\ C_1'.$$

$$d_H((1,0,1),(1,1,0)) = 2 \quad em\ C_1,$$

$$d_H((1,1,0),(1,0,1)) = 2 \quad em\ C_1'.$$

Assim, ψ é uma isometria. Além disso, como isometrias são isomorfismos, ψ também é um isomorfismo, mas com a propriedade adicional de preservar distâncias.

Exemplo elaborado pela autora.

2.3.3 Equivalência de Códigos

Após explorarmos os conceitos de isomorfismo e isometria em códigos lineares, introduzimos o conceito de equivalência entre códigos, que também preserva a estrutura e as propriedades de um código linear.

Definição 2.12. *((BETTEN MICHAEL BRAUN; WASSERMANN, 2006), p.35 e 36).*

Dois códigos lineares C_1 e C_2 de parâmetros (n,k) sobre $\mathbb{F}_q$ são ditos ***equivalentes*** *se existe uma transformação linear invertível T (uma isometria) que relaciona as palavras-código de C_1 com as de C_2. Mais formalmente, C_1 e C_2 são equivalentes se:*

$$\mathbf{c}_2 = T(\mathbf{c}_1), \quad \forall \mathbf{c}_1 \in C_1, \mathbf{c}_2 \in C_2.$$

Propriedades da Equivalência

A equivalência entre códigos preserva as seguintes propriedades:

- O comprimento n das palavras-código.

- A dimensão k do código.
- A distância mínima d do código.

Embora os códigos equivalentes possam diferir em suas representações, como as posições dos bits de informação e paridade, eles compartilham as mesmas capacidades de detecção e correção de erros.

Geração de um Código Equivalente

Para gerar um código equivalente C_2 a partir de C_1, podemos aplicar:

1. Permutação das Coordenadas: Reordenar as posições das palavras-código.
2. Escalonamento: Multiplicar as coordenadas das palavras-código por elementos não nulos de $\mathbb{F}_q$.
3. Combinação de Ambos: Aplicar uma permutação seguida de escalonamento.

Exemplo 2.13. *Código de Hamming* $(7, 4)$

Considere o código de Hamming $(7, 4)$ *definido pela matriz geradora:*

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

Definimos que toda palavra código gerada a partir da matriz G, pertence a um certo código C_1.

Aplicamos uma permutação das colunas de G, como $(1, 2, 3, 4, 5, 6, 7) \to (3, 2, 1, 4, 5, 7, 6)$. *A matriz* G'*, após a permutação, é:*

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

Agora, G' *gera um código* C_2 *equivalente a* C_1*. Qualquer palavra-código de* C_1 *pode ser transformada em uma palavra-código de* C_2 *pela permutação especificada.*

Verificação da Equivalência

Se tomarmos uma palavra de informação $\mathbf{u} = (1, 0, 1, 0)$*, a palavra-código gerada por G é:*

$$\mathbf{c}_1 = \mathbf{u} \cdot G = (1, 0, 1, 1, 1, 0, 1).$$

Aplicando a permutação, obtemos:

$$\mathbf{c}_2 = (1,0,1,1,0,1,1) \to (1,0,1,1,1,1,0).$$

Portanto, as palavras-código correspondentes em C_1 e C_2 são equivalentes.

Ao repetirmos esse processo com todas as outras palavras do código C_1, criamos um código equivalente C_2 que preserva a distância original de C_1 entre suas palavras código e também sua capacidade de detecção e correção de erros.

Exemplo elaborado pela autora.

Agora, vamos construir um código C que seja equivalente a um código C_0, ambos compostos por 6 palavras-código, e verificar a equivalência entre eles.

Código Original C_0:

O código C_0 é um código linear de comprimento $n = 3$ sobre $\mathbb{F}_2$. Suas palavras-código são dadas por:

$$C_0 = \{\mathbf{0}, \mathbf{c}_1, \mathbf{c}_2, \mathbf{c}_3, \mathbf{c}_4, \mathbf{c}_5\},$$

onde:

$$\mathbf{c}_1 = (1,0,0), \quad \mathbf{c}_2 = (0,1,0), \quad \mathbf{c}_3 = (0,0,1), \quad \mathbf{c}_4 = (1,1,0), \quad \mathbf{c}_5 = (1,0,1).$$

Transformação para um Código Equivalente C:

Para gerar um código equivalente C, aplicamos uma permutação das posições dos bits das palavras-código. Escolhemos a permutação $(1,2,3) \to (3,1,2)$. Assim, a transformação de cada palavra-código é dada por:

$$(x_1, x_2, x_3) \to (x_3, x_1, x_2).$$

As palavras-código de C são então:

$$C = \{\mathbf{0}, \mathbf{c}'_1, \mathbf{c}'_2, \mathbf{c}'_3, \mathbf{c}'_4, \mathbf{c}'_5\},$$

onde:

$$\mathbf{c}'_1 = (0,1,0), \quad \mathbf{c}'_2 = (0,0,1), \quad \mathbf{c}'_3 = (1,0,0), \quad \mathbf{c}'_4 = (0,1,1), \quad \mathbf{c}'_5 = (1,0,1).$$

Verificação da Equivalência:

Para verificar a equivalência, observamos que as palavras de C_0 podem ser obtidas a partir de C pela aplicação inversa da permutação $(3, 1, 2) \to (1, 2, 3)$. Por exemplo:

$$\mathbf{c}'_1 = (0, 1, 0) \to (1, 0, 0) = \mathbf{c}_1.$$

$$\mathbf{c}'_2 = (0, 0, 1) \to (0, 1, 0) = \mathbf{c}_2.$$

$$\mathbf{c}'_3 = (1, 0, 0) \to (0, 0, 1) = \mathbf{c}_3.$$

Para verificar se a distância de Hamming foi preservada, calculamos a distância de Hamming entre as palavras transformadas $\mathbf{c}_i$ e comparamos com a distância original entre $\mathbf{c}'_i$.

Distâncias Originais:

$$\begin{aligned} d_H(\mathbf{c}'_1, \mathbf{c}'_2) &= d_H((0, 1, 0), (0, 0, 1)) = 2, \\ d_H(\mathbf{c}'_1, \mathbf{c}'_3) &= d_H((0, 1, 0), (1, 0, 0)) = 2, \\ d_H(\mathbf{c}'_2, \mathbf{c}'_3) &= d_H((0, 0, 1), (1, 0, 0)) = 2. \end{aligned}$$

Distâncias Após a Transformação:

$$\begin{aligned} d_H(\mathbf{c}_1, \mathbf{c}_2) &= d_H((1, 0, 0), (0, 1, 0)) = 2, \\ d_H(\mathbf{c}_1, \mathbf{c}_3) &= d_H((1, 0, 0), (0, 0, 1)) = 2, \\ d_H(\mathbf{c}_2, \mathbf{c}_3) &= d_H((0, 1, 0), (0, 0, 1)) = 2. \end{aligned}$$

Como a distância entre todas as palavras foram preservadas, isso implica que a distância mínima do código foi preservada e consequentemente a sua capacidade de detecção e correção de erros também.

Esse processo confirma que C_0 e C são equivalentes, já que há uma correspondência biunívoca entre as palavras-código de ambos os conjuntos, preservando o comprimento e as propriedades estruturais do código linear.

2.4 Códigos Duais, Auto-ortogonais e Auto-Duais

2.4.1 Códigos Duais

Definição 2.14. *((BETTEN MICHAEL BRAUN; WASSERMANN, 2006), p.21 e 22).*

Seja C um código linear em $\mathbb{F}_q^n$, ou seja, $C \subseteq \mathbb{F}_q^n$, onde $\mathbb{F}_q$ é um corpo finito com q elementos. Definimos o ***código dual*** *de C, denotado por $C^\perp$, como:*

$$C^\perp := \{w \in \mathbb{F}_q^n \mid \langle c, w\rangle = 0,\ \forall c \in C\},$$

onde $\langle \cdot, \cdot \rangle$ representa o ***produto interno padrão****, definido por:*

$$\langle c, w\rangle := \sum_{i=1}^{n} c_i w_i \pmod q,$$

com $c = (c_1, c_2, \ldots, c_n) \in \mathbb{F}_q^n$ e $w = (w_1, w_2, \ldots, w_n) \in \mathbb{F}_q^n$.

Propriedades importantes:

- $C^\perp$ é um código linear em $\mathbb{F}_q^n$.
- Se C é um (n, k)-código, então $C^\perp$ é um $(n, n-k)$-código.
- O espaço vetorial $\mathbb{F}_q^n$ é a soma direta de C e $C^\perp$, ou seja:

$$\dim(C) + \dim(C^\perp) = n.$$

A seguir, iremos provar essas quatro propriedades de um código dual.

Demonstração: Propriedade 1: $C^\perp$ é um código linear

Para provar que $C^\perp$ é linear, basta mostrar que ele é fechado sob adição e multiplicação escalar.

- Seja $w_1, w_2 \in C^\perp$. Isso implica que $\langle c, w_1\rangle = 0$ e $\langle c, w_2\rangle = 0$ para todo $c \in C$. Logo:

$$\langle c, (w_1 + w_2)\rangle = \langle c, w_1\rangle + \langle c, w_2\rangle = 0 + 0 = 0.$$

  Assim, $w_1 + w_2 \in C^\perp$.

- Para $\alpha \in \mathbb{F}_q$ e $w \in C^\perp$, temos:

$$\langle c, (\alpha w)\rangle = \alpha\langle c, w\rangle = \alpha \cdot 0 = 0.$$

  Logo, $\alpha w \in C^\perp$.

Portanto, $C^{\perp}$ é fechado sob adição e multiplicação escalar, o que prova que $C^{\perp}$ é um subespaço vetorial e, consequentemente, um código linear.

Propriedade 2: Se C é um (n, k)-código, então $C^{\perp}$ é um $(n, n-k)$-código

Sabemos que C é um subespaço de $\mathbb{F}_q^n$. O código dual $C^{\perp}$ é definido como o conjunto de vetores ortogonais a C. Pelo teorema da dimensão de subespaços ortogonais em espaços vetoriais:

$$\dim(C) + \dim(C^{\perp}) = n.$$

Como $\dim(C) = k$, segue que $\dim(C^{\perp}) = n - k$. Assim, $C^{\perp}$ é um $(n, n-k)$-código.

Propriedade 3: O espaço $\mathbb{F}_q^n$ é a soma direta de C e $C^{\perp}$

Queremos mostrar que $\mathbb{F}_q^n$ pode ser escrito como a soma direta $\mathbb{F}_q^n = C \oplus C^{\perp}$. Isso significa provar as duas condições a seguir:

1. Todo vetor $v \in \mathbb{F}_q^n$ pode ser escrito como $v = c + w$, onde $c \in C$ e $w \in C^{\perp}$.
2. A interseção $C \cap C^{\perp}$ contém apenas o vetor nulo, ou seja, $C \cap C^{\perp} = \{0\}$.

(1) Representação de qualquer vetor como soma $v = c + w$:

Dado $v \in \mathbb{F}_q^n$, sabemos que C é um subespaço de dimensão k e que $C^{\perp}$ é o subespaço complementar, de dimensão $n - k$. Como o espaço vetorial $\mathbb{F}_q^n$ tem dimensão n, qualquer vetor $v \in \mathbb{F}_q^n$ pode ser projetado em C e $C^{\perp}$.

Mais precisamente:

- Seja $c \in C$, a projeção de v sobre C.
- Seja $w = v - c$, que representa a parte de v que não está em C.

Como $w = v - c$, é claro que w está ortogonal a todos os vetores em C. Assim, $w \in C^{\perp}$. Logo, v pode ser escrito como a soma:

$$v = c + w, \quad \text{onde } c \in C \text{ e } w \in C^{\perp}.$$

(2) Interseção de C e $C^{\perp}$:

Para mostrar que $C \cap C^{\perp} = \{0\}$, tomemos $v \in C \cap C^{\perp}$. Isso significa que $v \in C$ e $v \in C^{\perp}$ simultaneamente. Por definição de $C^{\perp}$, temos:

$$\langle v, c \rangle = 0, \quad \forall c \in C.$$

Agora, como $v \in C$, podemos escolher $c = v$. Substituindo:

$$\langle v, v \rangle = 0.$$

No produto interno padrão sobre $\mathbb{F}_q^n$, $\langle v, v \rangle = 0$ implica que $v = 0$, ou seja, v é o vetor nulo. Portanto:

$$C \cap C^{\perp} = \{0\}.$$

Combinando as partes (1) e (2), mostramos que:

$$\mathbb{F}_q^n = C \oplus C^{\perp},$$

onde C e $C^{\perp}$ são subespaços complementares e a soma é direta, pois $C \cap C^{\perp} = \{0\}$.

■

Corolário 2.15 ((BETTEN MICHAEL BRAUN; WASSERMANN, 2006), p.22)**.** .

As matrizes de verificação de um código C são as matrizes geradoras do código dual $C^{\perp}$, e vice-versa. Dualmente, as matrizes de verificação do código dual $C^{\perp}$ são as matrizes geradoras do código C.

Seja C um (n, k)-código linear em $\mathbb{F}_q^n$, e seja Γ uma matriz geradora de C. Queremos mostrar que:

1. A matriz geradora Γ de C é uma matriz de verificação de paridade para o código dual $C^{\perp}$.
2. Analogamente, a matriz geradora de $C^{\perp}$ é uma matriz de verificação de paridade para C.

Demonstração: (1) A matriz geradora de C é uma matriz de verificação de paridade para $C^{\perp}$

A matriz geradora Γ de C tem dimensão $k \times n$, onde k é a dimensão de C. Os vetores $c \in C$ são da forma $c = v \cdot \Gamma$, onde $v \in \mathbb{F}_q^k$ é um vetor de dimensão k.

Por definição, o código dual $C^{\perp}$ consiste nos vetores $w \in \mathbb{F}_q^n$ que satisfazem:

$$\langle c, w \rangle = 0, \quad \forall c \in C.$$

Substituímos $c = v \cdot \Gamma$ na equação acima:

$$\langle c, w \rangle = \langle v \cdot \Gamma, w \rangle.$$

Usando a propriedade distributiva do produto interno, temos:

$$\langle v \cdot \Gamma, w \rangle = v \cdot (\Gamma w^\top).$$

Para que $\langle c, w \rangle = 0$ para todo $v \in \mathbb{F}_q^k$, é necessário que:

$$\Gamma w^\top = 0.$$

Portanto, o vetor w pertence ao código dual $C^\perp$ se, e somente se, ele satisfaz:

$$w \cdot \Gamma^\top = 0.$$

Isso mostra que Γ é uma matriz de verificação para $C^\perp$, pois verifica quais vetores $w \in \mathbb{F}_q^n$ pertencem ao dual $C^\perp$.

(2) a matriz geradora de $C^\perp$ é uma matriz de verificação para C

Seja H a matriz geradora de $C^\perp$. Por definição, qualquer vetor $w \in C^\perp$ pode ser escrito como $w = u \cdot H$, onde $u \in \mathbb{F}_q^{n-k}$.

Agora, queremos verificar que H também serve como matriz de verificação para C. Por definição, um vetor $c \in C$ satisfaz:

$$\langle w, c \rangle = 0, \quad \forall c \in C.$$

Substituindo $w = u \cdot H$ na equação acima:

$$\langle w, c \rangle = \langle u \cdot H, c \rangle.$$

Usando a propriedade distributiva do produto interno, temos:

$$\langle u \cdot H, c \rangle. = u \cdot (Hc^\top)$$

Ou seja, temos:

$$Hc^\top = 0.$$

Isso ocorre porque $C^\perp$ é o conjunto de todos os vetores ortogonais a C. Assim, a matriz geradora H de $C^\perp$ captura as relações ortogonais que definem o código C.

■

Portanto:

- A matriz geradora Γ de C é uma matriz de verificação de paridade para $C^{\perp}$, pois determina os vetores $w \in C^{\perp}$ através da condição $\Gamma w^{\top} = 0$.
- A matriz geradora de $C^{\perp}$ desempenha o papel análogo para C, verificando os vetores $c \in C$ através da condição $Hc^{\top} = 0$.

Exemplo 2.16. *(Código de Hamming* $(7, 4)$ *e seu dual)*

Matriz geradora do código C:

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

Matriz geradora do código dual $C^{\perp}$:

$$H = \begin{bmatrix} 1 & 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 \end{bmatrix} = \Delta,$$

onde Δ *é a matriz de verificação do código* C*.*

1. Gerando uma palavra de C:

Escolhemos $v = [1, 0, 1, 0] \in \mathbb{F}_2^4$ *como sendo o vetor de informação. Multiplicando* v *por* Γ*:*

$$c = v \cdot \Gamma = [1, 0, 1, 0] \cdot \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix} = [1, 0, 1, 0, 1, 0, 1].$$

Portanto, $c = [1, 0, 1, 0, 1, 0, 1] \in C$*.*

2. Gerando uma palavra de $C^{\perp}$:

Escolhemos $u = [1, 0, 0] \in \mathbb{F}_2^3$*. Multiplicando* u *por* H*:*

$$w = u \cdot H = [1, 0, 0] \cdot \begin{bmatrix} 1 & 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 \end{bmatrix} = [1, 1, 0, 1, 1, 0, 0].$$

Portanto, $w = [1, 0, 1, 0, 1, 0, 1] \in C^{\perp}$.

3. Verificação da ortogonalidade:

Calculamos o produto interno $\langle c, w \rangle$*:*

$$\langle c, w \rangle = [1, 0, 1, 0, 1, 0, 1] \cdot [1, 1, 0, 1, 1, 0, 0]^{\top} = 1 \cdot 1 + 0 \cdot 1 + 1 \cdot 0 + 0 \cdot 1 + 1 \cdot 1 + 0 \cdot 0 + 1 \cdot 0$$

$$= 1 + 0 + 0 + 0 + 1 + 0 + 0 = 0 \pmod 2.$$

Logo, $\langle c, w \rangle = 0$*, confirmando que* c *e* w *são ortogonais.*

Exemplo elaborado pela autora.

Exemplo 2.17. *(Outro código* $(7, 4)$ *e seu dual)*

Matriz geradora do código C:

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

Matriz geradora do código dual $C^{\perp}$:

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

1. Gerando uma palavra de C:

Escolhemos $v = [1, 1, 0, 1] \in \mathbb{F}_2^4$*. Multiplicando* v *por* Γ'*:*

$$c = v \cdot \Gamma' = [1, 1, 0, 1] \cdot \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix} = [1, 1, 0, 1, 1, 0, 0].$$

Portanto, $c = [1, 1, 0, 1, 1, 0, 0] \in C$.

2. Gerando uma palavra de $C^{\perp}$:

Escolhemos $u = [0, 1, 1] \in \mathbb{F}_2^3$. *Multiplicando* u *por* H'*:*

$$w = u \cdot H' = [0, 1, 1] \cdot \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 \end{bmatrix} = [1, 1, 0, 0, 0, 1, 1].$$

Portanto, $w = [1, 1, 0, 0, 0, 1, 1] \in C^{\perp}$.

3. Verificação da ortogonalidade:

Calculando o produto interno $\langle c, w \rangle$*:*

$$\langle c, w \rangle = [1, 1, 0, 1, 1, 0, 0] \cdot [1, 1, 0, 0, 0, 1, 1]^{\top} = 1 \cdot 1 + 1 \cdot 1 + 0 \cdot 0 + 1 \cdot 0 + 1 \cdot 0 + 0 \cdot 1 + 0 \cdot 1$$

$$= 1 + 1 + 0 + 0 + 0 + 0 + 0 = 0 \pmod 2.$$

Logo, $\langle c, w \rangle = 0$*, confirmando que* c *e* w *são ortogonais.*

Exemplo elaborado pela autora.

2.4.2 Códigos Auto-Ortogonais

Seja $C \subseteq \mathbb{F}_q^n$ um código linear de comprimento n sobre o corpo finito $\mathbb{F}_q$. O **código dual** $C^{\perp}$ de C é definido como:

$$C^{\perp} = \{w \in \mathbb{F}_q^n \mid \langle c, w \rangle = 0, \, \forall c \in C\},$$

onde $\langle c, w \rangle = \sum_{i=1}^{n} c_i w_i$ (soma sobre $\mathbb{F}_q$) é o *produto interno padrão.*

Um código C é chamado de **auto-ortogonal** se:

$$C \subseteq C^{\perp}.$$

A condição $C \subseteq C^{\perp}$ significa que todo vetor $c \in C$ é ortogonal, com respeito ao produto interno padrão, a todos os outros vetores $c' \in C$. Equivalentemente, para um código C ser auto-ortogonal, a seguinte propriedade deve ser satisfeita:

$$\langle c, c' \rangle = 0, \quad \forall c, c' \in C.$$

Isso inclui a auto-ortogonalidade de cada vetor consigo mesmo:

$$\langle c, c \rangle = 0, \quad \forall c \in C.$$

Exemplo 2.18. *(Código Auto-Ortogonal)*

Considere o código linear $C \subseteq \mathbb{F}_2^4$ definido como:

$$C = \{[0000], [1111], [1001], [0110]\}.$$

Para verificar se C é auto-ortogonal, calculamos o produto interno padrão:

$$\langle c, c' \rangle = \sum_{i=1}^{n} c_i c_i' \mod 2, \quad \textit{para todos } c, c' \in C.$$

Produto interno das palavras consigo mesmas

$$\begin{aligned}
&\langle [0000], [0000] \rangle = 0 \cdot 0 + 0 \cdot 0 + 0 \cdot 0 + 0 \cdot 0 = 0, \\
&\langle [1111], [1111] \rangle = 1 \cdot 1 + 1 \cdot 1 + 1 \cdot 1 + 1 \cdot 1 = 4 \equiv 0 \pmod 2, \\
&\langle [1001], [1001] \rangle = 1 \cdot 1 + 0 \cdot 0 + 0 \cdot 0 + 1 \cdot 1 = 2 \equiv 0 \pmod 2, \\
&\langle [0110], [0110] \rangle = 0 \cdot 0 + 1 \cdot 1 + 1 \cdot 1 + 0 \cdot 0 = 2 \equiv 0 \pmod 2.
\end{aligned}$$

Produto interno entre palavras diferentes

$$\begin{aligned}
&\langle [0000], [1111] \rangle = 0 \cdot 1 + 0 \cdot 1 + 0 \cdot 1 + 0 \cdot 1 = 0, \\
&\langle [0000], [1001] \rangle = 0 \cdot 1 + 0 \cdot 0 + 0 \cdot 0 + 0 \cdot 1 = 0, \\
&\langle [0000], [0110] \rangle = 0 \cdot 0 + 0 \cdot 1 + 0 \cdot 1 + 0 \cdot 0 = 0, \\
&\langle [1111], [1001] \rangle = 1 \cdot 1 + 1 \cdot 0 + 1 \cdot 0 + 1 \cdot 1 = 2 \equiv 0 \pmod 2, \\
&\langle [1111], [0110] \rangle = 1 \cdot 0 + 1 \cdot 1 + 1 \cdot 1 + 1 \cdot 0 = 2 \equiv 0 \pmod 2, \\
&\langle [1001], [0110] \rangle = 1 \cdot 0 + 0 \cdot 1 + 0 \cdot 1 + 1 \cdot 0 = 0.
\end{aligned}$$

*Como todos os produtos internos resultaram em 0, temos que $C \subseteq C^{\perp}$. Portanto, o código $C = \{[0000], [1111], [1001], [0110]\}$ é **auto-ortogonal**.*

Exemplo elaborado pela autora.

2.4.3 Códigos Auto-duais

Seja $C \subseteq \mathbb{F}_q^n$ um código linear de comprimento n. Dizemos que C é um **código auto-dual** se ele satisfizer as seguintes condições:

1. C é auto-ortogonal, ou seja, $C \subseteq C^{\perp}$, onde $C^{\perp}$ é o código dual de C.

2. $\dim(C) = \frac{n}{2}$, o que implica que $C = C^{\perp}$.

Exemplo 2.19. *(Código Auto-Dual)*

Considere o código binário $C \subseteq \mathbb{F}_2^4$*, dado pelas palavras:*

$$C = \{(0000), (1111), (1010), (0101)\}.$$

Verificação de Auto-Ortogonalidade

Para verificar se $C \subseteq C^{\perp}$*, calculamos a forma bilinear padrão* $\langle \mathbf{u}, \mathbf{v} \rangle = \sum_{i=1}^{n} u_i v_i \mod 2$ *para todas as combinações de* $\mathbf{u}, \mathbf{v} \in C$*:*

- *Para* $\mathbf{u} = (0000)$ *e qualquer* $\mathbf{v} \in C$*, temos:*

$$\langle (0000), \mathbf{v} \rangle = 0.$$

- *Para* $\mathbf{u} = (1111)$ *e* $\mathbf{v} = (1111)$*, temos:*

$$\langle (1111), (1111) \rangle = 1 \cdot 1 + 1 \cdot 1 + 1 \cdot 1 + 1 \cdot 1 \mod 2 = 4 \mod 2 = 0.$$

Para $\mathbf{v} = (1010)$*, temos:*

$$\langle (1111), (1010) \rangle = 1 \cdot 1 + 1 \cdot 0 + 1 \cdot 1 + 1 \cdot 0 \mod 2 = 2 \mod 2 = 0.$$

Para $\mathbf{v} = (0101)$*, temos:*

$$\langle (1111), (0101) \rangle = 1 \cdot 0 + 1 \cdot 1 + 1 \cdot 0 + 1 \cdot 1 \mod 2 = 2 \mod 2 = 0.$$

- *Para* $\mathbf{u} = (1010)$ *e* $\mathbf{v} = (1010)$*, temos:*

$$\langle (1010), (1010) \rangle = 1 \cdot 1 + 0 \cdot 0 + 1 \cdot 1 + 0 \cdot 0 \mod 2 = 2 \mod 2 = 0.$$

Para $\mathbf{v} = (0101)$*, temos:*

$$\langle (1010), (0101) \rangle = 1 \cdot 0 + 0 \cdot 1 + 1 \cdot 0 + 0 \cdot 1 \mod 2 = 0.$$

Como todos os produtos internos são zero, concluímos que $C \subseteq C^{\perp}$*.*

Verificação de Dimensão

O código C *contém* $2^2 = 4$ *palavras, o que implica* $\dim(C) = 2$*. Como* $n = 4$*, temos* $\dim(C) = \frac{n}{2}$*.*

Como $C \subseteq C^{\perp}$ *e* $\dim(C) = \dim(C^{\perp})$*, concluímos que* $C = C^{\perp}$*. Portanto,* C *é um código auto-dual.*

Exemplo elaborado pela autora.

3 Eficiência e Limites Geométricos

Neste capítulo, exploraremos a relação entre a geometria de espaços métricos e a eficiência na construção de códigos corretores de erros. A noção de bolas e esferas no espaço métrico nos ajudará a compreender como o processo de correção de erros funciona, além de fornecer a base para estruturas como códigos ótimos e perfeitos. Investigaremos também os limites teóricos que regulam a densidade e a capacidade de códigos, descritos pelas cotas de Hamming e Gilbert-Varshamov.

3.1 Bolas e esferas

Definição 3.1. *(Esfera)((MILIES, 2009), p.8).*

*Seja (F^n, d_H) o espaço métrico de vetores de comprimento n sobre o corpo F, munido da métrica de Hamming d_H. Para um vetor $x \in F^n$ e um inteiro não negativo $r \geq 0$, define-se a **esfera** centrada em x com raio r como:*

$$S(x, r) = \{y \in F^n \mid d_H(x, y) = r\}.$$

A esfera contém todas as palavras em F^n que estão a uma distância exatamente r de x.

Definição 3.2. *(Bola)((MILIES, 2009), p.8).*

*Para um vetor $x \in F^n$ e um inteiro não negativo $r \geq 0$, define-se a **bola** centrada em x com raio r como:*

$$B(x, r) = \bigcup_{t=0}^{r} S(x, t) = \{y \in F^n \mid d_H(x, y) \leq r\}.$$

A bola contém todas as palavras em F^n que estão a uma distância menor ou igual a r de x.

Podemos expressar essa quantidade de palavras por:

$$|B(x, r)| = \sum_{t=0}^{r} \binom{n}{t} (q-1)^t.$$

Teorema 3.3. *((MILIES, 2009), p.9).*

Seja C um código com distância mínima d e seja

$$\kappa = \left\lfloor \frac{d_{min} - 1}{2} \right\rfloor.$$

Então, é possível detectar até $d_{min} - 1$ erros e corrigir até κ erros.

Para provar o teorema, dividiremos a demonstração em duas partes. Primeiro, mostraremos que o código pode detectar até $d_{\min} - 1$ erros, onde $d_{\min}$ é a distância mínima do código, e depois mostraremos que ele pode corrigir até κ erros.

Demonstração: Para mostrar que C pode detectar até $d - 1$ erros, consideremos a seguinte situação:

Seja $y \in F^n$ uma palavra recebida que é uma perturbação de alguma palavra código $x \in C$, tal que $d_H(y, x) \leq d - 1$.

Como a distância mínima d do código C satisfaz

$$d = \min_{x \neq x'} d_H(x, x'), \quad \forall x, x' \in C,$$

segue que qualquer outra palavra $x' \in C$, $x' \neq x$, está a pelo menos d posições diferentes de x. Logo,

$$d_H(y, x') \geq d_H(x, x') - d_H(x, y) \geq d - (d - 1) = 1.$$

Consequentemente, qualquer palavra recebida y que esteja a até $d - 1$ erros de x pode ser identificada como contendo pelo menos 1 erro. Assim, C pode detectar até $d - 1$ erros.

Agora, consideremos as bolas de Hamming definidas como

$$B(x, r) = \{y \in F^n \mid d_H(x, y) \leq r\},$$

onde $x \in C$ e $r = \kappa$. A ideia é garantir que bolas de Hamming de raio κ centradas em palavras código são disjuntas.

Por definição, a distância mínima d garante que $d_H(x, x') \geq d$ para quaisquer $x, x' \in C$, $x \neq x'$.

Se considerarmos duas bolas $B(x, \kappa)$ e $B(x', \kappa)$ centradas em x e x', para $x \neq x'$, então, para haver interseção, seria necessário que

$$\exists y \in B(x, \kappa) \cap B(x', \kappa) \implies d_H(x, x') \leq 2\kappa.$$

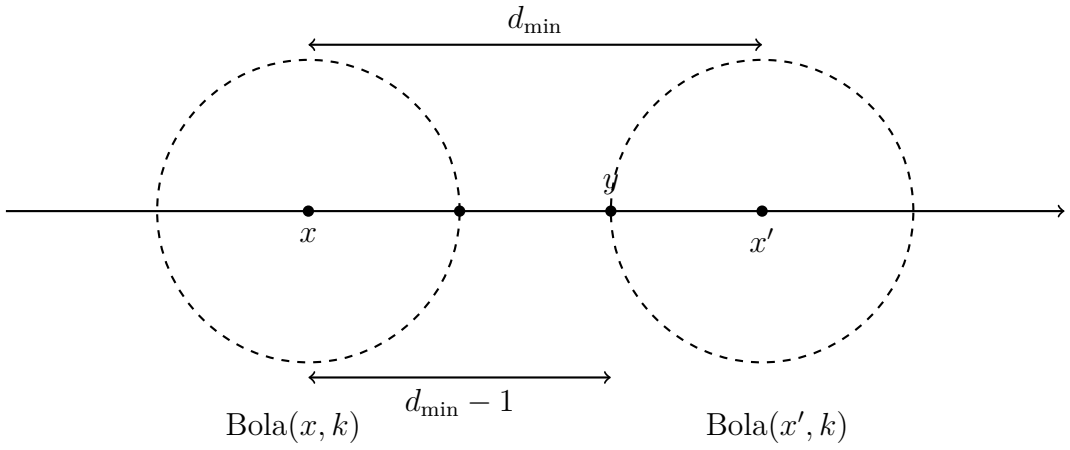
Como $\kappa = \left\lfloor \frac{d-1}{2} \right\rfloor$, temos que $2\kappa < d$, e, portanto, $B(x, \kappa) \cap B(x', \kappa) = \emptyset$. Isso implica que as bolas de Hamming de raio κ centradas em palavras código não se sobrepõem.

Assim, qualquer palavra $y \in F^n$ que esteja a uma distância κ de x pertence exclusivamente a $B(x, \kappa)$, o que permite corrigir até κ erros.

■

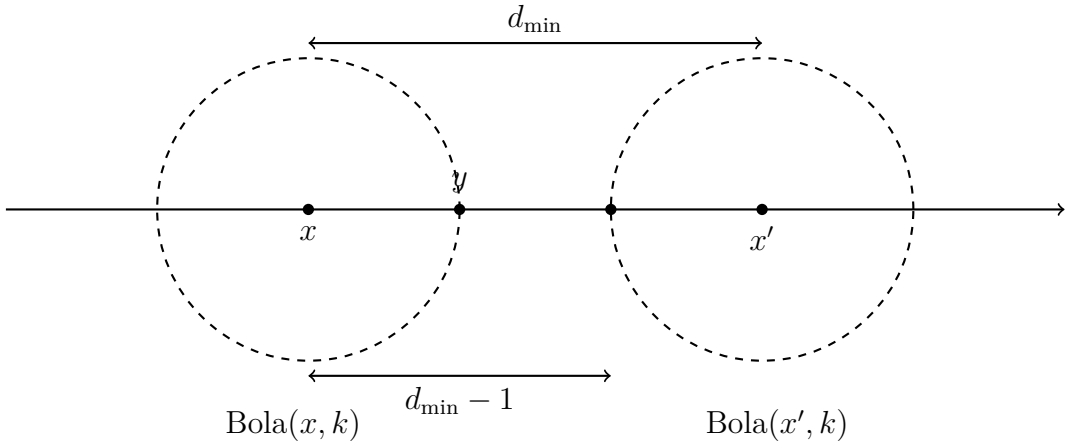
As figuras a seguir mostram dois casos diferentes que podem ocorrer em um código de Hamming, onde a distância mínima é $d_{\min} = 3$. Em ambos os casos, y é a palavra x que

Figura 3.1 – Bolas de raio κ centradas nas palavras x e x' de um código C, onde a palavra com erro y pertence a $B(x', \kappa)$.



Fonte: a autora

Figura 3.2 – Bolas de raio κ centradas nas palavras x e x' de um código C, onde a palavra com erro y pertence a $B(x, \kappa)$.



Fonte: a autora

foi recebida com erros, mas no caso da **Figura 3.1**, a quantidade t de erros é $t > k$. Isso faz com que a palavra y esteja mais próxima de x' do que da palavra correta x, assim ela pode ser detectada como contendo erros, pois é uma palavra que não pertence ao código, mas não pode ser corrigida, já que desse modo, ela seria corrigida para a palavra errada.

Já na **Figura 3.2**, temos $t \leq k$, fazendo assim com que y pertença à $B(x, \kappa)$, tornando o erro fácil de detectar e de corrigir.

Corolário 3.4 ((MILIES, 2009), p.13)**.** *Todas as esferas de raio r em A^n contêm o mesmo número de elementos.*

Demonstração: Seja A^n o espaço vetorial com elementos de comprimento n sobre um alfabeto $\mathbb{F}_q$ de q símbolos. Dado um ponto $x \in A^n$, queremos determinar o número de elementos na esfera $S(x, r)$ de raio r centrada em x.

Um ponto $y \in A^n$ estará a uma distância r de x se diferir de x em exatamente r posições. Para determinar o número de palavras y que satisfazem essa condição, seguimos os seguintes passos:

1. Escolhemos r posições entre as n posições disponíveis em x. Isso pode ser feito de $\binom{n}{r}$ maneiras.

2. Para cada uma das r posições escolhidas, o bit correspondente em y deve ser diferente do bit em x. Como o alfabeto contém q símbolos e um deles já está ocupado por x, restam $q-1$ opções para cada uma dessas posições. Assim, para r posições, há $(q-1)^r$ maneiras de preencher essas posições.

Portanto, o número total de pontos em $S(x, r)$ é dado por:

$$|S(x, r)| = \binom{n}{r}(q-1)^r.$$

A fórmula acima não depende do ponto x, mas apenas do raio r e do tamanho do alfabeto q. Isso implica que o número de elementos em uma esfera de raio r centrada em qualquer ponto $x \in A^n$ é o mesmo, o que demonstra o corolário.

■

3.2 Cota de Hamming

Teorema 3.5. *(Cota de Hamming)((MILIES, 2009), p.14).*

Seja C um código (n, M, d) sobre o alfabeto $\mathbb{F}_q$ com comprimento n, M palavras e distância mínima d. Denotando por $\kappa = \left\lfloor \frac{d-1}{2} \right\rfloor$ a capacidade de correção de erros de C, temos que:

$$M \leq \frac{q^n}{\sum_{t=0}^{\kappa} \binom{n}{t}(q-1)^t}.$$

Demonstração: Seja C um (n, M, d)-código em A^n, com $\kappa = \left\lfloor \frac{d-1}{2} \right\rfloor$, a capacidade de correção de erros do código. A soma das esferas de raio κ centradas nas palavras do código C cobre todas as palavras do código, além de todas as palavras que podem ser corrigidas para alguma palavra do código. Logo, temos:

$$\bigcup_{x \in C} B(x, \kappa) \subseteq A^n.$$

O número total de palavras em A^n é q^n, e cada esfera $B(x, \kappa)$ contém $\sum_{t=0}^{\kappa} \binom{n}{t}(q-1)^t$ palavras. Assim, a soma de todas as palavras contidas nas esferas é:

$$M \cdot \sum_{t=0}^{\kappa} \binom{n}{t}(q-1)^t,$$

onde M é o número de palavras do código C.

Como essas esferas estão contidas em A^n, segue que:

$$M \cdot (\sum_{t=0}^{\kappa} \binom{n}{t}(q-1)^t) \leq q^n.$$

Dividindo ambos os lados da desigualdade por $\sum_{t=0}^{\kappa} \binom{n}{t}(q-1)^t$, obtemos:

$$M \leq \frac{q^n}{\sum_{t=0}^{\kappa} \binom{n}{t}(q-1)^t}. \tag{3.1}$$

■

A **cota de Hamming** estabelece um limite superior para o número M de palavras código que um código (n, M, d) pode conter, dado o comprimento n, o tamanho do alfabeto q, e a distância mínima d.

3.2.1 Implicações:

A cota demonstra que, para um código com parâmetros fixos n, q, e d, existe um número máximo M de palavras que o código pode ter. Isso implica que não é possível construir um código arbitrariamente grande para uma distância mínima d, preservando sua capacidade de correção de erros.

Para aumentar d (e, consequentemente, a capacidade de correção de erros), é necessário reduzir o número M de palavras do código.

Observação 3.6. *Um código que atinge a igualdade na cota de Hamming é chamado de* ***código perfeito****, pois utiliza o espaço A^n de maneira ideal, sem sobreposição entre esferas e cobrindo o espaço de forma completa.*

Definição 3.7. *(Código Perfeito)((MILIES, 2009), p.14).*

Um código $C \subseteq A^n$, com distância mínima d e capacidade de correção de erros $\kappa = \left\lfloor \frac{d-1}{2} \right\rfloor$, é chamado de ***perfeito*** *se a união das bolas de raio κ, centradas em todas as palavras do código, cobre todo o espaço A^n. Em termos formais, temos:*

$$\bigcup_{x \in C} B(x, \kappa) = A^n.$$

Essa condição é chamada de **empacotamento de esferas**.

Em resumo, a cota de Hamming nos dá um limite superior para M, baseado na necessidade de evitar interseção entre as esferas.

Exemplo 3.8. *(Código de repetição binário de comprimento ímpar)*

Considere um código de repetição binário de comprimento ímpar n, onde $n = 2k+1$ e $k \in \mathbb{N}$. O código é construído repetindo cada bit da mensagem $k+1$ vezes. Para $n = 3$, o código é dado por:

$$C = \{000, 111\}.$$

A distância mínima do código é:

$$d = 3.$$

Como $d = 3$, o código pode corrigir até:

$$t = \left\lfloor \frac{d-1}{2} \right\rfloor = 1 \text{ erro.}$$

Para provar que o código é perfeito, mostramos que todas as palavras em $\mathbb{F}_2^3$ pertencem a uma esfera de raio $t = 1$ centrada em uma palavra de C.

1. O espaço $\mathbb{F}_2^3$ contém $2^3 = 8$ palavras.
2. Cada palavra do código possui uma esfera de raio 1, contendo:

- *A própria palavra (1 palavra).*
- *Todas as palavras que diferem por exatamente 1 bit (3 palavras).*

Total: $1 + 3 = 4$.

3. Como o código possui 2 palavras e cada esfera contém 4 palavras:

$$2 \times 4 = 8.$$

Portanto, o código cobre todo o espaço $\mathbb{F}_2^3$, mostrando que ele é perfeito.

Exemplo elaborado pela autora.

Definição 3.9. *(Código Ótimo)((MILIES, 2009), p.15).*

*Um código $C \subseteq A^n$ com parâmetros (n, M, d) é chamado de **ótimo** se o número de palavras M for igual a $A_q(n, d)$, onde $A_q(n, d)$ denota o tamanho máximo de um (n, M, d)-código q-ário. Em termos formais:*

$$M = A_q(n, d).$$

Isto significa que C possui a quantidade máxima de palavras possível dada sua distância mínima d. Além disso, C satisfaz a seguinte propriedade:

$$A^n \subseteq \bigcup_{c \in C} B(c, d_{\min} - 1),$$

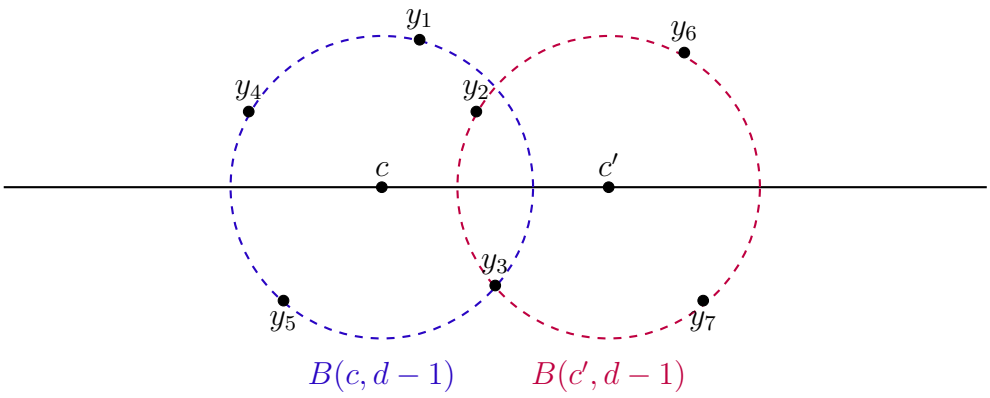
o que implica que:

$$q^n \leq \sum_{c \in C} |B(c, d_{\min} - 1)|. \tag{3.2}$$

Quando a igualdade é atingida, C é classificado como um código ótimo.

Observação 3.10. *O somatório da quantidade de elementos em todas as bolas de raio $d-1$ centradas em todas as palavras c pertencentes ao código, que se encontra nessa última expressão, só é possível de ser maior que q^n, pois o raio das bolas é $d-1 > \kappa$, o que implica que há interseção e algumas palavras são contadas mais de uma vez.*

Figura 3.3 – Bolas de raio $d-1$ centradas em duas palavras c e c' de um código C.



Fonte: a autora

3.3 Cota de Gilbert-Varshamov

Sabemos que o espaço A^n contém q^n palavras e que, em um código ótimo $C \subset A^n$, de tamanho $M = A_q(n, d)$, as bolas disjuntas de raio $\kappa = \lfloor (d-1)/2 \rfloor$ satisfazem:

$$q^n \geq M \cdot \sum_{t=0}^{\kappa} \binom{n}{t} (q-1)^t.$$

Assim, obtemos a cota de Hamming:

$$M \leq \frac{q^n}{\sum_{t=0}^{\kappa} \binom{n}{t} (q-1)^t}.$$

Por outro lado, ao considerarmos a equação (3.2), temos que:

$$q^n \leq M \cdot \sum_{t=0}^{d-1} \binom{n}{t} (q-1)^t.$$

Dividindo pelo somatório dos dois lados, obtemos a cota de Gilbert-Varshamov:

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

Teorema 3.11. *(Cota de Gilbert-Varshamov)((MILIES, 2009), p.15).*

Seja q o tamanho do alfabeto, n o comprimento do código e d a distância mínima. Então, existe um (n, M, d)-código q-ário com

$$M \geq \frac{q^n}{\sum_{t=0}^{d-1} \binom{n}{t} (q-1)^t},$$

Demonstração: Seja C um código (n, M, d)-q-ário. Seja $B(x, r)$ a bola de raio r centrada no ponto $x \in A^n$. A quantidade de palavras dentro de uma bola de raio r é dada por:

$$|B(x, r)| = \sum_{t=0}^{r} \binom{n}{t} (q-1)^t.$$

Agora, considerando as bolas de raio $d-1$ centradas nas palavras de C, temos que:

$$A^n \subseteq \bigcup_{c \in C} B(c, d-1).$$

Ou seja, a união das bolas de raio $d-1$ centradas nas palavras do código cobre todo o espaço A^n.

Se M for o número de palavras no código, o número total de palavras nas esferas centradas nas palavras de C é dado por:

$$M \cdot |B(c, d-1)| = M \cdot \sum_{t=0}^{d-1} \binom{n}{t} (q-1)^t.$$

Este é o número total de palavras cobrindo o espaço A^n, o qual deve ser pelo menos q^n, pois A^n contém q^n palavras. Logo, temos a desigualdade:

$$M \cdot \sum_{t=0}^{d-1} \binom{n}{t} (q-1)^t \geq q^n.$$

Como as bolas $B(c, d-1)$ não são disjuntas, isso significa que as palavras que podem ser corrigidas para uma palavra de C são contadas mais de uma vez. Isso ocorre

porque uma palavra que esteja dentro de duas ou mais bolas de raio $d-1$ centradas em palavras diferentes de C será contada mais de uma vez. Essa sobrecontagem é uma das razões pelas quais podemos obter uma limitação inferior para o número de palavras do código.

Dividindo dos dois lados pelo somatório, podemos deduzir a seguinte cota para M, o número de palavras do código:

$$M \geq \frac{q^n}{\sum_{t=0}^{d-1} \binom{n}{t}(q-1)^t}. \tag{3.3}$$

■

3.3.1 Estimativa de $A_q(n,d)$

Combinando as desigualdades (3.1) e (3.3), e levando em conta que $M = A_q(n,d)$ pois C é um código ótimo, temos:

$$\frac{q^n}{\sum_{t=0}^{d-1} \binom{n}{t}(q-1)^t} \leq A_q(n,d) \leq \frac{q^n}{\sum_{t=0}^{\kappa} \binom{n}{t}(q-1)^t}.$$

Essa expressão nos dá uma limitação superior e uma limitação inferior para o número máximo de palavras que um código pode ter, de modo que ele consiga detectar e corrigir o máximo possível de erros, sendo assim, o mais eficiente entre os códigos de sua classe $(n, M, d_{\min,\,\mathrm{k}})$.

3.4 Cota de Singleton

Definição 3.12. *((MILIES, 2009), p.16).*

Seja C um (n,M,d)-código q-ário. A cota de Singleton estabelece que o número máximo de palavras M em C satisfaz:

$$M \leq q^{n-d+1}.$$

Equivalentemente, podemos expressar a cota em termos da dimensão k de um código linear $[n,k,d]_q$:

$$k \leq n - d + 1.$$

A cota de Singleton representa um limite superior para o tamanho de um código ou sua dimensão, dado o comprimento n, a distância mínima d, e o tamanho do alfabeto q.

> Seja C um (n,M,d)-código ótimo; i.e., com $M = A_q(n,d)$. Afirmamos que, se c_1 e c_2 são duas palavras distintas de C, então as palavras c_1' e c_2', que

resultam destas eliminando as últimas $d-1$ posições, devem ser também distintas. De fato, se $c_1' = c_2'$, então c_1 e c_2 podem diferir apenas em posições que se encontram entre as $d-1$ que foram suprimidas. Isto significaria que $d(c_1, c_2) \leq d-1$, uma contradição.

Seja C' o código de comprimento $n-d+1$ que resulta de C encurtando todas suas palavras pela eliminação das últimas $d-1$ posições. O argumento acima mostra que $|C'| = |C|$. Como $C' \subseteq \mathbb{A}_q^{n-d+1}$, temos imediatamente que

$$A_q(n,d) = |C| \leq A_q(n-d+1).$$

(MILIES, 2009)

Comparação entre as cotas de Singleton e Hamming para $q \geq 4$:

A cota de Hamming é mais restritiva que a de Singleton apenas quando a soma no denominador da cota de Hamming cresce rapidamente em relação a q. No entanto, à medida que q aumenta, especialmente para $q \geq 4$, o termo q^{n-d+1} da cota de Singleton fornece uma limitação mais estrita.

Considere a cota de Hamming:

$$M \leq \frac{q^n}{\sum_{t=0}^{\lfloor (d-1)/2 \rfloor} \binom{n}{t} (q-1)^t}.$$

Para $q \geq 4$, o termo $(q-1)^t$ no denominador cresce rapidamente conforme t aumenta. Isso reduz o valor da soma no denominador em relação a q^n.

Na cota de Singleton:

$$M \leq q^{n-d+1}.$$

O valor de q^{n-d+1} é independente do crescimento de t, o que garante que essa limitação seja mais rigorosa para $q \geq 4$.

Exemplo 3.13. *Suponha $n = 10$, $d = 5$, $q = 4$:*

- Pela cota de Singleton:

$$M \leq 4^{10-5+1} = 4^6 = 4096.$$

- Pela cota de Hamming: O denominador é:

$$\sum_{t=0}^{\lfloor (5-1)/2 \rfloor} \binom{10}{t} (3)^t = \binom{10}{0}(3)^0 + \binom{10}{1}(3)^1 + \binom{10}{2}(3)^2.$$

Calculando:

$$\binom{10}{0}(3)^0 = 1, \ \binom{10}{1}(3)^1 = 30, \ \binom{10}{2}(3)^2 = 405.$$

Soma total:

$$1 + 30 + 405 = 436.$$

Logo:

$$M \leq \frac{4^{10}}{436} \approx 9552.3.$$

*Nesse caso, a cota de Singleton (*4096*) fornece uma limitação mais restrita do que a cota de Hamming (*9552.3*), mostrando ser superior para* $q \geq 4$.

Exemplo elaborado pela autora.

Referências

BETTEN MICHAEL BRAUN, H. F. A. K. A. K. A.; WASSERMANN, A. **Error-Correcting Linear Codes**. [S.l.]: Springer-Verlag Berlin Heidelberg, 2006. Citado 12 vezes nas páginas 14, 19, 25, 27, 30, 36, 38, 41, 42, 43, 47 e 49.

MILIES, C. P. Coloquio de matematica da região centro-oeste. In: **Introdução à Teoria dos codigos corretores de erros**. [S.l.: s.n.], 2009. Citado 10 vezes nas páginas 27, 31, 56, 58, 59, 60, 61, 63, 64 e 65.

NICOLETTI, E. R. **Aplicações de Algebra Linear aos Codigos Corretores de Erros e ao Ensino Medio**. Dissertação (Mestrado) — Universidade Estadual Paulista 'Julio de Mesquita Filho', 2015. Citado 2 vezes nas páginas 25 e 34.

NOLLI, D. **Codigos Lineares**. Dissertação (Mestrado) — Universidade Federal de Santa Catarina, 1985. Citado na página 23.