UNIVERSIDADE FEDERAL DE ALFENAS

HERNANDO BATISTA DA SILVA

Contribuições da Álgebra Linear para a Teoria dos Códigos Corretores de Erros

Alfenas/MG

2021

HERNANDO BATISTA DA SILVA

Contribuições da Álgebra Linear para a Teoria dos Códigos Corretores de Erros

Trabalho de Conclusão de Curso apresentado como parte dos requisitos para obtenção do título de Licenciado em Matemática pelo Instituto de Ciências Exatas da Universidade Federal de Alfenas. Área de concentração: Matemática Aplicada. Orientador: Anderson José de Oliveira.

Alfenas/MG

2021

HERNANDO BATISTA DA SILVA

Contribuições da Álgebra Linear para a Teoria dos Códigos Corretores de Erros

A Banca examinadora abaixo-assinada, aprova a Monografia apresentada como parte dos requisitos para obtenção do título de Licenciado em Matemática pelo Instituto de Ciências Exatas da Universidade Federal de Alfenas. Área de concentração: Matemática Aplicada.

Aprovado em: _____/_____/_____

Banca Examinadora:

Prof. Dr. Anderson José de Oliveira
Instituto de Ciências Exatas - UNIFAL-MG
Presidente

Profa. Dra. Cátia Regina de Oliveira Quilles Queiroz
Instituto de Ciências Exatas - UNIFAL-MG
Avaliador 1

Profa. Dra. Ângela Leite Moreno
Instituto de Ciências Exatas - UNIFAL-MG
Avaliador 2

Prof. Dr. José Carlos de Souza Júnior
Instituto de Ciências Exatas - UNIFAL-MG
Suplente

Agradecimentos

Agradeço primeiramente a Deus, por ter me capacitado para enfrentar este desafio. Agradeço aos meus professores, todos eles, sem exceção, por terem passado seus conhecimentos, não só os matemáticos, mas também os de vida, pois sei que cada um, a sua maneira, sempre quis o meu melhor. Em especial agradeço ao orientador deste trabalho papis Dr. Anderson José de Oliveira, mais conhecido como senhor, pela dedicação, apoio, amizade, e principalmente paciência, muita paciência, sempre lhe serei grato pelos seus ensinamentos. Agradeço aos meus amigos, por estarem juntos comigo nesta caminhada, que fizeram cada momento valer a pena, cada decepção, cada problema, cada nota baixa (que não foram poucas) foram aliviadas pela presença de vocês. Em especial aos meus amigos Natã e Matheus que sempre estiveram ao meu lado em todos os momentos e a minha amiga Paulinha Souza, minha irmãzinha do coração. Por último e não menos importante, agradeço minha família, que sem dúvida foi a que mais se abdicou para que pudesse chegar a esse momento. Minha esposa Etel, pelo apoio incondicional, pela paciência e principalmente por estar sempre ao meu lado, sendo meu suporte nos momentos difíceis, e meu filho Miguel, minhas desculpas por muitas vezes estar ausente, mas quero que saiba que você foi o maior motivo que tive para terminar esta jornada.

Resumo

No mundo moderno há evidente necessidade de aprimoramento dos sistemas de comunicação, principalmente na transmissão de informações digitais, resultantes do advento das novas tecnologias da informação e comunicação. Assim, no processo de transmissão de mensagens do emissor até o destinatário há interferências diversas que podem comprometer o conteúdo das mesmas. Para diminuir a probabilidade de erro na transmissão de informações digitais, são comumente utilizados os códigos corretores de erros e alguns desses códigos têm como base estrutural elementos da Álgebra Linear. Desta forma, o objetivo deste trabalho é identificar os elementos da Álgebra Linear que podem ser aplicados no processo de construção de códigos corretores de erros. Foram construídos os códigos $C(7,3)$ e $C(8,4)$, utilizando elementos de Álgebra Linear, com uma série de exemplos associados a esses códigos. Os resultados obtidos podem ser utilizados em diversar aplicações, como por exemplo no processo de geração de proteínas em um sistema de comunicação genético.

Palavras-chave: Transmissão da Informação. Ruídos. Matrizes. Vetores. Sistema de Comunicação.

Abstract

In the modern world, there is an evident need to improve communication systems, especially in the digital information transmission, resulting from the advent of new information and communication technologies. Thus, in the transmitting messages process from the sender to the receiver, there are several interferences that can compromise their content. In order to decrease the probability of error in the digital information transmission, error correction codes are commonly used end some of these codes are structurally based on Linear Algebra elements. Thus, the objective of this work is to identify the elements of Linear Algebra that can be applied in the process of error-correcting codes construction. The codes $C(7.3)$ and $C(8.4)$ were constructed by using elements of Linear Algebra, with a series of examples associated with these codes. The results obtained can be used in several applications, such as in the generating proteins process in a genetic communication system.

Keywords: Information Transmisssion. Noises. Matrices. Vectors. Communication System.

Sumário

INTRODUÇÃO

A necessidade de se comunicar está presente na sociedade desde os primórdios, um dos primeiros meios de comunicação conhecido, foi o das pinturas rupestres, sendo que se tem conhecimento de inscrições datadas de 3000 a.C. Ao longo do tempo, foram descobertos outros meios de comunicação, como por exemplo, os papiros, que datam até o século V d.C. No entanto, o estágio moderno da comunicação, foi a descoberta da tipografia pelo alemão Johann Gutemberg, em 1445, abrindo assim a era da comunicação social. Porém, o grande passo na comunicação social se deu em 1876, com a invenção do telefone pelo cientista Alexander Graham Bell. Em 1969, com a invenção da internet, se deu de fato origem a comunicação em massa a nível mundial.

Desde então a evolução nos sistemas de comunicação se deu em uma velocidade surpreendente, sendo que o ápice da comunicação social em massa se deu com a fusão das duas maiores invenções relacionadas à comunicação, o telefone e a internet, isso há apenas 24 anos.

Com tantas mudanças nas últimas décadas no que diz respeito a sistema de comunicação, houve também a preocupação de que as mensagens fossem transmitidas de forma segura e eficiente, pois a ocorrência de ruídos (erros) na transmissão da informação se tornaram cada vez mais um empecilho para os sistemas de comunicação. Para tentar minimizar esses erros na transmissão da informação, são utilizados os códigos corretores de erros, que são capazes de detectar e corrigir possíveis erros que podem ocorrer na transmissão de uma informação.

Esses códigos corretores de erros são utilizados há algum tempo, o primeiro deles, criado por Hamming em 1947, tinha a capacidade de detectar dois erros e corrigir um. Claro que nos dias atuais, os códigos tem capacidade de detectar e corrigir uma quantidade bem maior de erros, no entanto ainda é uma tecnologia que precisa ser aprimorada.

Segundo [1], o fato das comunicações digitais necessitarem de maior confiabilidade, somado com a ascensão do computador digital, como ferramenta essencial na sociedade tecnológica, os códigos corretores de erros têm ganhado papel fundamental para o aprimoramento da comunicação digital.

De acordo com [2], vale destacar também que os códigos corretores de erros são um dos grandes responsáveis pelo desenvolvimento e funcionamento de tecnologias que fazem parte do nosso dia-a-dia como, por exemplo, televisores, smartphones, computadores, música digital, internet entre tantas outras. Mais ainda, suas aplicações podem ser vistas nas mais diversas áreas da ciência, como por exemplo, na Engenharia Elétrica, na Biologia, na Computação Quântica e na Criptografia.

Para entendermos um pouco mais sobre os códigos corretores de erros, este trabalho detalha a construção de dois desses códigos, o código $C(7,3)$ e o código $C(8,4)$, no entanto, o foco principal é dar enfâse na importância da Álgebra Linear na construção dos mesmos.

Inicialmente, foram selecionados elementos da Álgebra Linear utilizados na construção dos códigos corretores de erros e suas respectivas definições, em seguida os principais elementos em um sistema de comunicação, além de apresentar a teoria acerca das construções dos códigos corretores de erros, por fim, foram construídos dois códigos corretores de erros, o $C(7,3)$ e o $C(8,4)$ com capacidade de detectar dois erros e corrigir um erro e com capacidade de detectar três erros e corrigir um erro, respectivamente.

Este trabalho está estruturado da seguinte forma: no Capítulo 1 são apresentadas as principais definições dos elementos da Álgebra Linear, no Capítulo 2 a teoria acerca dos Códigos Corretores de Erros, no Capítulo 3 são apresentadas as construções detalhadas dos códigos $C(7,3)$ e $C(8,4)$, apresentando as contribuições da Álgebra Linear nessas construções. Por fim, no Capítulo 4 são apresentadas as conclusões e perspectivas para trabalhos futuros.

1 *Elementos de Álgebra Linear*

Neste capítulo são apresentados os principais conceitos teóricos utilizados para a construção deste trabalho. Na Seção 1.1 serão apresentados conceitos sobre matrizes, nas Seções 1.2, 1.3 e 1.4 veremos as definições de grupos, anéis e corpos, respectivamente, na Seção 1.5 os principais resultados associados aos espaços vetoriais. As referências utilizadas foram: [3], [4], [5] e [6].

1.1 Matrizes

Matriz é uma tabela de elementos dispostos em linhas(m) e colunas(n).

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & ... & a_{1n} \\ a_{21} & a_{22} & a_{23} & ... & a_{2n} \\ . & . & . & . & . \\ . & . & . & . & . \\ . & . & . & . & . \\ a_{m1} & a_{m2} & a_{m3} & ... & a_{mn} \end{bmatrix}_{m\times n}$$

O conceito de matriz é muito utilizado em resolução de problemas, não porque ela ordena e simplifica o problema, mas sim por fornecer novos métodos de resolução.

As matrizes podem ser classificadas como:

- Matriz retangular: é uma matriz do tipo $m \times n$ (lê-se m por n) onde $m \neq n$. No exemplo a seguir temos uma matriz retangular 3×2.

$$A = \begin{bmatrix} 1 & 9 \\ 25 & 0 \\ 0 & 5 \end{bmatrix}_{3\times 2}.$$

- Matriz quadrada: é uma matriz cujo número de linhas é igual ao número de colunas, ou seja, $m = n$. A seguir será apresentado um exemplo de uma matriz quadrada 3×3.

$$A = \begin{bmatrix} 1 & 9 & 6 \\ 25 & 0 & 0 \\ 0 & 5 & 1 \end{bmatrix}_{3\times 3}.$$

- Matriz linha: é a matriz que possui apenas uma linha, denotada por $1 \times n$, conforme exemplo apresentado a seguir.

$$A = \begin{bmatrix} 1 & 9 & 6 \end{bmatrix}_{1\times 3}.$$

- Matriz coluna: é a matriz que possui apenas uma coluna, denotada por $m \times 1$, conforme exemplo apresentado a seguir.

$$A = \begin{bmatrix} 1 \\ 25 \\ 0 \end{bmatrix}_{3\times 1}.$$

- Matriz nula: é uma matriz quadrada ou não, onde todos os seus elementos são iguais a zero, a seguir uma matriz A 3×3 nula.

$$A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}_{3\times 3}.$$

- Matriz diagonal: é a matriz onde pelos menos um de seus elementos da diagonal principal é diferente de zero, e todos os elementos que não pertencem a diagonal principal são obrigatoriamente iguais a zero, conforme exemplo apresentado na matriz A a seguir.

$$A = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}_{3\times 3}.$$

- Matriz identidade: é a matriz cujos elementos da diagonal principal são todos iguais a 1, e todos os outros elementos iguais a zero, veja um exemplo apresentado a seguir.

$$A = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}_{3\times 3}.$$

- Matriz transposta: dada uma matriz $A = (a_{ij})_{m\times n}$, chama-se transposta de A a matriz $A^t = (a'_{ji})_{n\times m}$, tal que $a'_{ji} = a_{ij}$, para todo i e para todo j. Isto significa que, por exemplo $a'_{11}, a'_{21}, a'_{31}, \ldots, a'_{n1}$, são respectivamente iguais a $a_{11}, a_{12}, a_{13}, \ldots, a_{1n}$, sendo assim a primeira coluna de A^t é igual a primeira linha de A. Repetindo esse raciocínio, chegaríamos à conclusão de que as colunas de A^t são ordenadamente iguais as linhas de A. Por exemplo, se:

$$A = \begin{bmatrix} a & b & c \\ d & e & f \end{bmatrix}, \text{ então } A^t = \begin{bmatrix} a & d \\ b & e \\ c & f \end{bmatrix}.$$

Teorema 1.1.1 *A matriz transposta possui as seguintes propriedades:*

1. $(A^t)^t = A$ *para toda matriz* $A = (a_{ij})_{m\times n}$.

   ***Demonstração**: Fazendo* $(A^t)^t = ((a')'_{ij})_{m\times n}$ *resulta:* $(a')'_{ij} = a'_{ji} = a_{ij}$.

2. *Se* $A = (a_{ij})_{m\times n}$ *e* $B = (b_{ij})_{m\times n}$, *então* $(A + B)^t = A^t + B^t$.

   ***Demonstração**: Fazendo* $A+B = C = (c_{ij})_{m\times n}$ *e* $(A+B)^t = C^t = (c'_{ji})_{n\times m}$, *temos:* $c'_{ji} = c_{ij} = a_{ij} + b_{ij} = a'_{ji} + b'_{ji} = A^t + B^t$.

3. *Se* $A = (a_{ij})_{m\times n}$ *e* $K \in \mathbb{R}$, *então* $(KA)^t = KA^t$.

   ***Demonstração**: Fazendo* $(KA)^t = ((a')'_{ji})_{n\times m}$, *resulta:*
   $(a')'_{ji} = Ka_{ij} = Ka'_{ji} = KA^t$, *para todo* i, j.

4. *Se* $A = (a_{ij})_{m\times n}$ *e* $B = (b_{jk})_{n\times p}$, *então* $(AB)^t = A^t.B^t$.

   ***Demonstração**: Fazendo* $AB = C = (c_{ik})_{m\times p}$ *e* $(AB)^t = C^t = (c'_{ki})_{p\times m}$, *temos:* $c'_{ki} = c_{ik} = \sum_{j=1}^{n} a_{ij}b_{jk} = \sum_{j=1}^{n} b_{jk}a_{ij} = \sum_{j=1}^{n} b'_{kj}\ a'_{ji} = A^t.B^t$.

Matriz simétrica: é a matriz quadrada que é igual à sua transposta, ou seja, se trocarmos as linhas pelas colunas, a matriz continuará igual. A matriz A do tipo 3×3 a seguir é uma matriz simétrica.

$$A = \begin{bmatrix} 1 & 3 & 5 \\ 3 & 1 & 0 \\ 5 & 0 & 1 \end{bmatrix}_{3\times 3}.$$

Matriz antissimétrica: é toda matriz quadrada onde sua transposta é igual a sua [1]oposta, ou seja, $A^t = -A$. A seguir é apresentado um exemplo de uma matriz antissimétrica.

$$A = \begin{bmatrix} 0 & 7 & -1 \\ -7 & 0 & 4 \\ 1 & -4 & 0 \end{bmatrix}_{3\times 3}.$$

1.1.1 Igualdade de Matrizes

Duas matrizes A = $(a_{ij})_{m\times n}$ e B = $(b_{ij})_{m\times n}$ são iguais quando $a_{ij} = b_{ij}$ para todo $i \in \{1, 2, 3, 4, ..., m\}$ e para todo $j \in \{1, 2, 3, 4, ..., n\}$. As matrizes A e B a seguir são iguais.

$$A = \begin{bmatrix} -6 & 3 & 5 \\ 3 & 1 & 0 \\ 5 & 0 & 7 \end{bmatrix}_{3\times 3} \quad eB = \begin{bmatrix} -6 & 3 & 5 \\ 3 & 1 & 0 \\ 5 & 0 & 7 \end{bmatrix}_{3\times 3}.$$

Note que para a_{ij} e b_{ij} com $i = 1$ e $j = 1$, temos $a_{11} = -6 = b_{11}$, isto deve se repetir para todos os outros elementos com o mesmo índice "ij".

[1] Se a soma entre duas matrizes resultar em uma matriz nula, temos que as matrizes são opostas, ou seja a matriz oposta de A é a matriz $-A$.

1.1.2 Soma de Matrizes

Dadas duas matrizes A $= (a_{ij})_{m\times n}$ e B $= (b_{ij})_{m\times n}$, chama -se soma $A+B$, a matriz $C = (c_{ij})_{m\times n}$, tal que $(c_{ij}) = (a_{ij}) + (b_{ij})$ para todo i e para todo j. Isto significa que a soma de duas matrizes $A + B$ do tipo $m \times n$, resulta em uma matriz C também $m \times n$, e cada elemento é a soma dos elementos correspondentes de A e B.

Teorema 1.1.2 *A adição de matrizes do tipo $m \times n$, possui as seguintes propriedades:*

- *Associativa: $(A + B) + C = A + (B + C)$ para quaisquer A, B e C do tipo $m \times n$.*

  ***Demonstração**: Fazendo $(A + B) + C = X$ e $A + (B + C) = Y$, temos:*

  $x_{ij} = (a_{ij} + b_{ij}) + c_{ij} = a_{ij} + (b_{ij} + c_{ij}) = y_{ij}, \forall i, j$, logo $X = Y$.

- *Comutativa: $A + B = B + A$, quaisquer que sejam A e B do tipo $m \times n$.*

  ***Demonstração**: Fazendo $A + B = X$ e $B + A = Y$, temos que, $x_{ij} = a_{ij} + b_{ij} = b_{ij} + a_{ij} = y_{ij}$.*

  Logo, $x_{ij} = y_{ij}$, portanto, $A + B = B + A$.

- *Elemento neutro: Existe uma matriz M tal que $M + A = A + M = A$, qualquer que seja a matriz A do tipo $m \times n$.*

  ***Demonstração**: Seja $A + M = A$, então $a_{ij} + m_{ij} = a_{ij}$, assim $m_{ij} = 0 \Rightarrow M = 0$. Portanto o elemento neutro é a matriz nula do tipo $m \times n$.*

- *Elemento simétrico: Para toda matriz A do tipo $m \times n$, existe A' tal que $A + A' = A' + A = M$.*

  ***Demonstração**: Se $A + A' = M = 0$ então $a_{ij} + a'_{ji} = 0$, logo $a_{ij} = -a'_{ji}, \forall i, j$, ou seja, a simétrica da matriz $A_{m\times n}$. Para a adição é a matriz $A'_{m\times n}$, onde os elementos de A' são os opostos dos elementos de A.*

Definição 1.1.1 *Dadas duas matrizes $A = (a_{ij})_{m\times n}$ e $B = (b_{ij})_{m\times n}$ chama-se diferença $A - B$, a matriz soma de A com a oposta de B. Por exemplo, sejam:*

$$A = \begin{bmatrix} -6 & 3 & 5 \\ 3 & 1 & 0 \\ 5 & 0 & 7 \end{bmatrix} \text{ e } B = \begin{bmatrix} -4 & 2 & 9 \\ 1 & 0 & 4 \\ 7 & -3 & 2 \end{bmatrix}.$$

Temos que $A - B = A + (-B)$, assim:

$$A - B = \begin{bmatrix} -6 & 3 & 5 \\ 3 & 1 & 0 \\ 5 & 0 & 7 \end{bmatrix} + \begin{bmatrix} 4 & -2 & -9 \\ -1 & 0 & -4 \\ -7 & 3 & -2 \end{bmatrix} = \begin{bmatrix} -2 & 1 & -4 \\ 2 & 1 & -4 \\ -2 & 3 & 5 \end{bmatrix}.$$

1.1.3 Produto de Matrizes

Dadas duas matrizes $A = (a_{ij})_{m\times n}$ e $B = (b_{jk})_{n\times p}$, chama-se produto AB a matriz $C = (c_{ik})_{m\times p}$, tal que:

$c_{ik} = a_{i1}.b_{1k} + a_{i2}.b_{2k} + \ldots + a_{in}.b_{nk} = \sum_{j=1}^{n} a_{ij}b_{jk}$ para todo $i \in \{1,2,3,\ldots,m\}$ e todo $k \in \{1,2,3,\ldots,p\}$.

Teorema 1.1.3 *A multiplicação de matrizes possui as seguintes propriedades:*

- *Associativa:* $(AB)C = A(BC)$ *quaisquer que sejam as matrizes*

  $A = (a_{ij})_{m\times n}$, $B = (b_{jk})_{n\times p}$ *e* $C = (c_{kl})_{p\times r}$.

  ***Demonstação**: Fazendo* $D = AB = (d_{ik})_{m\times p}$, $E = (AB)C = (e_{il})_{m\times r}$ *e* $F = BC = (f_{jl})_{n\times r}$, *temos:*

  $e_{il} = \sum_{k=1}^{p} d_{ik}.c_{kl} = \sum_{k=1}^{p}(\sum_{j=1}^{n} a_{ij}.b_{jk}).c_{kl} = \sum_{k=1}^{p}(\sum_{j=1}^{n} a_{ij}.b_{jk}.c_{kl}) =$

  $= \sum_{j=1}^{n} a_{ij}.(\sum_{k=1}^{p} b_{jk}.c_{kl}) = \sum_{j=1}^{n} a_{ij}.f_{jl}$, *então* $(AB)C = A(BC)$.

- *Distributiva à direita:* $(A + B)C = AC + BC$, *quaisquer que sejam as matrizes* $A = (a_{ij})_{m\times n}$, $B = (b_{ij})_{m\times n}$ *e* $C = (c_{jk})_{n\times p}$.

  ***Demonstração**: Fazendo* $D = (A + B)C = (d_{ik})_{m\times p}$, *temos:*

  $d_{ik} = \sum_{j=1}^{n}(a_{ij} + b_{ij}).c_{jk} = \sum_{j=1}^{n}(a_{ij}.c_{jk} + b_{ij}.c_{jk}) = \sum_{j=1}^{n} a_{ij}.c_{jk} + \sum_{j=1}^{n} b_{ij}.c_{jk}$, *então* $(A + B)C = AC + BC$.

- *Distributiva à esquerda:* $C(A + B) = CA + CB$, *quaisquer que sejam as matrizes* $A = (a_{ij})_{m\times n}, B = (b_{ij})_{m\times n}$ *e* $C = (c_{ki})_{p\times m}$.

  ***Demonstração**: Fazendo* $D = C(A + B) = (d_{kj})_{p\times n}$, *temos:*

  $d_{kj} = \sum_{j=1}^{n} c_{ki}(a_{ij} + b_{ij}) = \sum_{j=1}^{n}(c_{ki}.a_{ij} + c_{ki}.b_{ij}) = \sum_{j=1}^{n} c_{ki}.a_{ij} + \sum_{j=1}^{n} c_{ki}.b_{ij}$, *então* $C(A + B) = CA + CB$.

Observação: É importante ressaltar que a multiplicação de matrizes não possui a propriedade comutativa, isto é, para duas matrizes quaisquer, nem sempre $AB = BA$, como veremos a seguir:

Sejam:

$$A = \begin{bmatrix} 2 & 3 \\ 1 & 2 \end{bmatrix} \text{ e } B = \begin{bmatrix} 1 & 0 \\ 2 & 4 \end{bmatrix}.$$

Note que

$$A.B = \begin{bmatrix} 2 & 3 \\ 1 & 2 \end{bmatrix} . \begin{bmatrix} 1 & 0 \\ 2 & 4 \end{bmatrix} = \begin{bmatrix} 8 & 12 \\ 5 & 8 \end{bmatrix}$$

Por sua vez

$$B.A= \begin{bmatrix} 1 & 0 \\ 2 & 4 \end{bmatrix} . \begin{bmatrix} 2 & 3 \\ 1 & 2 \end{bmatrix} = \begin{bmatrix} 2 & 3 \\ 8 & 14 \end{bmatrix}$$

Pode-se observar que

$$\begin{bmatrix} 8 & 12 \\ 5 & 8 \end{bmatrix} \neq \begin{bmatrix} 2 & 3 \\ 8 & 14 \end{bmatrix}$$

.

1.2 Grupos

O conceito da estrutura de grupo é a parte central da teoria de códigos corretores de erros. Esta estrutura tem papel fundamental na codificação, decodificação, análise de desempenho, etc.

Definição 1.2.1 *Um grupo é um conjunto $G \neq \emptyset$ com uma operação $(*)$ onde:*

$$G \times G \to G$$
$$(a, b) \mapsto a * b$$

munido das seguintes propriedades:

1. A operação deve ser associativa, ou seja,

   $a * (b * c) = (a * b) * c$, $\forall a, b, c \in G$.

2. $\exists$ o elemento neutro e para a aperação $(*)$, ou seja,

   $e * a = a = a * e$, $\forall a \in G$.

3. $\forall a \in G, \exists\ b$ (elemento inverso), tal que:

   $a * b = e = b * a$, $\forall a, b \in G$.

Quando a operação também é comutativa, ou seja, $a * b = b * a$, $\forall a, b \in G$, então o grupo é chamado de grupo comutativo ou grupo abeliano.

1.3 Anéis

Definição 1.3.1 *Um anel comutativo é uma estrutura constituída de um conjunto A munido de duas operações binárias.*

$$+ : A \to A \qquad e \qquad \cdot : A \to A$$
$$(a, b) \mapsto a + b \qquad\qquad (a, b) \mapsto a.b$$

que chamaremos respectivamente de adição e multiplicação, com as seguintes propriedades:

1. *Associativa da adição:* $\forall a, b, c \in A, (a + b) + c = a + (b + c)$.

2. *Existência de elemento neutro para a adição:* $\exists\ 0 \in A$, *tal que* $a + 0 = 0 + a = a, \forall a \in A$.

3. *Existência de elemento inverso para a adição:* $\forall a \in A, \exists -a \in A$ *tal que* $a + (-a) = 0 = (-a) + a$.

4. *Comutatividade da adição:* $\forall a, b \in A, a + b = b + a$.

5. *Associatividade da multiplicação:* $\forall a, b, c \in A, (a.b).c = a.(b.c)$.

6. *Existência de elemento neutro para a multiplicação: Existe* $1 \in A - \{0\}$, *tal que* $a.1 = 1.a = a, \forall a \in A$.

7. *Comutatividade da multiplicação:* $\forall a, b \in A, a.b = b.a$.

8. *Distributividade da multiplicação com relação à adição, à direita e à esquerda:* $\forall a, b, c \in A, a.(b + c) = a.b + a.c$ *e* $(b + c).a = b.a + c.a$.

1.4 Corpos

Definição 1.4.1 *Seja F um conjunto de elementos sobre o qual duas operações binárias, adição $(+)$ e a multiplicação $(.)$, são definidas. F com as duas operações binárias, é um corpo se as seguintes condições são satisfeitas:*

1. *F é um grupo comutativo sob $(+)$. O elemento identidade é o 0 (zero).*

2. *O conjunto dos elementos não zero em F é um grupo comutativo sob $(.)$. O elemento identidade é o 1 (um).*

3. *A multiplicação é distributiva sob a adição à direita e à esquerda, isto é, para quaisquer a, b, c em F, temos:*

$$a.(b + c) = a.b + a.c$$
$$(b + c).a = b.a + c.a$$

O número de elementos em um corpo é chamado de ordem do corpo. Um corpo com números finitos de elementos é chamado de **corpo finito**.

Importante fazermos algumas observações:

- Em um corpo, o inverso aditivo de um elemento a é denotado por $-a$, e o inverso multiplicativo de a é denotado por $a^{-}1$, desde que $a \neq 0$.

- Subtração de um corpo: a subtração de um elemento a por um outro elemento b, ambos pertencentes a um corpo, é definida como:
$$a - b \triangleq a + (-b)$$

- Divisão em um corpo:a divisão de um elemento a por um elemento não zero b, ambos pertencentes a um corpo, é definida como:
$$a \div b \triangleq a.b^{-}1$$

Definição 1.4.2 *Um corpo com um número finito de elementos, representado por $\mathbb{F}_p$, sendo p um número primo, é chamado de* ***corpo de Galois****.*

Exemplo 1.4.1 *considere o conjunto $\{0, 1\}$ cujas operações de adição e multiplicação módulo-2 são apresentadas nas tabelas 1 e 2.*

+	0	1
0	0	1
1	1	0

Tabela 1 – Operação de adição módulo-2

.	0	1
0	0	0
1	0	1

Tabela 2 – Operação de multiplicação módulo-2

Este corpo é usualmente chamado de *corpo binário* e é denotado por $\mathbb{F}_2$. Este corpo desempenha um papel importante na teoria de códigos e é amplamente usado em computadores digitais e sistemas de transmissão ou armazenamentos de dados digitais.

1.5 Espaços Vetoriais

Um espaço vetorial é um conjunto V munido das operações de soma de vetores e de multiplicação por escalar e que satisfazem as seguintes propriedades usuais dos espaços $\mathbb{R}_n$.

- Soma entre vetores que satisfaz:

  1. Associatividade: $(u + v) + w = u + (v + w)$, para quaisquer $u, v, w \in V$;
  2. Elemento neutro: Existe o vetor $0 \in V$ que satisfaz $v + 0 = 0 + v = v$, para qualquer $v \in V$;

3. Inverso aditivo: Para cada $v \in V$, existe o inverso $u = -v \in V$, que satisfaz $v + u = 0 = u + v$;
4. Comutatividade: $u + v = v + u$, para quaisquer $u, v \in V$;

- Multiplicação de vetor por escalar que satisfaz:

1. Associatividade da multiplicação por escalar: $a.(b.v) = (a.b).v$, para quaisquer $a, b \in \mathbb{R}$ e qualquer $v \in V$;
2. Elemento neutro: $1.v = v = v.1$, para todo $v \in V$;
3. Distributiva de um escalar em relação à soma de vetores: $a.(u + v) = a.u + a.v$, para qualquer $a \in \mathbb{R}$ e quaisquer $u, v \in V$;
4. Distributiva da soma de escalares em relação a um vetor: $(a + b).v = a.v + b.v$, para quaisquer $a, b \in \mathbb{R}$ e qualquer $v \in V$.

1.5.1 Combinação Linear

O espaço vetorial possui uma característica muito importante, que é a obtenção de novos vetores a partir de vetores dados, de acordo com as operações e proposições do espaço vetorial. Este processo é chamado de combinação linear.

1.5.2 Subespaços Vetoriais

Sejam V um espaço vetorial sobre $\mathbb{R}$ e W um subconjunto de V. Dizemos que W é um subespaço vetorial de V se as seguintes condições são satisfeitas:

1. $W \neq \emptyset$;
2. $u + v \in W$, para todos $u, v \in W$;
3. $\alpha.u \in W$, para todos $\alpha \in \mathbb{R}$ e $u \in W$.

Teorema 1.5.1 *Seja S um conjunto finito de elementos de um espaço vetorial V. O conjunto de todas as combinações lineares dos vetores de S, denotado por $[S]$, forma um subespaço vetorial de V.*

Demonstração: Seja $S = \{v_1, v_2, \ldots, v_n\}$ um conjunto de n elementos de V. Vamos verificar que valem as condições de subespaço vetorial para $[S]$:

1. O elemento neutro de V está em $[S]$, pois basta tomar todas as constantes α_i nulas, assim o resultado da combinação linear é o elemento neutro do espaço vetorial V;

2. Considere $u, w \in [S]$. Se $u \in [S]$, então $u = \alpha_1 v_1 + \alpha_2 v_2 + \cdots + \alpha_n v_n$. E se $w \in [S]$, então $w = \beta_1 v_1 + \beta_2 v_2 + \cdots + \beta_n v_n$. Temos que $u + w = (\alpha_1 v_1 + \alpha_2 v_2 + \cdots + \alpha_n v_n) + (\beta_1 v_1 + \beta_2 v_2 + \cdots + \beta_n v_n) = (\alpha_1 + \beta_1)v_1 + (\alpha_2 + \beta_2)v_2 + \cdots + (\alpha_n + \beta_n)v_n$. Como $(\alpha_i + \beta_i) \in \mathbb{R}$, temos que $u + w$ é também combinação linear dos elementos de S, logo $u + w \in [S]$;

3. Considere $u \in [S]$ e um escalar $\beta \in \mathbb{R}$. Se $u \in [S]$, então $u = \alpha_1 v_1 + \alpha_2 v_2 + \cdots + \alpha_n v_n$. Assim temos que $\beta u = \beta(\alpha_1 v_1 + \alpha_2 v_2 + \cdots + \alpha_n v_n) = \beta\alpha_1 v_1 + \beta\alpha_2 v_2 + \cdots + \beta\alpha_n v_n$. Como $\beta\alpha_i \in \mathbb{R}$, temos que βu é também combinação linear dos elementos de S, logo $\beta u \in [S]$.

   Assim, provamos que $[S]$ é um subespaço vetorial de V.

1.5.3 Vetores Linearmente Dependentes

Se qualquer um dos vetores de um subespaço vetorial for combinação linear dos demais, este conjunto de vetores é chamado linearmente dependente (LD).

1.5.4 Vetores Linearmente Independentes

É o conjunto de vetores de um subespaço vetorial qualquer, onde nenhum deles é combinação linear dos demais (LI).

1.5.5 Base

É um conjunto de vetores linearmente independentes, tal que este conjunto gera todo o espaço vetorial, ou seja, todos os outros vetores do subespaço são combinações lineares desses vetores.

Teorema 1.5.2 *Seja V um espaço vetorial e $\{v_1, v_2, \ldots, v_n\}$ um conjunto de elementos que geram V. Então, dentre esses elementos podemos extrair uma base para V.*

Demonstração: Se o conjunto $\{v_1, v_2, \ldots, v_n\}$ for LI, então por definição já é uma base para V e não temos nada a fazer.
Agora se o conjunto é LD, então existem escalares $\alpha_1, \alpha_2, \ldots, \alpha_n$ não todos nulos,tais que:

$$\alpha_1 v_1 + \alpha_2 v_2 + \cdots + \alpha_n v_n = e.$$

com e o elemento neutro de V.
Considere, por exemplo, que $\alpha_n \neq 0$, então podemos dividir a equação por α_n e isolar v_n.

$$\tfrac{\alpha_1}{\alpha_n} v_1 + \cdots + \tfrac{\alpha_{n-1}}{\alpha_n} + v_n = e \Rightarrow v_n = -\tfrac{\alpha_1}{\alpha_n} v_1 - \cdots - \tfrac{\alpha_{n-1}}{\alpha_n}.$$

Assim, v_n é uma combinação linear dos demais elementos. Pela propriedade de dependência linear, podemos extrair v_n do conjunto que ele continua gerando V. Fazendo

este mesmo processo uma quantidade finita de vezes, obteremos um subconjunto de $\{v_1, v_2, \ldots, v_n\}$ formado por r elementos LI ($r \leq n$) que ainda geram V,ou seja, formam uma base para V.

1.5.6 Dimensão

Corolário 1.5.1 *Qualquer base de um espaço vetorial V tem sempre o mesmo número de elementos. Esse número é chamado de dimensão de V e é denotado por $dim(V)$.*

Teorema 1.5.3 *Seja um espaço vetorial V gerado por um conjunto finito de vetores $\{v_1, v_2, \ldots, v_n\}$. Então, qualquer conjunto com mais de n vetores é necessariamente LD e, portanto qualquer conjunto LI tem no máximo n vetores.*

Demonstração: Considere um subconjunto de V com m elementos $W = \{w_1, w_2, \ldots w_m\}$, com $m > n$. Vamos mostrar que w é LD. Assim qualquer conjunto LI possui no máximo n elementos.
Como $\{v_1, v_2, \ldots, v_n\}$ gera V, pelo Teorema 1.5.2 podemos extrair desse conjunto uma base para V. Seja $\{v_1, v_2, \ldots, v_r\}$ esta base. Então existem escalares α_{ij}, com $i = \{1, 2, 3, \ldots, r\}$ e $j = \{1, 2, 3, \ldots, m\}$ tais que:

$$w_j = \alpha_{1j}v_1 + \alpha_{2j}v_2 + \cdots + \alpha_{ij}v_i + \cdots + \alpha_{rj}v_r. \tag{1.1}$$

Consideremos agora a combinação linear nula de $w_1, w_2, \ldots, w_m$:

$$\beta_1 w_1 + \beta_2 w_2 + \cdots + \beta_m w_m = e. \tag{1.2}$$

Substituindo as relações de (1.1) em (1.2) obtemos:

$\beta_1(\alpha_{11}v_1 + \alpha_{21}v_2 + \cdots + \alpha_{r1}v_r) + \cdots + \beta_m(\alpha_{1m}v_1 + \alpha_{2m}v_2 + \cdots + \alpha_{rm}v_r) = e \Rightarrow$
$\Rightarrow(\beta_1\alpha_{11} + \beta_2\alpha_{12} + \cdots + \beta_m\alpha_{1m})v_1 + \cdots + (\beta_1\alpha_{r1} + \beta_2\alpha_{r2} + \cdots + \beta_m\alpha_{rm})v_r = e.$
Como $\{v_1, v_2, \ldots, v_r\}$ é uma base para V,então este conjunto é LI. Assim, temos:

$$\begin{cases} \beta_1\alpha_{11} + \beta_2\alpha_{12} + \cdots + \beta_m\alpha_{1m} = 0 \\ \qquad\qquad\qquad\qquad\qquad\qquad\quad \vdots \\ \beta_1\alpha_{r1} + \beta_2\alpha_{r2} + \cdots + \beta_m\alpha_{rm} = 0 \end{cases}$$

Obtemos um sistema linear homogêneo com r equações e m incógnitas $\beta_1, \beta_2, \ldots, \beta_m$. E como $r \leq n < m$, o sistema admite soluções não triviais. Assim, existem escalares não todos nulos $\beta_1, \beta_2, \ldots, \beta_m$, tais que:

$$\beta_1 w_1 + \beta_2 w_2 + \cdots + \beta_n w_n = e.$$

Portanto, $W = \{w_1, w_2, \ldots, w_m\}$ é LD. Assim, qualquer conjunto com mais de n elementos é LD, ou seja, qualquer conjunto LI possui no máximo n elementos.

Teorema 1.5.4 *Qualquer conjunto de vetores LI de um espaço vetorial V de dimensão finita pode ser completado de modo a formar uma base de V.*

Demonstração: Seja $dim(V) = n$ e $v_1, v_2, \ldots, v_r$ elementos LI em V. Pelo Teorema 1.5.3, $r \leq n$. Se os elementos $v_1, v_2, \ldots, v_r$ geram V, então $\{v_1, v_2, \ldots, v_r\}$ já é uma base.
Agora, se isso não ocorre, então existe um $v_{r+1} \in V$ que não é combinação linear de $v_1, v_2, \ldots, v_r$ então $\{v_1, v_2, \ldots, v_r\}$ ainda é LI, pois caso contrário a equação:

$$\alpha_1 v_1 + \alpha_2 v_2 + \cdots + \alpha_r v_r + \alpha_{r+1} v_{r+1} = e$$

seria verdadeira para $\alpha_{r+1} \neq 0$ e, então poderíamos dividir a equação por α_{r+1} e isolar v_{r+1}, escrevendo ele como uma combinação linear de $\{v_1, v_2, \ldots, v_r\}$, o que é uma contradição. Se $\{v_1, v_2, \ldots, v_r, v_{r+1}\}$ gera V, então é uma base. Caso contrário, repetimos o mesmo processo até completar a base. Esse processo termina em um número finito de passos,uma vez que pelo Teorema 1.5.3 não podemos ter um conjunto LI com mais de n elementos, já que $dim(V) = n$.

2 Códigos Corretores de Erros

Neste capítulo veremos as definições dos elementos que são importantes para um sistema de comunicação. Na Seção 2.1 serão apresentados os elementos importantes em um sistema de comunicação, nas Seções 2.2 e 2.3 serão apresentadas as definições de matriz geradora e matriz verificadora de paridade, respectivamente, na seção 2.4 veremos o que é peso de Hamming, na Seção 2.5 será apresentado como se calcula a distância mínima de um código de bloco linear, na Seção 2.6 as capacidades de correção e detecção de erros de um código, na Seção 2.7 a definição de síndrome de erros, na Seção 2.8 veremos o que é um código perfeito, na Seção 2.9 será apresentado como é feita a correção de erros pela síndrome, por fim na Seção 2.10 apresentaremos o conceito de arranjo padrão. As referências utilizadas foram: [3], [4], [5] e [6].

2.1 Elementos Importantes em um Sistema de Comunicação

A função de um sistema de comunicação é processar e transportar informações desde a origem até o destinatário, ou seja, é o processo no qual uma mensagem é enviada por um emissor por meio de um determinado canal até a mesma ser recebida por um receptor, no entanto, neste processo pode haver a interferência de ruídos. O objetivo do sistema de comunicação é detectar e corrigir esses ruídos, para que a mensagem recebida seja a mais confiável possível. A seguir serão apresentados os elementos fundamentais de um sistema de comunicação, ilustrados na Figura 1.

- Fonte de informação: produz a mensagem que será enviada.
- Transmissor: possibilita enviar a mensagem com um sinal adequado.
- Canal: meio utilizado para enviar a mensagem do transmissor para o receptor.
- Receptor: realiza as ações inversas ao que foi feito pelo transmissor, reconstruindo a mensagem.
- Destino: para quem/onde a mensagem foi destinada.

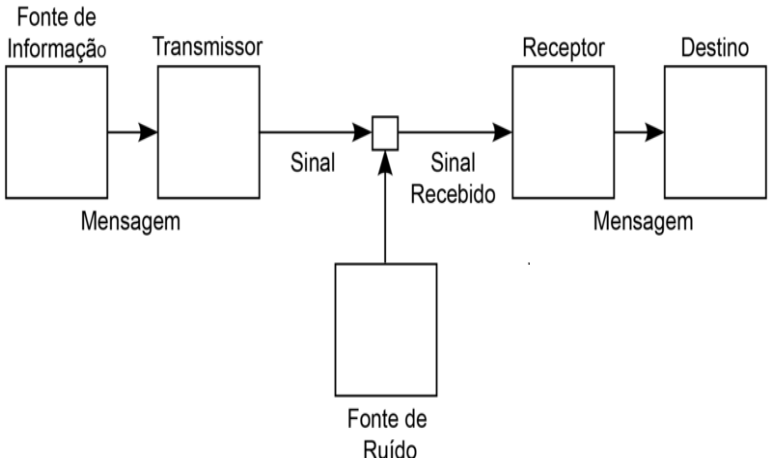

Figura 1 – Elementos de um Sistema de Comunicação.

2.1.1 Codificação

Definição 2.1.1 *Código é um conjunto de símbolos usado na transmissão e recepção de mensagens. Os elementos básicos para se construir um código são o alfabeto e as palavras-código.*

Alfabeto é um conjunto finito de elementos. Cada um desses elementos chama-se dígito ou bit. Quando o número de elementos do alfabeto é q, diz-se que o código é q-ário, as palavras-código representam uma sequência finita de dígitos. O número de dígitos de uma palavra-código é o seu comprimento.

Definição 2.1.2 *Codificação é a conversão de uma dada sequência de dígitos, originados de um alfabeto fonte, em outra, originada de um alfabeto código.*

2.1.2 Códigos de Blocos

Os códigos de blocos se caracterizam pelo fato do processo de codificação ser feito sobre blocos de bits ou blocos de símbolos. Sendo que um feixe de bits ou símbolos é segmentado em blocos de k bits ou símbolos, a partir dos quais são geradas palavras-código com n bits ou símbolos. Assim, a notação que caracteriza um código de bloco é $C(n, k)$.

Se k bits estão contidos em um bloco de n bits, então a quantidade de bits de redundância introduzidos no processo de codificação é $n - k$. A quantidade de palavras-

código diferentes que podem ser geradas é q^k. Para códigos binários, teremos 2^k palavras-código.

2.2 Matriz Geradora

Definição 2.2.1 *Uma matriz geradora, G, é aquela que permite obter os vetores códigos, c_j, correspondentes às mensagens, m_i, a partir do produto interno determinado por $c_j = m_i.G$.*

Ela é uma consequência direta de uma base do subespaço vetorial. É uma matriz de dimensão $k \times n$, que consiste do arranjo formado pelos vetores linearmente independentes, ou vetores geradores, que compõem uma base do subespaço.

2.2.1 Matriz Sistemática e não Sistemática

A partir da base do subespaço vetorial, podemos obter a matriz geradora na forma não-sistemática G, em que cada vetor da base corresponderá a uma linha da matriz G. Sendo assim, neste caso a matriz geradora na forma não-sistemática será:

$$G = \begin{bmatrix} g_0 \\ g_1 \\ g_2 \end{bmatrix},$$

em que g_0, g_1 e g_2 são os vetores da base do subespaço vetorial.

Para obter uma matriz da forma sistemática $G^{'}$, basta efetuar operações com as linhas da matriz G de modo que a matriz fique do tipo:

$G^{'} = [P_{k\times(n-k)}|I_{k\times k}]$, ou seja

$$G' = \left[\begin{array}{cccc|cccc} p_{00} & p_{01} & \cdots & p_{0,n-k-1} & 1 & 0 & \ldots & 0 \\ p_{10} & p_{11} & \cdots & p_{1,n-k-1} & 0 & 1 & \ldots & 0 \\ . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . \\ p_{k-1,0} & p_{k-1,1} & \cdots & p_{k-1,n-k-1} & 0 & 0 & \ldots & 1 \end{array}\right] = \begin{bmatrix} g_0^{'} \\ g_1^{'} \\ . \\ . \\ . \\ g_{k-1}^{'} \end{bmatrix}.$$

2.3 Matriz Verificadora de Paridade

Um vetor recebido qualquer, r, pode ser entendido como um vetor código, o qual pode sofrer alguma alteração, consequência da adição de um padrão de erro no momento

de sua transmissão através de um canal de comunicação. Uma tarefa do decodificador é verificar se o vetor recebido é ou não um vetor código ou um vetor válido. Para obter êxito nesta tarefa, pode-se comparar o vetor recebido com todos os vetores códigos existentes, no entanto, se a ordem desses vetores for de algumas dezenas, esta terefa se tornará inviável.

$$H = \begin{bmatrix} h_0 \\ h_1 \\ . \\ . \\ . \\ h_{n-k-1} \end{bmatrix} = \begin{bmatrix} h_{00} & h_{01} & h_{02} & ... & h_{0,n-1} \\ h_{10} & h_{11} & h_{12} & ... & h_{1,n-1} \\ . & . & . & . & . \\ . & . & . & . & . \\ . & . & . & . & . \\ h_{n-k-1,0} & h_{n-k-1,1} & h_{n-k-1,2} & ... & h_{n-k-1,n-1} \end{bmatrix}_{(n-k)\times n}.$$

Quando a matriz G está na forma sistemática, ou seja, $G^{'} = [P_{k\times(n-k)}|I_{k\times k}]$, a obtenção de H é direta.

$$H = [I_{(n-k)\times(n-k)}|P^T] = \left[\begin{array}{ccccc|cccc} 1 & 0 & 0 & ... & 0 & P_{00} & P_{10} & ... & P_{k-1,0} \\ 0 & 1 & 0 & ... & 0 & P_{01} & P_{11} & ... & P_{k-1,1} \\ . & . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . & . \\ 0 & 0 & 0 & ... & 1 & P_{0,n-k-1} & P_{1,n-k-1} & ... & P_{k-1,n-k-1} \end{array}\right].$$

2.4 Peso e Distância de Hamming

2.4.1 Peso de Hamming

Definição 2.4.1 *Peso de Hamming de um vetor v, cuja notação é $\omega(v)$, é definido como o número de elementos não nulos em v. Para um vetor binário, o peso de Hamming é igual ao número de dígitos* "1" *contidos em v.*

2.4.2 Distância de Hamming

Definição 2.4.2 *Dados dois elementos $x = (x_1, x_2, \ldots, x_n)$ e $y = (y_1, y_2, \ldots, y_n)$ de um espaço A^n, chama-se distância de Hamming de x a y, o número de coordenadas que estes elementos se diferem, isto é:*

$$d(x, y) = \{i | x_i \neq y_i, 1 \leq i \leq n\}, i \in \mathbb{Z}.$$

2.5 Distância Mínima de um Código de Bloco Linear

A distância mínima (d_{min}) representa a menor distância de Hamming entre todos os pares de vetores códigos. Quanto maior for a distância mínima de um código, maior será sua capacidade de corrigir e detectar erros.

Como a distância de Hamming entre dois vetores código é determinada como sendo $d(c_1, c_2) = \omega(c_1 \oplus c_2) = \omega(c_3)$, basta verificar o peso de cada palavra código, com exceção da palavra toda nula. O menor peso encontrado corresponde à distância mínima do código.

2.6 Capacidade de Correção e Detecção de Erros

A capacidade de correção de erros (t) de um código de bloco linear é o número de erros corrigíveis garantidamente por palavra código. Este número é dado pela expressão: $t = \lfloor \frac{d_{min}-1}{2} \rfloor$, onde $\lfloor x \rfloor$ é o maior inteiro que não excede o valor de x.

Agora se o código for usado apenas para detectar erros ao invés de corrigir, a capacidade de detecção de erros é dado por: $\varrho = d_{min} - 1$, onde ϱ é o número de erros detectados.

2.7 Síndrome de Erro

Síndrome de erro (S) é um vetor com $n - k$ bits definido pele operação: $S = r.H^t$, sendo $r = (r_1, r_2, r_3, \ldots, r_n)$ um vetor recebido, resultante da transmissão de um vetor código $c = (c_1, c_2, c_3, \ldots, c_n)$, através de um canal com ruído. Assim $r = c \oplus e$, em que $e = (e_1, e_2, e_3, \ldots, e_n)$ é um vetor erro ou padrão de erro introduzido pelo canal. Então, podemos reescrever a operação desta forma:

$[S = (c \oplus e).H^t = c.H^t \oplus e.H^t]$. Como $c.H^t = 0$, temos que $S = e.H^t$.

A síndrome está associada a um padrão de erro. Esta é uma importante propriedade, fundamental para o processo de decodificação, ou seja, cada padrão de erro corrigível deve estar associado a uma síndrome específica.

Um código de bloco linear $C(n, k)$ com capacidade de correção de t erros é capaz de corrigir um total de 2^{n-k} padrões de erros.

2.8 Códigos Perfeitos

Os códigos que corrigem exclusivamente todos os padrões menores ou iguais a t erros, ou seja, não corrige nenhum padrão maior que t erros, são denominados códigos

perfeitos. Para saber se um código é perfeito basta verificar se a seguinte igualdade é válida: $2^{n-k} = \sum_{i=0}^{t} \binom{n}{i}$.

2.9 Correção de Erros pela Síndrome

O processo de correção de erros pela síndrome é feito a partir da identificação do padrão de erro mais provável por meio do cálculo da síndrome de erros. Uma vez conhecido o padrão de erro, é possível fazer a correção do erro, somando-se o vetor r com o padrão de erro e, associado a ele, pois $r = c \oplus e \Rightarrow c^{'} = r \oplus e$, em que $c^{'}$ é a melhor estimativa do vetor código que foi transmitido pelo canal ruidoso.

Para fazer a correção de erros pela síndrome, é preciso seguir alguns passsos:

- A partir da capacidade de correção de erros do código, determinar a síndrome para todos os padrões de erros corrigíveis;
- Calcular a síndrome de r;
- Localizar o padrão de erro correspondente à síndrome calculada;
- O vetor código será aquele determinado por $c^{'} = r \oplus e$.

No entanto, o erro só será corrigido se o padrão de erro correspondente à síndrome do vetor recebido for igual ao padrão de erro introduzido pelo canal, isto é, o padrão de erro indroduzido pelo canal deve ser um padrão de erro corrigível.

2.10 Arranjo Padrão

O arranjo padrão é um esquema de decodificação baseado em síndrome de erros. Considere as 2^k palavras código de um código de bloco linear $c_1, c_2, \ldots, c_{2^k}$ e também os 2^{n-k} padrões de erros $e_1, e_2, \ldots, e_{2^{n-k}}$, associados às 2^{n-k} síndromes possíveis. Sendo assim, um arranjo padrão é constituído por todas as palavras do espaço vetorial V_n, de acordo com os passos apresentados a seguir:

1. Um arranjo padrão é formado por 2^k subconjuntos, sendo que cada subconjunto é uma coluna do arranjo padrão;
2. A primeira linha do arranjo padrão é composta por todas as 2^k palavras códigos. Obrigatoriamente, a palavra toda nula é a palavra que ocupa a posição superior esquerda do arranjo;

3. A primeira coluna do arranjo padrão é formada por todos os 2^{n-k} padrões de erros, incluindo o vetor todo zero que ocupa a posição superior esquerda do arranjo.

A segunda linha do arranjo é formada pela soma de e_2 com cada um dos vetores códigos da primeira linha. Este procedimento se repete até que a $(n-k)$ é-sima linha complete o arranjo. Cada linha do arranjo é chamada de **coset** e o vetor do coset que pertence a primeira coluna é chamado de **líder do coset**. O arranjo padrão permite a decodificação direta da palavra recebida pela identificação de um dos 2^k subconjuntos, ou coluna, ao qual pertence a palavra recebida. Uma vez identificada essa coluna, o vetor código decodificado é o vetor da primeira linha da coluna identificada, como mostrado na Tabela 1 a seguir.

$c_1 = e_1 = 0$	c_2	...	c_i	...	c_{2^k}
e_2	$e_2 + c_2$	...	$e_2 + c_i$	...	$e_2 + c_{2^k}$
.	.	...	.	...	.
.	.	...	.	...	.
.	.	...	.	...	.
e_j	$e_j + c_2$	...	$e_j + c_i$	...	$e_j + c_{2^k}$
.	.	...	.	...	.
.	.	...	.	...	.
.	.	...	.	...	.
e_{2^n-k}	$e_{2^n-k} + c_2$	...	$e_{2^n-k} + c_i$	...	$e_{2^n-k} + c_{2^k}$

Tabela 3 – Arranjo Padrão.

Vale ressaltar que para o processo de construção de códigos corretores de erros, codificação e decodificação, os tópicos apresentados anteriormente, apesar de em sua maioria estarem definidos sobre $\mathbb{R}$, foi utilizada a estrutura de corpos finitos, em particular, o corpo $\mathbb{F}_2 = \{0, 1\}$.

3 Construção dos Códigos de Blocos $C(7,3)$ e $C(8,4)$

Neste capítulo serão apresentadas as construções dos códigos $C(7,3)$ e $C(8,4)$, buscando explicitar a importância dos elementos de Álgebra Linear nestas construções, tais como matrizes, espaços e subespaços vetoriais, vetores LD e LI, base e dimensão, além de elementos de sistemas de comunicação e códigos, que utilizam essas estruturas em sua construção. Essas construções tomaram como referências [7] e [8].

Na seção 3.1 é apresentada a construção do código $C(7,3)$, na seção 3.2 é apresentada a construção do código $C(8,4)$ e na seção 3.3 são apresentadas as contribuições da Álgebra Linear nestas construções.

3.1 Construção do Código $C(7,3)$

Conforme já mencionado, todo código corretor de erros é representado por $C(n,k)$, para esta construção temos $n=7$ e $k=3$.

Seja o subespaço vetorial formado pelos $2^k \Rightarrow 2^3 = 8$ vetores:

$[\{(0000000),(0011001),(1010101),(0101011),(1001100),(0110010),(1111110),(1100111)\}]$.

Desses oitos vetores, escolhe-se três vetores LI que geram todo o subespaço vetorial. Sejam: $[a=(0101011),\ b=(1010101)$ e $c=(0011001)]$.

Note que:

$$[\alpha(0101011)+\beta(1010101)+\delta(0011001)=(0000000)]$$

$$\begin{cases} \beta = 0 \\ \alpha = 0 \\ \beta + \delta = 0 \Rightarrow \delta = 0 \end{cases}$$

Vamos agora mostrar que estes três vetores, além de serem LI, também geram todo o subespaço vetorial.

$000 = 0(0101011) + 0(1010101) + 0(0011001) = (0000000)$
$001 = 0(0101011) + 0(1010101) + 1(0011001) = \mathbf{(0011001)}$
$010 = 0(0101011) + 1(1010101) + 0(0011001) = \mathbf{(1010101)}$
$100 = 1(0101011) + 0(1010101) + 0(0011001) = \mathbf{(0101011)}$
$011 = 0(0101011) + 1(1010101) + 1(0011001) = (1001100)$
$101 = 1(0101011) + 0(1010101) + 1(0011001) = (0110010)$

$110 = 1(0101011) + \ 1(1010101) + \ 0(0011001) = (1111110)$
$111 = 1(0101011) + \ 1(1010101) + \ 1(0011001) = (1100111)$
Como os vetores $a,\ b$, e c são LI e geram todo o subsepaço vetorial, então $a,\ b$, e c formam uma base do subespaço vetorial.

3.1.1 Matriz Geradora

A partir da base do subespaço vetorial, pode-se obter a matriz geradora G, onde cada vetor da base corresponderá a uma linha da matriz G. Sendo assim, neste caso, a matriz geradora será:

$$G = \begin{bmatrix} g_0 \\ g_1 \\ g_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix}_{3\times 7}.$$

Esta matriz G é da forma não-sistemática. Para obter uma matriz da forma sistemática $G^{'}$, basta efetuar operações com as linhas da matriz G de modo que a matriz fique do tipo:

$$G^{'} = [P_{k\times(n-k)}|I_{k\times k}].$$

Neste caso, temos $n = 7$ e $k = 3$, logo $G^{'} = [P_{3\times 4}|I_{3\times 3}]$.

Podemos fazer as seguintes operacões para obter a matriz geradora na forma sistemática $G^{'}$.
${g_0}^{'} = g_1 + g_2 = (1010101) + (0011001) = (1001100)$,
${g_1}^{'} = g_0 + g_2 = (0101011) + (0011001) = (0110010)$,
${g_2}^{'} = g_2 = (0011001)$.

$$G^{'} = \begin{bmatrix} {g_0}^{'} \\ {g_1}^{'} \\ {g_2}^{'} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix}_{3\times 7}.$$

Na Tabela 2 são apresentados os vetores mensagens e seus respectivos vetores códigos.

3.1.2 Matriz Verificadora de Paridade

A partir da matriz geradora do código $C(7,3)$, pode-se obter a matriz verificadora de paridade H. Como:

m	G′	c=m.G′
000	0(1001100) + 0(0110010) + 0(0011001)	(0000000)
001	0(1001100) + 0(0110010) + 1(0011001)	(0011001)
010	0(1001100) + 1(0110010) + 0(0011001)	(0110010)
100	1(1001100) + 0(0110010) + 0(0011001)	(1001100)
011	0(1001100) + 1(0110010) + 1(0011001)	(0101011)
101	1(1001100) + 0(0110010) + 1(0011001)	(1010101)
110	1(1001100) + 1(0110010) + 0(0011001)	(1111110)
111	1(1001100) + 1(0110010) + 1(0011001)	(1100111)

Tabela 4 – Palavras-código do código $C(7,3)$ na forma sistemática.

$$G^{'} = [P_{k\times(n-k)}|I_{k\times k}] = \left[\begin{array}{cccc|ccc} 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 \end{array}\right]_{3\times 7},$$

temos que:

$$H = [I_{(n-k)\times(n-k)}|P^t] = \left[\begin{array}{cccc|ccc} 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 \end{array}\right]_{4\times 7}.$$

3.1.3 Distância Mínima

Vamos calcular a distância mínima do código $C(7,3)$, apresentada na Tabela 3 a seguir.

m	c	ω
000	0000000	0
001	0011001	3
010	1010101	4
100	0101011	4
011	1001100	3
101	0110010	3
110	1111110	6
111	1100111	5

Tabela 5 – Distância mínima do código $C(7,3)$.

Para este código, a distância mínima é igual a 3. Normalmente, os códigos com distância mínima calculada, são representados pela notação $C(n,k,d_{min})$. Sendo assim, o código aqui estudado será representado pela seguinte notação: $C(7,3,3)$.

3.1.4 Capacidade de Correção e Detecção de Erros

A capacidade de correção de erros(t) de um código é dada por: $t = \lfloor \frac{d_{min}-1}{2} \rfloor$, e a capacidade de detecção de erros (ϱ), dada por: $\varrho = d_{min} - 1$. Calculando a capacidade de correção e detecção de erros do código $C(7,3,3)$, respectivamente, temos:
$t = \lfloor \frac{d_{min}-1}{2} \rfloor \Rightarrow t = \lfloor \frac{3-1}{2} \rfloor \Rightarrow t = 1$,
$\varrho = d_{min} - 1 \Rightarrow \varrho = 3 - 1 \Rightarrow \varrho = 2$.

Com base nos resultados, pode-se afirmar que o código $C(7,3,3)$ tem a capacidade de detectar até dois erros, no entanto a capacidade de correção é de apenas um erro.

3.1.5 Síndrome de Erro

Para calcular a síndrome de erros é necessário utilizar a expressão $S = e.H^t$. Calcularemos a síndrome de erros do código $C(7,3,3)$, apresentada na Tabela 6, lembrando que:

$$H = \left[\begin{array}{cccc|ccc} 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 \end{array}\right]_{4\times 7}.$$

Logo:

$$H^t = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix}_{7\times 4}.$$

e(t=1)	S
1000000	1000
0100000	0100
0010000	0010
0001000	0001
0000100	1001
0000010	0110
0000001	0011

Tabela 6 – Síndrome de erros do código $C(7,3,3)$.

3.1.6 O Código $(7,3,3)$ é perfeito?

Vamos verificar se o código $C(7,3,3)$ é perfeito por meio da igualdade: $2^{n-k} = \sum_{i=0}^{t} \binom{n}{i}$. Temos $n = 7, k = 3$ e $t = 1$, logo:

$$2^{7-3} = \sum_{i=0}^{1} \binom{7}{i} \Rightarrow 2^4 = \binom{7}{0}.\binom{7}{1} \Rightarrow 16 = 1 + 7 \Rightarrow 16 = 8,$$ o que não é verdade.

Portanto o código $C(7,3,3)$ não é um código perfeito.

3.1.7 Correção de Erros pela Síndrome

Já vimos que um código de bloco linear $C(n,k)$ com capacidade de correção de t erros é capaz de corrigir um total de 2^{n-k} padrões de erros. Sendo assim, para o código $C(7,3,3)$ temos, $2^{7-3} = 2^4 = 16$, portanto este código é capaz de corrigir dezesseis padrões de erros. Lembrando que para este código $t = 1$. Na Tabela 7, são apresentados os dezesseis padrões de erros.

$e(t=1)$	S
0000000	0000
1000000	1000
0100000	0100
0010000	0010
0001000	0001
0000100	1001
0000010	0110
0000001	0011
1100000	1100
1000010	1110
0100100	1101
0000110	1111
1010000	1010
1000001	1011
0101000	0101
0100001	0111

Tabela 7 – Padrões de erros do código $C(7,3,3)$.

A seguir serão apresentados alguns exemplos com casos onde a síndrome de erros foi utilizada na correção de um erro, em um processo de transmissaõ da informação.

Exemplo 3.1.1 *Suponha que o vetor* $c = (1001100)$ *do código* $C(7,3,3)$ *tenha sido transmitido e corrompido por um ruído no canal, de modo que na recepção foi detectado o vetor* $r = (1001110)$. *O primeiro passo é identificar a qual síndrome o vetor* r *está associado.*

$$S = r.H^t = (1001110).\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix}_{8\times 4} = (0110).$$

Note que na Tabela 7*, a síndrome* (0110) *está associada ao padrão de erro* (0000010)*, como* $c^{'} = r \oplus e$*, então* $c^{'} = 1001110 + 0000010 = 1001100$*, recuperando dessa forma a mensagem enviada originalmente.*

Exemplo 3.1.2 *Suponha que o vetor* $c = (0101011)$ *do código* $C(7,3,3)$ *tenha sido transmitido e corrompido por um ruído no canal, de modo que na recepção foi detectado o vetor* $r = (1101001)$*. Vamos identificar a qual síndrome o vetor* r *está associado.*

$$S = r.H^t = (1101001).\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix}_{7\times 4} = (1110).$$

Note que na Tabela 7*, a síndrome* (1110) *está associada ao padrão de erro* (1000010)*, como* $c^{'} = r \oplus e$*, então* $c^{'} = 1101001 + 1000010 = 0101011$*.*

Note que a correção foi feita de maneira correta, mesmo o vetor apresentando padrão de dois erros. É importante ressaltar que para o código $C(7,3,3)$ *nem sempre um código com dois erros será corrigido de forma correta.*

3.1.8 Arranjo Padrão

A seguir serão apresentados alguns exemplos de utilização do arranjo padrão do código $C(7,3,3)$ no processo de decodificação de vetores recebidos em um processo de transmissão da informação. Este arranjo padrão é apresentado na Tabela 8.

Exemplo 3.1.3 *Usando o arranjo padrão do código* $(7,3,3)$*, decodifique o vetor recebido* 1001000*.*

Note que este vetor está representado na Tabela 8 *na cor vermelha, este vetor está associado ao padrão de erro* 0000100 *que é o primeiro vetor de sua respectiva linha, como já vimos* $c^{'} = c \oplus e \Rightarrow c^{'} = (1001000 + 0000100) \Rightarrow c^{'} = 1001100$*, se observarmos*

novamente o vetor recebido na Tabela 8*, veremos que o primeiro vetor de sua respectiva coluna é justamente o vetor encontrado* 1001100*.*

Exemplo 3.1.4 *Usando o arranjo padrão do código* $(7,3,3)$*, decodifique o vetor recebido* 1111101*.*

Note que este vetor está representado na Tabela 8 *na cor azul, o qual está associado ao padrão de erro* 0101000 *que é o primeiro vetor de sua respectiva linha, como já vimos* $c^{'} = c \oplus e \Rightarrow c^{'} = (1111101 + 0101000) \Rightarrow c^{'} = 1010101$*, se observarmos novamente o vetor recebido na Tabela* 8*, veremos que o primeiro vetor de sua respectiva coluna é justamente o vetor encontrado* 1010101*.*

Exemplo 3.1.5 *Usando o arranjo padrão do código* $(7,3,3)$*, decodifique o vetor recebido* 1011001*.*

Note que este vetor está representado na Tabela 8 *na cor verde, associado ao padrão de erro* 1000000 *que é o primeiro vetor de sua respectiva linha, como já vimos* $c^{'} = c \oplus e \Rightarrow c^{'} = (1011001 + 1000000) \Rightarrow c^{'} = 0011001$*, se observarmos novamente o vetor recebido na Tabela* 8*, veremos que o primeiro vetor de sua respectiva coluna é justamente o vetor encontrado* 0011001*.*

0000000	**0011001**	1010010	**1001100**	**0101011**	**1010101**	**1111110**	**1100111**
1000000	1011001	1110010	0001100	1101011	0010101	0111110	0100111
0100000	0111001	0010010	1101100	0001011	1110101	1011110	1000111
0010000	0001001	0100010	1011100	0111011	1000101	1101110	1110111
0001000	0010001	0111010	1000100	0100011	1011101	1110110	1100111
0000100	0011101	0110110	1001000	0101111	1010001	1111010	1100011
0000010	0011011	0110000	1001110	0101001	1010111	1111100	1100101
0000001	0011000	0110011	1001101	0101010	1010100	1111111	1100110
1100000	1111001	1010010	0101100	1001011	0110101	0011110	0000111
1000010	1011010	1110000	0001110	1101001	0010111	01110100	0100101
0100100	0111101	0010110	1101000	0001111	1110001	1011010	1000011
0000110	0011111	0110100	1001010	0101101	1010011	1111000	1100001
1010000	1001001	1100010	0011100	1111011	0000101	0101110	0110111
1000001	1011000	1110011	0001101	1101010	0010100	0111111	0100110
0101000	0110001	0011010	1100100	0000011	1111101	1010110	1001111
0100001	0111000	0010011	1101101	0001010	1110100	1011111	1000110

Tabela 8 – Arranjo Padrão $C(7,3,3)$.

3.2 Construção do Código $C(8,4)$

Seja o subespaço vetorial formado pelos $2^k \Rightarrow 2^4 = 16$ vetores:

$\{(00000000), (10110001), (11010010), (11100100), (01111000), (01100011), (01010101), (11001001), (00110110), (10101010), (10011100), (10000111), (00011011), (00101101), (01001110), (11111111)\}$.

Desses dezesseis vetores, vamos escolher quatro vetores LI que geram todo o subespaço vetorial. Sejam:

$a = (01111000), b = (11100100), c = (11010010)$ e $d = (10110001)$.

Note que:

$\alpha(01111000) + \beta(11100100) + \theta(11010010) + \delta(10110001) = (00000000)$

$$\begin{cases} \alpha + \beta + \theta = 0 \Rightarrow \theta = 0 \\ \alpha + \beta + \delta = 0 \Rightarrow \delta = 0 \\ \alpha + \theta + \delta = 0 \\ \alpha = 0 \\ \beta = 0 \end{cases}.$$

Vamos agora mostrar que estes quatro vetores, além de serem LI, também geram todo o subespaço vetorial.

$0000 = 0(01111000) + 0(11100100) + 0(11010010) + 0(10110001) = (00000000)$
$0001 = 0(01111000) + 0(11100100) + 0(11010010) + 1(10110001) = \mathbf{(10110001)}$
$0010 = 0(01111000) + 0(11100100) + 1(11010010) + 0(10110001) = \mathbf{(11010010)}$
$0100 = 0(01111000) + 1(11100100) + 0(11010010) + 0(10110001) = \mathbf{(11100100)}$
$1000 = 1(01111000) + 0(11100100) + 0(11010010) + 0(10110001) = \mathbf{(01111000)}$
$0011 = 0(01111000) + 0(11100100) + 1(11010010) + 1(10110001) = (01100011)$
$0101 = 0(01111000) + 1(11100100) + 0(11010010) + 1(10110001) = (01010101)$
$1001 = 1(01111000) + 0(11100100) + 0(11010010) + 1(10110001) = (11001001)$
$0110 = 0(01111000) + 1(11100100) + 1(11010010) + 0(10110001) = (00110110)$
$1010 = 1(01111000) + 0(11100100) + 1(11010010) + 0(10110001) = (10101010)$
$1100 = 1(01111000) + 1(11100100) + 0(11010010) + 0(10110001) = (10011100)$
$0111 = 0(01111000) + 1(11100100) + 1(11010010) + 1(10110001) = (10000111)$
$1011 = 1(01111000) + 0(11100100) + 1(11010010) + 1(10110001) = (00011011)$
$1101 = 1(01111000) + 1(11100100) + 0(11010010) + 1(10110001) = (00101101)$
$1110 = 1(01111000) + 1(11100100) + 1(11010010) + 0(10110001) = (01001110)$
$1111 = 1(01111000) + 1(11100100) + 1(11010010) + 1(10110001) = (11111111)$

Como os vetores $a,\ b,\ c$ e d são LI e geram todo o subsepaço vetorial, então $a,\ b,\ c$ e d formam uma base do subespaço vetorial.

3.2.1 Matriz Geradora

A partir da base do subespaço vetorial, podemos obter a matriz geradora G, de forma que cada vetor base corresponderá a uma linha da matriz G. Sendo assim, neste caso a matriz geradora será:

$$G = \begin{bmatrix} g_0 \\ g_1 \\ g_2 \\ g_3 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}_{4\times 8}.$$

Para obter uma matriz da forma sistemática $G^{'}$, basta efetuar operações com as linhas da matriz G, de modo que a matriz fique do tipo:

$[G^{'} = [P_{k\times(n-k)}|I_{k\times k}]]$.
Neste caso, temos $n = 8$ e $k = 4$, logo $G^{'} = [P_{4\times 4}|I_{4\times 4}]$.

$$G^{'} = \begin{bmatrix} {g_0}^{'} \\ {g_1}^{'} \\ {g_2}^{'} \\ {g_3}^{'} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 1 & 1 & \mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ 1 & 1 & 1 & 0 & \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{0} \\ 1 & 1 & 0 & 1 & \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{0} \\ 1 & 0 & 1 & 1 & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} \end{bmatrix}_{4\times 8}.$$

Note que a matriz G já está na forma sistemática.

A Tabela 8 a seguir apresenta os vetores mensagens e seus respectivos vetores códigos.

m	G$^{'}$	c=m.G$^{'}$
0000	0(01111000)+ 0(11100100)+ 0(11010010)+ 0(10110001)	(0000**0000**)
0001	0(01111000)+ 0(11100100)+ 0(11010010)+ 1(10110001)	(1011**0001**)
0010	0(01111000)+ 0(11100100)+ 1(11010010)+ 0(10110001)	(1101**0010**)
0100	0(01111000)+ 1(11100100)+ 0(11010010)+ 0(10110001)	(1110**0100**)
1000	1(01111000)+ 0(11100100)+ 0(11010010)+ 0(10110001)	(0111**1000**)
0011	0(01111000)+ 0(11100100)+ 1(11010010)+ 1(10110001)	(0110**0011**)
0101	0(01111000)+ 1(11100100)+ 0(11010010)+ 1(10110001)	(0101**0101**)
1001	1(01111000)+ 0(11100100)+ 0(11010010)+ 1(10110001)	(1100**1001**)
0110	0(01111000)+ 1(11100100)+ 1(11010010)+ 0(10110001)	(0011**0110**)
1010	1(01111000)+ 0(11100100)+ 1(11010010)+ 0(10110001)	(1010**1010**)
1100	1(01111000)+ 1(11100100)+ 0(11010010)+ 0(10110001)	(1001**1100**)
0111	0(01111000)+ 1(11100100)+ 1(11010010)+ 1(10110001)	(1000**0111**)
1011	1(01111000)+ 0(11100100)+ 1(11010010)+ 1(10110001)	(0001**1011**)
1101	1(01111000)+ 1(11100100)+ 0(11010010)+ 1(10110001)	0010**1101**)
1110	1(01111000)+ 1(11100100)+ 1(11010010)+ 0(10110001)	(0100**1110**)
1111	1(01111000)+ 1(11100100)+ 1(11010010)+ 1(10110001)	(1111**1111**)

Tabela 9 – Palavras-código do código $C(8,4)$ na forma sistemática.

3.2.2 Matriz Verificadora de Paridade

A partir da matriz geradora do código $C(8,4)$, pode-se obter a matriz verificadora de paridade (H).Como:

$$G' = [P_{k\times(n-k)}|I_{k\times k}] = \left[\begin{array}{cccc|cccc} 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 \end{array}\right]_{4\times 8}.$$

temos que:

$$H = [I_{(n-k)\times(n-k)}|P^T] = \left[\begin{array}{cccc|cccc} 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 \end{array}\right]_{4\times 8}.$$

3.2.3 Distância Mínima do Código $C(8,4)$

Vamos calcular a distância mínima do código $C(8,4)$, apresentada na Tabela 9 a seguir.

m	c	ω
0000	00000000	0
0001	10110001	4
0010	11010010	4
0100	11100100	4
1000	01111000	4
0011	01100011	4
0101	01010101	4
1001	11001001	4
0110	00110110	4
1010	10101010	4
1100	10011100	4
0111	10000111	4
1011	00011011	4
1101	00101101	4
1110	01001110	4
1111	11111111	8

Tabela 10 – Distância mínima do código $C(8,4)$.

Para este código a distância mínima é igual a 4. Este código pode ser representado pela notação $C(8,4,4)$.

3.2.4 Capacidade de Correção e Detecção de Erros

Vamos calcular a capacidade de correção e detecção de erros do código $C(8,4,4)$, respectivamente.
$[t = \lfloor \frac{d_{min}-1}{2} \rfloor \Rightarrow t = \lfloor \frac{4-1}{2} \rfloor \Rightarrow t = 1]$,
e
$[\varrho = d_{min} - 1 \Rightarrow \varrho = 4 - 1 \Rightarrow \varrho = 3]$.

Com base nos resultados, pode-se afirmar que o código $C(8,4,4)$ tem a capacidade de detectar até três erros, no entanto a capacidade de correção é de apenas um erro, garantidamente.

3.2.5 Síndrome de Erro

Vamos calcular a síndrome de erros do código $C(8,4,4)$, apresentada na Tabela 11 a seguir. Lembrando que:

$$H = \left[\begin{array}{cccc|cccc} 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 \end{array}\right]_{4\times 8}.$$

Logo:

$$H^t = \left[\begin{array}{cccc} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \end{array}\right]_{8\times 4}.$$

Daí

e(t=1)	S
10000000	1000
01000000	0100
00100000	0010
00010000	0001
00001000	0111
00000100	1110
00000010	1101
00000001	1011

Tabela 11 – Síndrome de erros do código $C(8,4,4)$.

3.2.6 O Código $C(8,4,4)$ é perfeito?

Temos $n = 8, k = 4$ e $t = 1$, logo:

$[2^{8-4} = \sum_{i=0}^{1} \binom{8}{i} \Rightarrow 2^4 = \binom{8}{0}.\binom{8}{1} \Rightarrow 16 = 1 + 8 \Rightarrow 16 = 9]$, o que não é verdade.

Portanto o código $C(8,4,4)$ não é um código perfeito.

3.2.7 Correção de Erros pela Síndrome

Para o código $C(8,4,4)$ temos, $2^{8-4} = 2^4 = 16$, portanto este código é capaz de corrigir desseseis padrões de erros. Lembrando que para este código $t = 1$. A Tabela 12 apresenta os dezesseis padrões de erros.

$e(t=1)$	S
00000000	0000
10000000	1000
01000000	0100
00100000	0010
00010000	0001
00001000	0111
00000100	1110
00000010	1101
00000001	1011
10010000	1001
10000001	0011
01010000	0101
01000001	1111
00010010	1100
00001010	1010
00000011	0110

Tabela 12 – Padrões de erros do código $C(8,4,4)$.

A seguir serão apresentados alguns exemplos com casos onde a síndrome de erros foi utilizada na correção de um erro em um processo de transmissão da informação.

Exemplo 3.2.1 *Suponha que o vetor* $c = (11010010)$ *do código* $C(8,4,4)$ *tenha sido transmitido e corrompido por um ruído no canal, de modo que na recepção foi detectado o vetor* $r = (11000010)$*. O primeiro passo é identificar a qual síndrome o vetor* r *esta associado.*

$$[S = r.H^t = (11000010).\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \end{bmatrix}_{8\times 4} = (0001)].$$

Note que na Tabela 12 *a síndrome* (0001) *está associada ao padrão de erro* (00010000)*, como* $c^{'} = r \oplus e$*, então* $[c^{'} = 11000010 + 00010000 = 11010010]$*, recuperando a mensagem originalmente enviada.*

Exemplo 3.2.2 *Suponha que o vetor* $c = (11001001)$ *do código* $C(8,4,4)$ *tenha sido transmitido e corrompido por um ruído no canal, de modo que na recepção foi detectado o vetor* $r = (01000001)$*. Calculando a síndrome, temos:*

$$[S = r.H^t = (01000001).\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \end{bmatrix}_{8\times 4} = (1111)].$$

De acordo com a Tabela 12 *a síndrome* (1111) *está associada ao padrão de erro* (01000001)*, como* $c^{'} = r \oplus e$*, então* $[c^{'} = 01000001 + 01000001 = 00000000]$.

Note que a correção foi feita, no entanto de maneira equivocada, pois o código $C(8,4,4)$ *corrige garantidamente padrões de até um erro. Neste exemplo, ocorreram dois erros. Sendo assim, se a palavra recebida tiver mais de um erro, ela pode ser corrigida, mas não se tem a certeza de que foi corrigida corretamente.*

3.2.8 Arranjo Padrão

O arranjo padrão para o código $C(8,4,4)$ está representado nas Tabelas 13, 14 e 15.

Exemplo 3.2.3 *Usando o arranjo padrão do código* $C(8,4,4)$*, decodifique o vetor recebido* 11100000*.*

Note que este vetor está representado na Tabela 13 *na cor vermelha, associado ao padrão de erro* 00000100 *que é o primeiro vetor de sua respectiva linha, como já vimos*

$[c^{'} = c \oplus e \Rightarrow c^{'} = (11100000 + 00000100) \Rightarrow c^{'} = 11100100]$, *se novamente olharmos para o vetor recebido na Tabela* 13, *veremos que o primeiro vetor de sua respectiva coluna é justamente o vetor encontrado* 11100100.

Exemplo 3.2.4 *Suponhamos que tenha sido enviado o vetor* 11111111, *e que o mesmo foi corrompido por um ruído e chegado ao destinatário como* 11111001, *se observarmos na Tabela* 13 *o vetor recebido está representado na cor azul, associado ao padrão de erro* 10000001 *que é o primeiro vetor de sua respectiva linha, como já vimos* $[c^{'} = c \oplus e \Rightarrow c^{'} = (11111001 + 10000001) \Rightarrow c^{'} = 01111000]$, *note que o vetor corrigido pelo arranjo padrão é diferente do vetor originalmente enviado.*

00000000	**10110001**	**11010010**	**11100100**	**01111000**	**01100011**
10000000	00110001	01010010	01100100	11111000	11100011
01000000	11110001	10010010	10100100	00111000	00100011
00100000	10010001	11110010	11000100	01011000	01000011
00010000	10100001	11000010	11110100	01101000	01110011
00001000	10111001	11011010	11101100	01110000	01101011
00000100	10110101	11010110	11100000	01111100	01100111
00000010	10110011	11010000	11100110	01111010	01100001
00000001	10110000	11010011	11100101	01111001	01100010
10010000	00100001	01000010	01110100	11101000	11110011
10000001	00110000	01010011	01100101	11111001	11100010
01010000	11100001	10000010	10110100	00101000	00110011
01000001	11110000	10010011	10100101	00111001	00100010
00010010	11010011	11100000	10000110	01001010	00010001
00001010	10111011	11011000	11101110	01110010	01101001
00000011	10110010	11010001	11100111	01111011	01100000

Tabela 13 – Arranjo padrão $C(8,4,4)$.

00000000	**01010101**	**11001001**	**00110110**	**10101010**	**10011100**
10000000	11010101	01001001	10110110	00101010	00011100
01000000	00010101	10001001	01110110	11101010	11011100
00100000	01110101	11101001	00010110	10001010	10111100
00010000	01000101	11011001	00100110	10111010	10001100
00001000	01011101	11000001	00111110	10100010	10010100
00000100	01010001	11001101	00110010	10101110	10011000
00000010	01010111	11001011	00110100	10101000	10011110
00000001	01010100	11001000	00110111	10101011	10011101
10010000	11000101	01011001	10100110	00111010	00001100
10000001	11010100	01001000	10110111	00101011	00011101
01010000	00000101	10011001	01100110	11111010	11001100
01000001	00010100	10001000	01110111	11101011	11011101
00010010	01110111	10111011	01000100	10111000	10001110
00001010	01011111	11000011	00111100	10100000	10010110
00000011	01010110	11001010	00110101	10101001	10011111

Tabela 14 – Arranjo padrão $C(8,4,4)$.

00000000	**10000111**	**00011011**	**00101101**	**01001110**	**11111111**
10000000	00000111	10011011	10101101	11001110	01111111
01000000	11000111	01011011	01101101	00001110	10111111
00100000	10100111	00111011	00001101	01101110	11011111
00010000	10010111	00001011	00111101	01011110	11101111
00001000	10001111	00010011	00100101	01000110	11110111
00000100	10000011	00011111	00101001	01001010	11111011
00000010	10000101	00011001	00101111	01001100	11111101
00000001	10000110	00011010	00101100	01001111	11111110
10010000	00010111	10001011	10111101	11011110	01101111
10000001	00000110	10011010	10101100	11001111	01111110
01010000	11010111	01001011	01111101	00011110	10101111
01000001	11000110	01011010	01101100	00001111	10111110
00010010	10010101	00001001	00111111	01011100	11101101
00001010	10001101	00010001	00100111	01000100	11110101
00000011	10000100	00011000	00101110	01001101	11111100

Tabela 15 – Arranjo padão $C(8,4,4)$.

3.3 Contribuições da Álgebra Linear nas Construções Realizadas.

No processo de construção dos códigos corretores de erros, inicialmente foram escolhidos um total de 2^k vetores para formar um subespaço vetorial, sendo que cada vetor possuia comprimento de n bits. A seguir é necessário escolher k vetores linearmente independentes e que geram todo o subespaço, gerando assim a base do subespaço vetorial.

Cada vetor da base será uma linha de uma matriz,chamada de matris geradora, a qual será o ponto de partida para construirmos a matriz verificadora de paridade na

forma sistemática, neste processo utilizaremos matrizes identidade e transposta.

É necessário também neste processo utilizarmos a definição de corpo finito, uma vez que se trabalha com o corpo $\mathbb{F}_2$, pois se trata de um código binário.

Podemos observar que foram utilizados vários elementos da Álgebra Linear nas construções realizadas, tais como: subespaço vetorial, base, vetores linearmente dependentes e independentes, matriz identidade e transposta, multiplicação de matrizes e estrutura algébrica de corpos finitos.

4 *Considerações Finais e Perspectivas para Trabalhos Futuros*

Neste trabalho foi explicitada a importância de alguns elementos da Álgebra Linear na construção de códigos corretores de erros, voltado em especial para matemáticos e engenheiros de comunicação.

Também foi mostrado que a Álgebra Linear contribui de forma significativa para minimizar ruídos no processo de transmissão da informação, pois é a base para se construir os códigos corretores de erros.

Foram apresentadas as construções dos códigos $C(7,3)$ e $C(8,4)$. Nessas construções foram apresentadas as matrizes geradoras, verificação de paridade, a síndrome de erros, as capacidades de detecção e correção de erros, além do arranjo padrão, de vital importância no processo de decodificação. Foram apresentados alguns exemplos de decodificação utilizando tanto a síndrome de erros quanto a construção do arranjo padrão para ambos os códigos.

Diversos elementos de Álgebra Linear foram utilizados neste trabalho, como matrizes, vetores LD e LI, combinações lineares, base e dimensão, de forma que considera-se que o objetivo proposto foi alcançado.

Explicitada a importância da Álgebra Linear na construção de um código corretor de erros, pode-se afirmar que este trabalho contribuiu para o desenvolvimento do discente, uma vez que utiliza na prática conteúdos antes vistos apenas na teoria.

Por fim, vale ressaltar que este trabalho pode ser estendido a novas pesquisas, pois esta área de pesquisa é extensa e ainda com muitas questões a serem respondidas, como por exemplo, as possibilidades de aplicações dos elementos estudados, seja em um sistema de comunicação padrão como em um sistema de comunicação genético.

Este trabalho de conclusão de curso foi iniciado presencialmente, mas devido à pandemia de COVID-19, a partir de março de 2020 a condução do mesmo foi feita de forma remota, através de plataformas digitais como Google Meet, e-mail e WhatsApp. Um dos resultados deste trabalho foi apresentado no X ERMAC-RS, realizado de forma remota, de 01 a 03 de dezembro de 2020, devido à pandemia de Covid-19.

Referências

[1] FARIA, L. C. B.; PALAZZO Jr., R. **Existências de Códigos Corretores de Erros e Protocolos de Comunicação em Sequências de DNA.** [S.l.], 2011.

[2] LIRA, E.H.C. **Códigos corretores de Erros no Ensino Médio: Um estudo sobre o código de Hamming**.2018. 116f. Dissertação de Mestrado-Universiddae Federal Rural de Pernambuco, Recife,2018.

[3] BOLDRINI, J. L. **Álgebra linear.** 3.ed. ampl. e rev. São Paulo: HARBRA, 1986.

[4] LIN, S. COSTELLO, J.D. **Error Control Coding: Fundamentals and Applications**, Prentice Hall, 1983.

[5] MILIES, C. P. **Introdução à Teoria dos Códigos Corretores de Erros.** Colóquio de Matemática da Região Centro-Oeste: SBM, 2009. Disponível em www.sbm.org.br/docs/coloquios/CO-1-09.pdf. Ultimo acesso em 15/04/2020.

[6] STRANG, G. **Álgebra Linear e suas aplicações.** 4.ed. São Paulo: Cengage Learning, 2009.

[7] LIN, Shu. **Error control coding: fundamentals and applications.** Coautoria de Daniel J. Costello. 2nd ed. Upper Saddle River, New Jersey: Pearson/Prentice Hall, c2004.

[8] MENEGHESSO,**C. Códigos corretores de erros.** 2012. 61f. Trabalho de Conclusão de Curso – Faculdade de Matemática, Universidade Federal de São Carlos, São Carlos, 2012.