À minha irmã...

Agradecimentos

Seria injusto não agradecer às pessoas que de diferentes formas contribuiram para a vivência deste momento:

Ao meu orientador Professor Doutor César Silva pela grande competência e rigor com que me orientou e pelo apoio, disponibilidade e interesse manifestado;

Aos meus pais, por TUDO...;

Ao meu namorado por me respeitar todos os dias;

E por último, à minha irmã, por estar sempre presente.

Resumo

O presente trabalho tem como objectivo principal apresentar uma introdução à teoria de códigos detectores e correctores de erros, dando ênfase à teoria dos códigos lineares e à teoria dos códigos perfeitos. Discute-se ainda uma possível abordagem deste tema em contexto de sala de aula no ensino secundário.

Palavras-chave

Teoria de códigos; código linear; código perfeito; ensino.

Abstract

The present work has the principal objective of presenting an introduction to the theory of error detecting and correcting codes, emphasizing the theory of linear codes and the theory of perfect codes. It is also discussed a possible approach to this issue in the secondary schools.

Keywords

Code theory; linear code; perfect code; teaching.

Conteúdo

Introdução

É aceite de forma generalizada que a Teoria de Códigos enquanto disciplina teve início na década de quarenta do século vinte, estando à sua génese associada a investigação dos matemáticos Richard Wesley Hamming (1915 - 1998) e Claude Elwood Shannon (1916 - 2001) e do físico Marcel Golay (1902 - 1989).

No final da década de quarenta Hamming trabalhava no Laboratório Bell de Tecnologia. Nesta altura os computadores eram máquinas muito caras e Hamming apenas podia trabalhar com os computadores do laboratório aos fins de semana. Nessa época os programas estavam muito sujeitos a erros e, sempre que o computador detectava um erro num programa, este era interrompido passando o computador a ler o programa seguinte. Esta situação levou Hamming a desenvolver um código capaz de detectar até dois erros e corrigir um, se este for único. O seu artigo [7], publicado apenas em 1950 depois de aceite o pedido de patente dos códigos aí desenvolvidos, é considerado um dos artigos fundadores da Teoria de Códigos. Foi Hamming o responsável pela introdução de um conceito de distância entre palavras de código, pela introdução dos códigos hoje conhecidos como códigos de Hamming e por vários resultados sobre códigos perfeitos.

Também o matemático Claude Shannon teve um papel determinante no desenvolvimento inicial da Teoria de Códigos, em particular nas aplicações desta teoria à Criptografia. Além de ter tido um papel importante em Teoria de Códigos, Shannon é também conhecido como o pai da Teoria de Informação, sendo o seu artigo [11] de 1948, também publicado no "The Bell System Technical Journal", o artigo que deu origem a esta área de investigação, intimamente relacionada com os códigos.

A partir do artigo [11] de Shannon, Golay desenvolveu alguns códigos correctores de erros, ainda hoje muito usados em comunicações digitais. Estes códigos, hoje conhecidos como códigos Golay, surgiram no artigo [6], de 1949, considerado também um dos artigos mais marcantes da Teoria de Códigos. Por exemplo, a nave espacial Voyager, nas missões de 1979 e 1980, necessitou de um código muito eficiente, uma vez que foi necessário transmitir

muitas fotografias coloridas de Jupiter e Saturno, tal foi conseguido recorrendo a um código de Golay.

Actualmente muitas das ideias desenvolvidas em Teoria de Códigos são usadas nos mais diversos contextos, como por exemplo nos telemóveis, nos sistemas de processamento de imagens digitais, nas comunicações via satélite e nos sistemas de codificação de dados da internet.

A ideia base da Teoria de Códigos é possibilitar a transmissão de mensagens de forma segura e de tal modo que, mesmo que ocorram erros, estes sejam detectados e corrigidos. Um bom código deverá ser "grande"(ter muitas palavras) para transmitir tanta informação quanto possível e as palavras deverão ser tão pequenas quanto possível, a fim de facilitar a transmissão. O código deverá ainda ser capaz de detectar e corrigir um grande número de possíveis erros.

O presente trabalho constitui uma introdução à Teoria de Códigos detectores e correctores de erros, sendo ainda discutida a possibilidade de abordar o tema no contexto do ensino secundário.

O trabalho está estruturado do seguinte modo: no Capítulo 1 apresentamos as definições básicas da Teoria de Códigos e descrevemos o "problema fundamental da teoria dos códigos", que, de forma imprecisa, se pode dizer que é o problema de encontrar os códigos mais eficientes; seguidamente, no Capítulo 2, particularizamos a teoria para o caso dos códigos lineares, os quais possuem a vantagem de se poderem obter a partir de um pequeno número de palavras; no Capítulo 3 discutimos brevemente os códigos perfeitos que, num certo sentido a precisar, são os códigos mais eficientes; por fim, no Capítulo 4, discutimos uma possível abordagem da Teoria de Códigos no contexto do ensino secundário, apresentando uma proposta de uma aula introdutória ao tema. A fim de tornar este trabalho tão auto-contido quanto possível, incluímos ainda no final dois apêndices onde se incluem algumas definições e resultados básicos usados ao longo do texto.

Capítulo 1

Códigos, distâncias, detecção e correcção de erros

O objectivo principal deste capítulo é introduzir os conceitos básicos da Teoria de Códigos e apresentar alguns dos problemas que se colocam nesta teoria. Nomeadamente, na Secção 1.1 introduzimos a noção de alfabeto e o conceito de distância que é uma peça fundamental para discutir a detecção e correcção de erros e na Secção 1.2 abordamos o chamado "Problema Fundamental da Teoria de Códigos"que, de alguma forma, é o problema de determinar os códigos que conseguem codificar mais informação.

1.1 Alfabetos, erros e distância de Hamming

Designaremos um conjunto finito com q elementos, $F_q = \{\alpha_1, \ldots, \alpha_q\}$, por *alfabeto*. Em geral, nos exemplos que vamos considerar ao longo deste trabalho, tomaremos sempre $F_q = \{0, \ldots, q-1\}$ para algum $q \in \mathbb{N}$. Um conjunto $\mathcal{C}$ formado por sequências finitas (eventualmente de tamanhos variáveis) de elementos de F_q designa-se por *código q-ário*. Usualmente, um código 2-ário designa-se por *código binário* e um código 3-ário diz-se um código ternário. Designamos os elementos de um código por *palavras*.

Dado $n \in \mathbb{N}$, denotamos por F_q^n o conjunto das sequências de n elementos de F_q. Assim

$$F_q^n = \{a_1 \cdots a_n \colon a_i \in F_q\}.$$

Designamos um subconjunto de F_q^n por *código q-ário de comprimento n*. Deste modo um código q-ário de comprimento n é um conjunto de sequências de comprimento n de elementos de F_q. É imediato que F_q^n têm q^n elementos.

Neste trabalho consideraremos apenas códigos de comprimento fixo.

Exemplo 1.1.1. Fazendo $F_2 = \{0, 1\}$, é fácil verificar que

$$F_2^4 = \{0000, 0001, 0010, 0100, 1000, 0011, 0101, 1001, 0110,$$
$$1100, 0111, 1010, 1011, 1101, 1110, 1111\}.$$

Por exemplo, o conjunto $\mathcal{C} = \{0101, 1100, 1110\} \subset F_2^4$ é um código binário de comprimento 4 constituído pelas palavras 0101, 1100 e 1110.

Num dado código algumas palavras têm aparentemente mais semelhanças do que outras. Por exemplo, no código do Exemplo 1.1.1, a palavra 1110 parece estar "mais relacionada" com a palavra 1100 do que com a palavra 0101. Para podermos dar um sentido preciso a esta afirmação é necessário introduzir um conceito de distância entre as palavras que nos permita identificar quais as que se encontram mais próximas e quais as mais afastadas de uma determinada palavra. Concretamente, dadas duas palavras $a = a_1 \cdots a_n$, $b = b_1 \cdots b_n \in F_q^n$ define-se a *distância de Hamming* entre a e b, que denotamos por $d(a, b)$, como o número de índices $i \in \{1, \ldots, n\}$ tais que $a_i \neq b_i$, i.e.,

$$d(a, b) = \sharp\{i : a_i \neq b_i, 1 \leq i \leq n\},$$

onde $\sharp A$ designa o número de elementos do conjunto A. Assim, a distância de Hamming entre duas palavras é o número de posições em que estas diferem.

Exemplo 1.1.2. Consideremos o conjunto $F_2^3 = \{000, 001, 010, 100, 011, 101, 110, 111\}$. Temos

$$d(001, 000) = 1, \quad d(010, 100) = 2 \quad \text{e} \quad d(111, 000) = 3.$$

É de fácil verificação que a distância de Hamming é uma métrica em F_q^n. De facto tem-se o seguinte.

Proposição 1.1.3. Dados $a, b, c \in F_q^n$ verificam-se as propriedades:

a) $d(a, b) = 0$ se e só se $a = b$;

b) $d(a, b) = d(b, a)$;

c) $d(a, b) \leq d(a, c) + d(c, b)$.

Demonstração. a) Pela definição, $d(a, b) = 0$ verifica-se se e só se $a_i \neq b_i$ não se verifica para nenhum $i = 1, \ldots, n$, ou seja, se e só se $a_i = b_i$ para $i = 1, \ldots, n$ e portanto se e só se $a = b$.

b) Pela definição de distância de Hamming tem-se que para $a, b \in F_q^n$

$$d(a,b) = \sharp\{i : a_i \neq b_i, 1 \leq i \leq n\} = \sharp\{i : b_i \neq a_i, 1 \leq i \leq n\} = d(b,a).$$

c) Sejam $a, b, c \in F_q^n$. Verifica-se

$$\begin{aligned} d(a,c) + d(c,b) &= \sharp\{i : a_i \neq c_i, 1 \leq i \leq n\} + \sharp\{i : c_i \neq b_i, 1 \leq i \leq n\} \\ &= \sharp\{i : a_i \neq c_i \wedge a_i = b_i, 1 \leq i \leq n\} \\ &\quad + \sharp\{i : a_i \neq c_i \wedge a_i \neq b_i, 1 \leq i \leq n\} \\ &\quad + \sharp\{i : b_i \neq c_i \wedge a_i = b_i, 1 \leq i \leq n\} \\ &\quad + \sharp\{i : b_i \neq c_i \wedge a_i \neq b_i, 1 \leq i \leq n\} \\ &\geq \sharp\{i : a_i \neq c_i \wedge a_i \neq b_i, 1 \leq i \leq n\} \\ &\quad + \sharp\{i : b_i \neq c_i \wedge a_i \neq b_i, 1 \leq i \leq n\} \\ &\geq \{i : a_i \neq b_i, 1 \leq i \leq n\} = d(a,b) \end{aligned}$$

o que mostra o pretendido.

□

Dado um código $\mathcal{C}$, a *distância mínima* de $\mathcal{C}$ é o número

$$d = \min\{d(a,b) : a, b \in \mathcal{C},\ a \neq b\}.$$

Naturalmente, o conceito de distância mínima apenas está bem definido para códigos com mais do que uma palavra. Em todo o restante texto assumiremos sempre que os códigos considerados possuem mais do que uma palavra. Como veremos a seguir, quanto maior é a distância mínima melhor é o código num certo sentido.

Exemplo 1.1.4. Consideremos o código $\mathcal{C} = \{00, 10, 11\} \subseteq F_2^2$. Temos

$$d(00,10) = 1, \quad d(00,11) = 2 \quad \text{e} \quad d(10,11) = 1$$

pelo que a distância mínima para este código é $d = 1$.

Diz-se que um código $\mathcal{C}$ é *s-detector de erros* se detecta qualquer combinação de s erros em qualquer palavra. Diz-se que $\mathcal{C}$ é *t-corrector de erros* se permite corrigir qualquer combinação de t erros em qualquer palavra. Dada uma palavra $a_1 \cdots a_n$ de um código $\mathcal{C}$ que é transmitida, dizemos que $a_1 \cdots a_n$ sofreu um *erro de substituição* se for recebida como $b_1 \cdots b_n$ onde $b_j \neq a_j$ para algum $j \in \{1, \ldots, n\}$ e $b_i = a_i$ para $i \in \{1, \ldots, n\}, i \neq j$. Dizemos ainda que $a_1 \cdots a_n$ sofreu um *erro de transposição* se for recebida como $b_1 \cdots b_n$

onde $b_j = a_k$ e $b_k = a_j$ para alguns $j, k \in \{1, \ldots, n\}$ com $j \neq k$ e $a_j \neq a_k$ e ainda $b_i = a_i$ para $i \in \{1, \ldots, n\}, i \neq j, k$. De forma análoga, dada uma palavra $a_1 \cdots a_n$ de um código $\mathcal{C}$ que é transmitida, dizemos que $a_1 \cdots a_n$ sofreu um ℓ *erros de substituição* se a palavra recebida for $b_1 \cdots b_n$ onde $b_{j_i} \neq a_{j_i}$ para alguns $j_1, \ldots, j_\ell \in \{1, \ldots, n\}$ e $b_i = a_i$ para $i \in \{1, \ldots, n\} \setminus \{j_1, \ldots, j_\ell\}$. Dizemos que $a_1 \cdots a_n$ sofreu um ℓ *erros de transposição* se a palavra recebida for $b_1 \cdots b_n$ onde $b_{j_i} = a_{k_i}$ e $b_{k_i} = a_{j_i}$ para alguns $j_1 \ldots, j_\ell, k_1, \ldots, k_\ell \in \{1, \ldots, n\}$, com $j_i \neq k_i$ e $a_{j_i} \neq a_{k_i}$ para $i \in \{1, \ldots, \ell\}$ e $b_i = a_i$ para $i \in \{1, \ldots, n\} \setminus \{j_1, \ldots, j_\ell, k_1, \ldots, k_\ell\}$.

No próximo resultado apresentam-se algumas majorações relacionadas com a distância mínima de um código. Designamos por $\lfloor a \rfloor$ o maior inteiro menor ou igual a a e por $\lceil a \rceil$ o menor inteiro maior ou igual a a. Temos naturalmente $\lfloor a \rfloor \leq a \leq \lceil a \rceil$.

Teorema 1.1.5. Seja d a distância mínima do código $\mathcal{C}$. Temos que

i) O código $\mathcal{C}$ detecta s erros de substituição em qualquer palavra se e só se $d \geq s + 1$;

ii) O código $\mathcal{C}$ corrige t erros de substituição em qualquer palavra se e só se $d \geq 2t + 1$;

iii) O código $\mathcal{C}$ pode ser usado para detectar até $d - 1$ erros de substituição;

iv) O código $\mathcal{C}$ pode ser usado para corrigir até $\left\lfloor \frac{d-1}{2} \right\rfloor$ erros de substituição.

Demonstração. i) Sejam a e b duas palavras num código $\mathcal{C}$ tais que $d(a, b) = d$. Se ao enviar a palavra a ocorrerem d erros que façam com que se receba b, esses erros nunca serão detectados. Portanto, se $\mathcal{C}$ detecta s erros em qualquer palavra então $s < d$. Por outro lado, se $d \geq s + 1$, se for codificada a palavra a e se for recebida uma palavra b com $d(a, b) = r \leq s < d$ podemos reconhecer que b não pertence ao código e portanto detectar o erro.

ii) Sejam a e b duas palavras de um código $\mathcal{C}$ tais que $d(a, b) \leq 2t$. Isto implica que as duas palavras a e b diferem no máximo em $2t$ posições. Se nessas $2t$ posições ocorrerem t erros ao ser transmitida a palavra a podemos obter uma palavra que está a uma distância inferior ou igual a t de b e deste modo não conseguimos corrigir o erro. Concluímos que $d > 2t$, o que é equivalente a $d \geq 2t + 1$, uma vez que d é um número inteiro. Por outro lado, se $d \geq 2t+1$ e ao transmitir uma palavra a ocorrerem um número de erros inferior ou igual a t, a palavra resultante está a uma distância de a inferior ou igual a t e, atendendo à distância mínima, a uma distância de qualquer outra palavra superior ou igual a $t + 1$. Podemos portanto identificar a palavra como

sendo a uma vez que esta é a palavra do código mais próxima da palavra recebida e assim corrigir o erro.

iii) Basta atender à alínea i) e notar que

$$d \geq s+1 \;\Leftrightarrow\; s \leq d-1.$$

iv) Basta atender à alínea ii) e notar que

$$d \geq 2t+1 \;\Leftrightarrow\; t \leq \frac{d-1}{2}$$

e portanto, como $t \in \mathbb{N}$, tem-se $t \leq \left\lfloor \frac{d-1}{2} \right\rfloor$.

□

Podemos também enunciar um resultado semelhante ao Teorema 1.1.5 para erros de transposição.

Teorema 1.1.6. Seja d a distância mínima do código $\mathcal{C}$. Temos que

i) Se $d \geq 2s+1$ então o código $\mathcal{C}$ detecta s erros de transposição em qualquer palavra.

ii) Se $d \geq 4t+1$ então o código $\mathcal{C}$ corrige t erros de transposição em qualquer palavra.

iii) O código $\mathcal{C}$ pode ser usado para detectar até $\lfloor \frac{d-1}{2} \rfloor$ erros de transposição;

iv) O código $\mathcal{C}$ pode ser usado para corrigir até $\lfloor \frac{d-1}{4} \rfloor$ erros de transposição.

Demonstração. i) Sejam a e b duas palavras do código $\mathcal{C}$ e seja $d \geq 2s+1$. Se for codificada a palavra a e se ocorrerem $r \leq s$ erros de transposição de tal forma que seja recebida uma palavra b, é imediato que

$$d(a,b) \leq 2r \leq 2s < 2s+1.$$

Como b dista menos que $2s+1$ de a concluímos que b não pertence ao código e portanto o erro é detectado.

ii) Sejam a e b duas palavras do código $\mathcal{C}$. Seja $d \geq 4t+1$. Se ao transmitir uma palavra a ocorrer um número de erros de transposição inferior ou igual a t, a palavra resultante está a uma distância de a inferior ou igual a $2t$ e, atendendo à distância mínima, a uma distância de qualquer outra palavra superior ou igual a $2t+1$. Podemos portanto identificar a palavra como sendo a uma vez que esta é a palavra do código mais próxima da palavra recebida o que permite corrigir o erro.

iii) Basta atender à alínea i) e notar que

$$d \geq 2s+1 \Leftrightarrow s \leq \frac{d-1}{2}$$

e portanto, como $s \in \mathbb{N}$, tem-se $s \leq \left\lfloor \frac{d-1}{2} \right\rfloor$.

iv) Basta atender à alínea ii) e notar que

$$d \geq 4t+1 \Leftrightarrow t \leq \frac{d-1}{4}$$

e portanto, como $t \in \mathbb{N}$, tem-se $t \leq \left\lfloor \frac{d-1}{4} \right\rfloor$.

□

O Teorema 1.1.5 motiva a seguinte definição: dado um código $\mathcal{C}$ com distância mínima d, designamos o número

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

por *capacidade* do código C. Assim, a capacidade de um código é o número máximo de erros de substituição que ele permite corrigir.

Exemplo 1.1.7. Consideremos o código $\mathcal{C} = \{1000, 2000, 3000, 4000\}$. A distância entre quaisquer duas palavras código é 1. No entanto, se ocorrer um erro de transposição, isto é se trocarmos a primeira posição com outra, o erro é detectado. Temos apesar disso $d < 2 \times 1 + 1 = 3$. Concluímos que não existe equivalência em i) do Teorema 1.1.6.

Exemplo 1.1.8. Consideremos o código

$$C = \{000, 100, 101, 111\}.$$

É imediato que este código permite detectar 1 erro de transposição em qualquer palavra (uma vez que o número de zeros em cada palavra é distinto e mantém-se no caso de erros de transposição) no entanto a distância mínima é 1. Verifica-se assim que, nem mesmo para códigos binários, temos equivalência em i) do Teorema 1.1.6.

Exemplo 1.1.9. Consideremos o código

$$C = \{1010, 2121, 3232, 0303\}.$$

É imediato que este código permite corrigir 2 erros de transposição em qualquer palavra (uma vez que qualquer palavra possui sempre um dígito que não é comum a uma outra palavra dada) no entanto a distância mínima é 4. Verifica-se assim que não temos equivalência em ii) do Teorema 1.1.6. Note-se que este código permite ainda detectar três erros de substituição ou corrigir um erro de substituição em qualquer palavra.

Um dos objectivos na construção de um código é conseguir obter uma distância mínima elevada. No entanto, uma distância mínima elevada vai fazer com que o comprimento das palavras aumente e consequentemente dificultar a transmissão da mensagem.

Um *código q-ário* (n, M, d) que se designa muitas vezes por código $(n, M, d)_q$ é um código com M palavras de tamanho n, constituídas por caracteres de um alfabeto com q símbolos, e que estão a uma distância (de Hamming) mínima d umas das outras.

Exemplo 1.1.10. Usando apenas os caracteres 0 e 1 pretendemos codificar a mensagem $APRIPRA$ em que P significa "parar", R significa "retroceder", A significa "avançar" e I significa "iniciar".

Definimos o seguinte código:

$$I = 00,\ R = 01,\ A = 10 \quad \text{e } P = 11.$$

Trata-se portanto de um código $(2, 4, 1)_2$. Ou seja, é um código binário (constituído por dois símbolos) com quatro palavras de comprimento dois e onde a distância mínima é 1. Vejamos:

$$d(00, 01) = 1, \quad d(00, 10) = 1, \quad d(00, 11) = 2$$
$$d(11, 01) = 1 \quad d(11, 10) = 1 \quad \text{e} \quad d(10, 01) = 2.$$

A mensagem codificada fica 10110100110110. Imaginemos que há um engano ao escrever a mensagem e nas posições três e quatro, em vez de 11 escrevemos 10 obtendo a mensagem 10100100110110. Esse erro não é detectado, apenas estamos a alterar o significado da mensagem: em vez de parar, enviamos a mensagem avançar. Verificámos portanto que, neste caso, o código não detecta erros de substituição. Note-se que já sabíamos isto à partida, atendendo a que o código tem distância mínima 1 e à alínea i) do Teorema 1.1.5. Se nos enganarmos novamente e em vez de 10 escrevermos 01 nas posições um e dois obtendo a mensagem 01110100110110 (trocámos as duas posições) o erro continua a não ser detectado porque 01 continua a ter significado. Verifica-se assim que o código não detecta erros de transposição. Note-se que temos neste caso $(2, 4, 1)_2 = F_2^2$.

Exemplo 1.1.11. Consideremos agora um código $(4, 4, 2)_2$ constituído pelas palavras

$$I = 0100,\ R = 1000,\ A = 0001 \quad \text{e} \quad P = 0010.$$

Este código pode também ser usado para codificar a mensagem $APRIPRA$ do exemplo 1.1.10. Usando este código a mensagem fica 0001001010000100001010000001. Agora

obteve-se um código mais extenso porque as palavras são maiores. Supondo que nos enganamos e em vez de 1000 escrevemos 1010, o erro já é detectado porque não existe correspondência para a palavra 1010. Ou seja, escrevemos uma palavra que está à distância 1 e neste código a distância mínima é 2. De facto, como sabemos pela alínea i) do Teorema 1.1.5, atendendo a que a distância mínima é superior ou igual a dois, o código detecta um erro de substituição. Mas, se nos enganarmos e em vez de 0100 escrevermos 0010 o erro não é detectado porque 0010 tem correspondência no código. Portanto, este código ainda não detecta erros de transposição.

Exemplo 1.1.12. Suponhamos agora um código $(3, 4, 3)_3$ em que

$$I = 0002,\ R = 0220,\ A = 2110 \quad \text{e} \quad P = 1122.$$

Mais uma vez podemos codificar a mensagem $APRIPRA$ dos Exemplos 1.1.10 e 1.1.11. Tem-se neste caso 21101122022000021122022021 10.

Este código já detecta erros de transposição. Suponhamos por exemplo que em vez de escrevermos 0002 escrevemos 0020. Houve troca de dígitos nas duas últimas posições mas o erro é detectado porque não existe correspondência para 0020. Podemos verificar que o mesmo sucede com quaisquer duas palavras deste código e mesmo que a transposição ocorra entre duas palavra distintas. Verifica-se que este código já detecta erros de transposição. Note-se que a distância mínima neste caso é 3.

Consideraremos agora um tipo de códigos no qual as palavras estão tão afastadas quanto é possível.

Exemplo 1.1.13 (Códigos de repetição)**.** Dados números naturais q e n, definimos o *código de repetição de tamanho n com q símbolos*, $\mathcal{R}_{n,q}$, por

$$\mathcal{R}_{n,q} = \{a_1 \dots a_n : a_1 \in F_q \ \wedge \ a_1 = \cdots = a_n\}.$$

Qualquer palavra pertencente a este código tem as posições todas iguais. Assim, o número de palavras existente é igual ao número de símbolos e portanto $M = q$. Como os dígitos são todos iguais, a distância mínima entre quaisquer duas palavras é exactamente igual ao comprimento de cada palavra, ou seja, $d = n$. Portanto, concluímos que o $\mathcal{R}_{n,q}$ é um código $(n, q, n)_q$.

A título de exemplo, consideremos o código de repetição binário

$$\mathcal{R}_{3,2} = \{000, 111\}.$$

Como se trata de um código binário e como as palavras pertencentes ao código têm os dígitos todos iguais então existem duas palavras. Sabemos também que $d(000, 111) = 3$ que é igual ao comprimento das palavras portanto, como já vimos acima, $\mathcal{R}_{3,2}$ é um código $(3, 2, 3)_2$.

1.2 O Problema Fundamental da Teoria dos Códigos

Dois códigos dizem-se *equivalentes* se um pode ser obtido do outro através de uma combinação das seguintes operações:

i) Permutação das posições dos elementos do código

ii) Permutação de símbolos aparecendo numa posição fixa

Reconhecemos facilmente que dois códigos equivalentes permitem transmitir exactamente a mesma informação.

Exemplo 1.2.1. O código $\mathcal{C}_1 = \{120, 012, 000\}$ e o código $\mathcal{C}_2 = \{012, 201, 222\}$ são equivalentes uma vez que a permutação

$$\begin{array}{ccc} 1 & 2 & 0 \\ \downarrow & \downarrow & \downarrow \\ 0 & 1 & 2 \end{array}$$

faz corresponder um deles ao outro.

Como vimos nos exemplos anteriores, para encontrarmos códigos mais "fiáveis" tivemos de aumentar q (o número de símbolos), n (o tamanho das palavras) e d (a distância mínima). Tal originou códigos mais extensos, o que torna a sua transmissão mais complicada. Este é o preço a pagar por mais fiabilidade. Em Teoria de Códigos temos sempre de encontrar um equilíbrio entre fiabilidade e facilidade de transmissão.

Assim, um bom código deve ter as seguintes características:

- n pequeno para ser mais fácil enviar mensagens;
- M grande para ter capacidade de enviar muita informação;
- d grande para ter capacidade de detectar e corrigir erros.

Encontrar um ponto de equilíbrio entre n, M e d constitui o chamado *Problema Fundamental da Teoria de Códigos.*

Para q fixo, denotamos por $A_q(n,d)$ o maior valor possível de M para uma dada distância mínima d e um dado n. Assim, para q, n e d dados, um número de palavras igual a $A_q(n,d)$ maximiza a capacidade de enviar informação. Obter o valor exacto ou mesmo estimar $A_q(n,d)$ para q, n e d dados é uma tarefa em geral complicada, no entanto para $d=1$ e $d=n$ estes valores são fáceis de obter.

Teorema 1.2.2. Para quaisquer $q, n \in \mathbb{N}$ temos

i) $A_q(n,1) = q^n$;

ii) $A_q(n,n) = q$.

Demonstração. i) Se $d=1$ não temos qualquer restrição, uma vez que a distância entre duas palavras de código diferentes é sempre pelo menos 1. Assim, o código de maior comprimento n com distância mínima 1 é F_q^n, que tem q^n elementos.

ii) Seja C um código $(n,M,n)_q$. Como as palavras estão a distância mínima n, quaisquer duas palavras distintas diferem em todas as posições. Em particular, a primeira posição de cada palavra deve ser diferente. Assim, $M \leq q$ o que implica que $A_q(n,n) \leq q$. No entanto, sabemos que o código de repetição $\mathcal{R}_{n,q}$ do Exemplo 1.1.13 é um código $(n,q,n)_q$ e portanto $A_q(n,n) \geq q$. Assim conclui-se que $A_q(n,n) = q$.

□

Verifica-se ainda a seguinte relação.

Teorema 1.2.3. Para todo o $n \geq 2$ e todo o $d \leq n$ temos

$$A_q(n,d) \geq q\, A_q(n-1,d).$$

Demonstração. Seja $C = \{c_1^1 \cdots c_{n-1}^1, \ldots, c_1^M \cdots c_{n-1}^M\}$ um código $(n-1,M,d)_q$. Construímos a partir deste o código

$$C' = \{c_1^1 \cdots c_{n-1}^1 0, \ldots, c_1^1 \cdots c_{n-1}^1 (q-1), \ldots, c_1^M \cdots c_{n-1}^M 0, \ldots, c_1^M \cdots c_{n-1}^M (q-1)\}.$$

Assim cada palavra $c_1^i \cdots c_{n-1}^i$ do código C, dá origem a q palavras de C' da forma $c_1^i \cdots c_{n-1}^i d$ com $d \in \{0, \cdots, q-1\}$.

Naturalmente C' é um código q-ário com palavras de tamanho n e tem qM palavras. Por outro lado, dadas as palavras $c_i^1 \ldots c_{n-1}^i d_1$ e $c_j^1 \ldots c_{n-1}^j d_2$, temos

$$d(c_i^1 \ldots c_{n-1}^i d_1,\, c_i^1 \ldots c_{n-1}^i d_2) = \begin{cases} d(c_i^1 \ldots c_{n-1}^i,\, c_j^1 \ldots c_{n-1}^j) & \text{se } d_1 = d_2 \\ d(c_i^1 \ldots c_{n-1}^i,\, c_j^1 \ldots c_{n-1}^j) + 1 & \text{se } d_1 \neq d_2 \end{cases}.$$

Temos então que C' é um código $(n, qM, d)_q$. Conclui-se que $A_q(n,d) \geq qM = qA_q(n-1,q)$, o que completa a demonstração. □

No caso de códigos binários podemos dizer um pouco mais do que no caso geral. Começamos com um exemplo.

Exemplo 1.2.4. Consideremos o código binário

$$\mathcal{C} = \{00000, 11100, 00111, 11011\}.$$

É fácil verificar que este código é um código $(5,4,3)_2$. Vejamos como construir a partir deste um código com palavras à distância mínima 4, aumentando apenas em uma posição o tamanho das palavras, ou seja, construiremos a partir deste um código $(6,4,4)_2$ que denotaremos por $\mathcal{C}'$. Sejam $\mathcal{C}_1 = 00000, \mathcal{C}_2 = 11100, \mathcal{C}_3 = 00111, \mathcal{C}_4 = 11011$ e $\mathcal{C}'_1, \mathcal{C}'_2, \mathcal{C}'_3, \mathcal{C}'_4$ as quatro palavras do código $\mathcal{C}'$. Ora, ao aumentarmos um dígito em cada palavra temos de ter em conta que queremos $d = 4$. Definimos $\mathcal{C}'_1 = 000000$, ou seja, adicionou-se o símbolo "0". Já não podemos criar $\mathcal{C}'_2$ adicionando "0" a $\mathcal{C}_2$ porque a distância mínima entre $\mathcal{C}'_1$ e $\mathcal{C}'_2$ seria 3. Portanto, $\mathcal{C}'_2 = 111001$. Da mesma forma, construímos $\mathcal{C}'_3 = 001110$ e $\mathcal{C}'_4 = 110111$ e obtemos um código $(6,4,4)_2$. Poder-se-ia também ter começado por adicionar o digito "1" em $\mathcal{C}'_1$ e seguindo um raciocínio semelhante obteríamos um código $(6,4,4)_2$.

Portanto, tínhamos um código com distância mínima três (ímpar) e ao aumentar em uma posição o tamanho das palavras, obtivemos um código com distância mínima quatro. Ora, ao repararmos no exemplo anterior podemos concluir que, quando não temos distância ímpar, a construção anterior não funciona.

Podemos generalizar o exemplo anterior para qualquer código binário com distância mínima ímpar. Para tal, dado um código binário $\mathcal{C} = (n, M, d)_2$ e uma palavra $a = a_1 \cdots a_n \in \mathcal{C}$ definimos o *peso* de a, $\omega(a)$, por

$$\omega(a) = \sharp\{i \in \{1, \ldots, n\} : a_i = 1\}.$$

Dadas duas palavras $a, b \in (F_2)^n$ definimos ainda a soma de a com b por $a + b = c_1 \cdots c_n$ onde $c_i = a_i + b_i \mod 2$.

Assim, por exemplo em $(F_2)^4$, temos $\omega(1101) = 3$, $\omega(1001) = 2$ e, uma vez que $1101 + 1001 = 0100$, $\omega(1101 + 1001) = \omega(0100) = 1$.

O conceito de peso de uma palavra é muito útil uma vez que podemos usá-lo para obter a distância mínima num código binário. Como veremos mais à frente o mesmo

sucede num código q-ário, desde que q seja uma potência de um número primo. De facto, dados $a, b \in (n, M, d)_2$ temos

$$\begin{aligned} d(a,b) &= \sharp\{i \in \{1,\ldots,n\} : a_i \neq b_i\} \\ &= \sharp\{i \in \{1,\ldots,n\} : a_i - b_i \neq 0\} \\ &= \sharp\{i \in \{1,\ldots,n\} : a_i - b_i = 1 \mod 2\} \\ &= \omega(a+b). \end{aligned} \tag{1.1}$$

Esta caracterização da distância de Hamming, válida como foi visto no caso particular de códigos binários, permite-nos obter o seguinte resultado.

Teorema 1.2.5. Seja d um número ímpar. Então existe um código binário $(n, M, d)_2$ se e só se existe um código binário $(n+1, M, d+1)_2$, isto é

$$A_2(n,d) = A_2(n+1, d+1).$$

Demonstração. Seja $\mathcal{C}$ um código binário $(n, M, d)_2$ e consideremos o código binário $\mathcal{C}'$, com palavras de tamanho $n+1$, dado por

$$\mathcal{C}' = \{x_1 \cdots x_n x_{n+1} : x_1 \cdots x_n \in \mathcal{C} \ \wedge \ x_{n+1} = \sum_{i=1}^{n} x_i \mod 2\}.$$

Começamos por mostrar que todas as palavras de $\mathcal{C}'$ têm peso par. De facto, dada $x = x_1 \cdots x_{n+1} \in \mathcal{C}'$ temos

$$\begin{aligned} \omega(x) &= \sharp\{i \in \{1,\ldots,n+1\} : x_i = 1\} \\ &= \sharp\{i \in \{1,\ldots,n\} : x_i = 1\} + \begin{cases} 1 & \text{se } \sum_{i=1}^{n} x_i = 1 \mod 2 \\ 0 & \text{se } \sum_{i=1}^{n} x_i = 0 \mod 2 \end{cases} \\ &= \sharp\{i \in \{1,\ldots,n\} : x_i = 1\} + \begin{cases} 1 & \text{se } \omega(x_1 \cdots x_n) \text{ é ímpar} \\ 0 & \text{se } \omega(x_1 \cdots x_n) \text{ é par} \end{cases}. \end{aligned}$$

Deste modo $w(x_1 \cdots x_{n+1})$ fica par. Sejam $x', y' \in \mathcal{C}'$. Uma vez que todas as palavras do código $\mathcal{C}'$ têm peso par, $d(x', y') = \omega(x' + y')$ é par, o que implica que a distância mínima d' do código $\mathcal{C}'$ também seja par. Como $d \leq d' \leq d+1$ e d é ímpar então $d' = d+1$ e portanto C' é um código $(n+1, M, d+1)_2$.

Reciprocamente, seja $\mathcal{T}$ um código binário $(n+1, M, d+1)_2$. Consideremos duas palavras $x, y \in \mathcal{T}$ tais que $d(x,y) = d+1$. Consideramos uma posição k em que x e y

difiram. Apagando essa posição de todas as palavras pertencentes a $\mathcal{T}$ obtemos um código $(n, M, d)_2$. De facto, o código obtido não é um código com distância mínima superior a d porque as palavras que se obtém de x e y apagando a posição k estão a distância exactamente d e não é um código com distância mínima inferior a d porque apagando apenas a posição k nas palavras que estavam a uma dada distância $s \geq d+1$ obtemos palavras a distância $s \geq d+1$ se na posição k as palavras têm o mesmo símbolo ou a distância $s-1 \geq d$ se na posição k as palavras têm símbolos distintos. □

Vamos agora considerar os conceitos de bola fechada, bola aberta e esfera em F_q^n, os quais nos ajudam mais facilmente a contar o número de palavras que se encontram a uma determinada distância de uma palavra dada.

Como a distância de Hamming é uma métrica em F_q^n, para qualquer palavra $a \in F_q^n$ e para qualquer inteiro $r \geq 0$, está definida a *bola fechada de raio r e centro a*

$$B_r^n[a] = \left\{ b \in F_q^n : d(a,b) \leq r \right\},$$

a *bola aberta de raio r e centro a*

$$B_r^n(a) = \left\{ b \in F_q^n : d(a,b) < r \right\}$$

e a *esfera de raio r e centro a*

$$S_r^n(a) = \left\{ b \in F_q^n : d(a,b) = r \right\}.$$

Teorema 1.2.6. Verifica-se o seguinte:

1. Se $0 \leq r \leq n$, a bola fechada $B_r^n[a]$ contém

$$\binom{n}{0} + \binom{n}{1}(q-1) + \binom{n}{2}(q-1)^2 + ... + \binom{n}{\lfloor r \rfloor}(q-1)^{\lfloor r \rfloor}$$

palavras.

2. Se $r > n$ temos que $B_r^n[a] = F_q^n$ e portanto $B_r^n[a]$ contém q^n palavras.

Demonstração. Sejam $a, b \in F_q^n$. Fixemos a e consideremos todas as palavras b que verificam $d(a,b) = m$ $(m \leq n)$. Existem exactamente m posições em que a e b diferem e essas m posições podem ser escolhidas de $\binom{n}{m}$ maneiras. Para além disso, uma vez que estamos a considerar um código q-ário, a primeira posição de b que é diferente de a pode ser escolhida de $q-1$ maneiras, sendo ainda a liberdade de escolha igual para quaisquer posições em que a e b difiram. Portanto, o número de palavras b a distância exactamente

m de a é dado por $\binom{n}{m}(q-1)^m$, o que implica que se $0 \leq r \leq n$, a bola fechada $B_r^n[a]$ contém

$$\binom{n}{0} + \binom{n}{1}(q-1) + \binom{n}{2}(q-1)^2 + \cdots + \binom{n}{\lfloor r \rfloor}(q-1)^{\lfloor r \rfloor}$$

palavras e mostra 1..

Por outro lado, se $r > n$ temos que $B_r^n[a] = F_q^n$. Como, F_q^n tem q^n elementos conclui-se trivialmente que $B_r^n[a]$ tem exactamente q^n palavras e fica demonstrado 2.. □

Da demonstração do Teorema 1.2.6 concluímos imediatamente que $B_r^n(a)$ tem

$$\binom{n}{0} + \binom{n}{1}(q-1) + \binom{n}{2}(q-1)^2 + \cdots + \binom{n}{\lceil r-1 \rceil}(q-1)^{\lceil r-1 \rceil}$$

elementos se $r \leq n$ e q^n elementos se $r > n$ e que $S_r^n(a)$ tem

$$\binom{n}{r}(q-1)^r$$

elementos se $r \in \{0, \ldots, n\}$ e, caso contrário, isto é se $r \in \mathbb{R}^+ \setminus \{0, \ldots, n\}$, $S_r^n(a)$ não tem elementos. Deste modo é imediato que, para $r \in \mathbb{R}^+$, temos

$$B_r^n(a) = \bigcup_{i=0}^{\lceil r-1 \rceil} S_i^n(a) \quad \text{e} \quad B_r^n[a] = \bigcup_{i=0}^{\lfloor r \rfloor} S_i^n(a).$$

Exemplo 1.2.7. Consideremos F_2^4. Já sabemos que o conjunto tem $2^4 = 16$ elementos. Agora, calculemos os elementos deste conjunto com base no Teorema 1.2.6. Ora, sabemos que para quaisquer duas palavras $a, b \in (F_2)^4$, temos $d(a,b) \leq 4$. Portanto $r = 4$ e assim temos,

$$\binom{4}{0} + \binom{4}{1}(2-1) + \binom{4}{2}(2-1)^2 + \binom{4}{3}(2-1)^3 + \binom{4}{4}(2-1)^4 = 1 + 4 + 6 + 4 + 1 = 16.$$

Exemplo 1.2.8. Vejamos agora, por exemplo, o número de palavras de uma bola fechada, de uma bola aberta e de uma esfera com centro numa palavra de F_2^4 e raio 3. Temos pelo Teorema 1.2.6 e pelas considerações que o precedem que o número de palavras da bola fechada

$$B_3^4[a] = \left\{ b \in F_2^4 : d(a,b) \leq 3 \right\},$$

é exactamente

$$\binom{4}{0} + \binom{4}{1}(2-1) + \binom{4}{2}(2-1)^2 + \binom{4}{3}(2-1)^3 = 15,$$

da bola aberta

$$B_3^4(a) = \left\{ b \in F_2^4 : d(a,b) < 3 \right\}$$

é

$$\binom{4}{0} + \binom{4}{1}(2-1) + \binom{4}{2}(2-1)^2 = 11$$

e da esfera

$$S_3^4(a) = \left\{ b \in F_2^4 : d(a,b) = 4 \right\}.$$

é

$$\binom{4}{3}(2-1)^3 = 4.$$

Note-se ainda por exemplo que o número de palavras da bola aberta $B_{7/2}^4(a)$ é

$$\binom{4}{0} + \binom{4}{1}(2-1) + \binom{4}{2}(2-1)^2 + \binom{4}{3}(2-1)^3,$$

uma vez que $\lceil 7/2 - 1 \rceil = \lceil 5/2 \rceil = 3$.

Teorema 1.2.9 (Limite superior de Hamming)**.** Um código q-ário (n, M, d) satisfaz

$$M \left\{ \binom{n}{0} + \binom{n}{1}(q-1) + \cdots + \binom{n}{\kappa}(q-1)^\kappa \right\} \leq q^n, \tag{1.2}$$

onde κ designa a capacidade do código. Assim temos

$$A_q(n,d) \leq \frac{q^n}{\sum_{i=0}^{\lfloor \frac{d-1}{2} \rfloor} \binom{n}{i}(q-1)^i}. \tag{1.3}$$

Demonstração. Seja $\mathcal{C}$ um código q-ário $(n, M, d)_q$ e $a, b \in \mathcal{C}$. Pelo Teorema 1.2.6, sabemos que qualquer bola fechada de centro numa palavra de código com raio κ contém exactamente

$$\binom{n}{0} + \binom{n}{1}(q-1) + \cdots + \binom{n}{\kappa}(q-1)^\kappa$$

palavras. Como quaisquer duas palavras de código estão a distância igual ou superior a d, quaisquer duas bolas fechadas de raio κ centradas em palavras distintas não podem ter palavras comuns. De facto, suponhamos por contradição que a e b são duas palavras de código distintas tais que $B_\kappa[a] \cap B_\kappa[b] \neq \emptyset$. Então existe $c \in B_\kappa[a] \cap B_\kappa[b]$. Temos pela desigualdade triangular

$$d(a,b) \leq d(a,c) + d(c,b) \leq \kappa + \kappa = 2\kappa$$

o que contradiz o seguinte facto:

$$\kappa = \left\lfloor \frac{d-1}{2} \right\rfloor \leq \frac{d-1}{2} < \frac{d}{2} \quad \Rightarrow \quad d > 2\kappa.$$

Portanto, se o código tiver M palavras, concluímos que existem pelo menos

$$M\left\{\binom{n}{0}+\binom{n}{1}(q-1)+\cdots+\binom{n}{\kappa}(q-1)^{\kappa}\right\}$$

elementos de F_q^n distintos. Naturalmente, este número tem de ser inferior ou igual ao número total de palavras de F_q^n: q^n. Este facto permite obter a estimativa (1.2). Notando que $M \leq A_q(n,d)$ obtemos (1.3). □

Um código que satisfaz a igualdade em (1.2) diz-se um *código perfeito*.

Note-se que a estimativa em (1.3) indica explicitamente uma majoração para o número de palavras de um código, fixados os parâmetros q, n e d. Muitas vezes esta estimativa fica muito longe do valor de $A_q(n,d)$.

Exemplo 1.2.10. Por exemplo, para o código $(11, M, 3)_2$ podemos facilmente calcular uma majoração. Vejamos,

$$M\left\{\binom{11}{0}+\binom{11}{1}\right\} \leq 2^{11} \quad \Leftrightarrow \quad 12M \leq 2048 \quad \Leftrightarrow \quad M \leq \frac{512}{3}.$$

No entanto, segundo a tabela da página 14 de [3], sabemos que $144 \leq A_2(11,3) \leq 158$. Portanto, podemos concluir que neste caso a estimativa fica muito longe do valor de $A_2(11,3)$ visto que $\frac{512}{3} \cong 170,7$.

Exemplo 1.2.11. Por exemplo para o código binário $(7, M, 3)_2$ temos

$$M\left\{\binom{7}{0}+\binom{7}{1}\right\} \leq 2^{7} \quad \Leftrightarrow \quad 8M \leq 128 \quad \Leftrightarrow \quad M \leq 16.$$

De acordo com a tabela da página 14 de [3] sabemos que $A_2(7,3) = 16$, pelo que neste caso a estimativa é óptima.

Podemos estabelecer ainda um limite inferior para o número máximo de palavras de um código q-ário dadas a distância mínima e o tamanho das palavras, como veremos de seguida.

Teorema 1.2.12 (Limite inferior de Gilbert-Varshamov)**.** Um código q-ário perfeito (n, M, d) satisfaz

$$M\left\{\binom{n}{0}+\binom{n}{1}(q-1)+\cdots+\binom{n}{d-1}(q-1)^{d-1}\right\} \geq q^n. \tag{1.4}$$

Assim temos

$$A_q(n,d) \geq \frac{q^n}{\sum_{i=0}^{d-1}\binom{n}{i}(q-1)^i}. \tag{1.5}$$

Demonstração. Seja C um código $(n, M, d)_q$ perfeito. Assim $M = A_q(n, d)$. Como C é perfeito, dada $a_1 \cdots a_n \in F_q^n$ existe $c_1 \cdots c_n \in C$ tal que $d(c_1 \cdots c_n,\ a_1 \cdots a_n) < d$. De facto, caso contrário, juntando a palavra $a_1 \cdots a_n$ ao código C obteríamos um código $(n, M+1, d)_q$ e portanto teríamos $M = A_q(n, d) \geq M + 1$, o que é absurdo.

Deste modo, qualquer palavra $a_1 \cdots a_n$ está numa bola fechada de centro em alguma palavra de código e raio $d - 1$ e portanto

$$F_q^n \subseteq \bigcup_{c \in C} B_{d-1}^n[c].$$

Como $B_{d-1}^n[c]$ possui

$$\sum_{r=0}^{d-1} \binom{n}{r} (q-1)^r$$

elementos, existem M palavras de código e F_q^n tem q^n elementos, conclui-se que

$$M \sum_{r=0}^{d-1} \binom{n}{r} (q-1)^r \geq q^n. \tag{1.6}$$

e verifica-se (1.4). Atendendo a que o código é perfeito, substituindo M por $A_q(n, d)$ em (1.6) verifica-se (1.5). □

Teorema 1.2.13 (Limite superior de Singleton)**.** Um código q-ário (n, M, d) satisfaz

$$M q^{d-1} \leq q^n \tag{1.7}$$

Assim temos

$$A_q(n, d) \leq q^{n-d+1}. \tag{1.8}$$

Demonstração. Seja C um código $(n, M, d)_q$. Dados

$$c_1^i \cdots c_n^i,\ c_1^j \cdots c_n^j \in C, \quad c_1^i \cdots c_n^i \neq c_1^j \cdots c_n^j,$$

temos que $c_d^i \cdots c_n^i$ e $c_d^j \cdots c_n^j$ são palavras distintas, uma vez que caso contrário as palavras $c_1^i \cdots c_n^i$ e $c_1^j \cdots c_n^j$ só poderiam diferir em $d - 1$ posições e consequentemente o número $d(c_1^i \cdots c_n^i, c_1^j \cdots c_n^j)$ seria inferior a d. Seja C' o código

$$C' = \{c_d^1 \cdots c_n^1,\ \ldots,\ c_d^M \cdots c_n^M\} \subseteq (F_q)^{n-d+1}.$$

Como $(F_q)^{n-d+1}$ tem q^{n-d+1} palavras temos que $M \leq q^{n-d+1}$ e verifica-se (1.7). Deste modo, como $A_q(n, d)$ é o maior valor de M para um código q-ário com n palavras e distância mínima d, concluímos que $A_q(n, d) \leq q^{n-d+1}$ e temos (1.8). □

Pode verificar-se facilmente que se $q \geq 4$ o limite superior de Singleton é menor ou igual do que o limite superior de Hamming, sendo neste caso preferível considerar o limite superior de Singleton.

Exemplo 1.2.14. Consideremos o código do Exemplo 1.2.10 e calculemos o limite superior de Singleton. Temos

$$A_2(11,3) \leq 2^{11-3+1} = 512.$$

podemos concluir que, neste caso, é preferível considerar o limite superior de Hamming ($A_2(11,3) \leq 512/3 \cong 170,7$) calculado no Exemplo 1.2.10 do que o limite superior de Singleton. Note-se que, de acordo com a tabela da página 14 de [3], tem-se

$$144 \leq A_2(11,3) \leq 158.$$

Exemplo 1.2.15. Consideremos um código com parâmetros $(5, M, 3)_4$. Este código tem capacidade

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

e portanto o limite superior de Hamming para este código é

$$M \leq \frac{4^5}{(4-1)^0 + 5(4-1)} \Leftrightarrow M \leq 16.$$

Por outro lado, o limite superior de Singleton para este código é $M \leq 4^{5-3+1} = 64$. Concluímos que $A_4(5,3) \leq \min\{16, 64\} = 16$. Desta forma, o limite superior de Hamming é neste caso melhor do que o limite superior de Singleton.

Podemos ainda obter o limite inferior de Gilbert-Varshamov:

$$A_4(5,3) \geq \frac{4^5}{(4-1)^0 + 5(4-1) + 10(4-1)^2} > 9.$$

Concluímos que $10 \leq A_4(5,3) \leq 16$.

Capítulo 2

Códigos lineares

Neste capítulo analisamos uma família de códigos fundamental na teoria: a família dos códigos lineares. Concretamente, na Secção 2.1 discutimos a forma de introduzir uma estrutura de espaço linear em F_q^n, sendo q uma potência de um número primo, na Secção 2.2 abordamos o problema de codificar usando um código linear e na Secção 2.3 verificamos que os códigos lineares admitem um algoritmo de descodificação muito eficiente.

2.1 O conjunto F_q^n como espaço linear

Neste capítulo estamos interessados em considerar em F_q operações de soma e multiplicação por forma a introduzir em F_q uma estrutura de corpo (finito). De acordo com um teorema de Galois existe um corpo finito com q elementos se e só se q é uma potência de um número primo (isto é $q = p^r$ onde p é um número primo e $r \in \mathbb{N}$) e, a menos de isomorfismo, esse corpo é único (ver por exemplo [2]). Assim, neste capítulo consideraremos apenas alfabetos F_q em que q é uma potência de um número primo.

Dado um número primo p, temos, a menos de isomorfismo, um único corpo finito com p elementos, o qual é obtido considerando em F_p a soma módulo p e a multiplicação módulo p. O corpo assim obtido denota-se por $\mathbb{Z}_p$.

Podemos ainda facilmente ver que, considerando em F_q uma estrutura de corpo finito (naturalmente q é uma potência de um número primo) e considerando em F_q^n as operações de soma e de multiplicação por um escalar definidas, para cada $a_1 \cdots a_n, b_1 \cdots b_n \in F_q^n$ e $\alpha \in F_q$, respectivamente por

$$a_1 \cdots a_n + b_1 \cdots b_n = c_1 \cdots c_n,$$

onde $c_i = a_i + b_i$, $i = 1, \ldots, n$ ($+$ é a soma em F_q) e

$$\alpha \cdot a_1 \cdots a_n = d_1 \cdots d_n,$$

onde $d_i = \alpha \cdot a_i$, $i = 1, \ldots, n$ ($\cdot$ é a soma em F_q), o conjunto F_q^n fica com estrutura de espaço linear sobre F_q. Se C é um subespaço linear de F_q^n de dimensão k então C diz-se um código-$[n, k]$ linear (sobre F_q).

Um exemplo importante é o caso em que p é um número primo e portanto fazemos $F_p = \mathbb{Z}_p$. Neste caso consideramos em $(\mathbb{Z}_p)^n$ as operações de soma e de multiplicação por um escalar definidas respectivamente por

$$a_1 \cdots a_n + b_1 \cdots b_n = c_1 \cdots c_n,$$

onde $c_i = a_i + b_i \mod q$, $i = 1, \ldots, n$ e

$$\alpha \cdot a_1 \cdots a_n = d_1 \cdots d_n,$$

onde $d_i = \alpha \cdot a_i \mod q$, $i = 1, \ldots, n$, para cada $a_1 \cdots a_n, b_1 \cdots b_n \in \mathbb{Z}_p^n$ e $\alpha \in \mathbb{Z}_p$. Ficamos assim com uma estrutura de espaço linear em $(\mathbb{Z}_p)^n$. Muitos dos exemplos de códigos considerados neste trabalho são subespaços lineares de $(\mathbb{Z}_p)^n$.

Considerarmos códigos lineares é vantajoso logo à partida porque a estrutura de espaço vectorial permite-nos obter todas as palavras de código a partir de um conjunto pequeno de palavras de código. Já para especificar um código não-linear é muitas vezes necessário listar todas as palavras código. Além disso, existem procedimentos específicos para codificar e descodificar no contexto dos códigos lineares. Por outro lado, para considerarmos códigos lineares, temos de nos restringir aos alfabetos F_q com q potência de um número primo, o que à partida restringe as nossas possibilidades. Ainda assim, códigos em que q não é uma potência de um número primo podem ser obtidos muitas vezes a partir de códigos lineares considerando um alfabeto maior.

Designemos por *peso da palavra* x, e representemos por $\omega(x)$, o número de coordenadas de x não nulas, ou seja, $\omega(x) = d(x, 0)$ em que 0 é o vector nulo de F_q^n. Já antes considerámos o conceito de peso e uma relação entre peso e distância no caso particular dos códigos binários. O lema seguinte pode assim ser visto como uma generalização de (1.1).

Lema 2.1.1. Se $x, y \in F_q^n$ com q potência de um número primo então $d(x, y) = \omega(x - y)$.

Demonstração. Como $x - y$ tem entradas não nulas precisamente nas posições em que x

e y diferem, temos

$$\begin{aligned} d(x,y) =& \sharp\{i \in \{1,\ldots,n\} : x_i \neq y_i\} \\ =& \sharp\{i \in \{1,\ldots,n\} : x_i - y_i \neq 0\} \\ =& \omega(x-y) \end{aligned}$$

e portanto $d(x,y) = \omega(x-y)$. □

Exemplo 2.1.2. Sejam $x = 001$ e $y = 011$ duas palavras de $(\mathbb{Z}_2)^3$. Temos então

$$d(001, 011) = \omega(001 - 011) = \omega(010) = 1.$$

Definimos o *peso do código linear* C e denotamos por $\omega(C)$ o menor dos pesos das palavras do código não nulas.

Teorema 2.1.3. A distância mínima de um código linear C é $\omega(C)$.

Demonstração. Seja $D(C)$ a distância mínima do código C. Dados $x, y \in C$ com $x \neq y$ sabemos que $x - y \in C$ (porque C é um subespaço vectorial). Assim, de acordo com o Lema 2.1.1, podemos afirmar que $d(x,y) = \omega(x-y) \geq \omega(C)$, pois $\omega(C)$ é o menor dos pesos. Logo $d(C) \geq \omega(C)$

Reciprocamente, como $\omega(C)$ é o menor peso das palavras de código não nulas e $0 \in C$ (porque C é um subespaço vectorial), existe $x \in C$ tal que $\omega(C) = \omega(x) = \omega(x-0) = d(x,0) \geq d(C)$. Podemos então concluir que $d(C) = \omega(C)$. □

Nos códigos lineares, não vamos considerar exactamente o mesmo conceito de equivalência de códigos que considerámos no Capítulo 1.

Diremos que dois códigos lineares são *equivalentes* se um se puder obter a partir de outro através de uma combinação das seguintes operações:

i) Permutação das posições dos elementos do código;

ii) Multiplicação de símbolos aparecendo numa posição fixa por um escalar não nulo.

Dado um código linear C, designamos por *base do código* C uma base de C enquanto subespaço vectorial de F_q^n. Uma matriz $k \times n$ cujas linhas formam uma base do código-$[n,k]$ linear C diz-se uma *matriz geradora* de C.

Teorema 2.1.4. Seja G a matriz geradora de um código- $[n,k]$, então, efectuando algumas das operações seguintes:

a) Permutação de duas linhas;

b) Multiplicação de uma linha por um escalar não nulo;

c) Soma de duas linhas;

d) Permutação de duas colunas;

e) Multiplicação de uma coluna por um escalar não nulo.

transformamos G numa matriz geradora de um código equivalente. Em particular, podemos sempre transformar a matriz G numa matriz da forma $[I_k \mid A]$, geradora de um código equivalente.

Demonstração. É imediato verificar que as operações a), b), c), d) e e) transformam o código noutro equivalente. Seja G a matriz geradora de um código-$[n, k]$ e denotemos por g_{ij} o elemento da linha i e da coluna j e designemos ainda respectivamente por $l_1, \cdots, l_k$ e $c_1, \cdots, c_n$ as linhas e colunas da matriz G. Para cada $j = 1, \cdots, k$ consideramos a coluna c_j que contém os elementos $g_{1j}, g_{2j}, \ldots, g_{j-1,j}, \ g_{jj}, \ldots, g_{kj}$. Aplicamos primeiro sucessivamente os seguintes passos às colunas $c_1, \ldots, c_k$:

1. a) Se $g_{jj} \neq 0$, passamos para o Passo 2.
   b) Se $g_{jj} = 0$ e se existe $i > j$ tal que $g_{ij} \neq 0$ então comutam-se as linhas l_j e l_i.
   c) Se $g_{jj} = 0$ e $g_{ij} = 0$ para todo o $i > j$ então escolhe-se h tal que $g_{jh} \neq 0$ e comutam-se as colunas c_j e c_h.
2. Tem-se $g_{jj} \neq 0$ e multiplica-se a linha l_j por g_{jj}^{-1}.
3. Tem-se $g_{jj} = 1$. Para cada $i = 1, \ldots, k$ com $i \neq j$ substitui-se a linha l_i por $l_i - g_{ij} l_j$.

Depois de percorrer estas etapas conseguimos obter a partir da matriz geradora G uma matriz da forma $[I_k \mid A]$ geradora de um código equivalente. □

Exemplo 2.1.5. Seja C o código-$[6, 4]$ definido sobre F_2 com a seguinte matriz geradora

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

Condensando a matriz G podemos transformá-la numa matriz da forma $[I_4 \mid A]$.

De facto,

$$G = \begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 & 0 & 1 \end{bmatrix} \xrightarrow[L_3' \to L_1 + L_3]{} \begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 \end{bmatrix}$$

$$\xrightarrow[L_4' \to L_3' + L_4]{} \begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix} \xrightarrow[L_3'' \to L_3' + L_2]{} \begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}$$

$$\xrightarrow[L_1' \to L_1 + L_2]{} \begin{bmatrix} 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix} \xrightarrow[L_1'' \to L_1' + L_4']{} \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}$$

$$\xrightarrow[C_3 \leftrightarrow C_4]{} \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 \end{bmatrix} = [I_4|A].$$

Obtivemos assim uma matriz que gera um código equivalente.

2.2 Codificar com um código linear

Seja C um código-$[n, k]$ linear sobre F_q e G uma matriz geradora. O código C permite codificar q^k palavras distintas. Podemos então identificar cada uma dessas palavras com um vector de $\{1, \ldots, q\}^k$. Claro que as palavras de C tem tamanho n e portanto existe nesta codificação uma certa redundância. A redundância referida faz com que a distância mínima seja maior do que 1, permitindo-nos muitas vezes, como já sabemos, detectar e corrigir erros.

Vamos ver como é que, escolhida uma palavra a transmitir, que identificamos com um vector $\mu = (\mu_1, \ldots, \mu_k)$ podemos codificá-la, obtendo uma palavra do código C. Como referimos, identificamos cada palavra a transmitir com o vector $\mu = (\mu_1, \ldots, \mu_k)$. Para

codificar esta palavra multiplicamos o vector linha $[\mu_1 \cdots \mu_k]$ por G:

$$\mu G = [\mu_1 \cdots \mu_n] \begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{k1} & \cdots & a_{kn} \end{bmatrix} = \left[\sum_{i=1}^{k} \mu_i a_{i1} \ \cdots \ \sum_{i=1}^{k} \mu_i a_{in} \right]$$

Vejamos dois exemplos.

Exemplo 2.2.1. Consideremos a matriz $[I_4 \mid A]$ do exemplo 2.1.5. Esta matriz gera um código-$[6, 4]$ binário linear. Assim podemos codificar 2^4 palavras que identificamos com os vectores de $\{0, 1\}^4$. De acordo com o exposto acima, dado um vector $(\mu_1, \mu_2, \mu_3, \mu_4)$ associamos-lhe a palavra de código:

$$\begin{aligned} \mu G &= \begin{bmatrix} \mu_1 & \mu_2 & \mu_3 & \mu_4 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 \end{bmatrix} \\ &= \begin{bmatrix} \mu_1 & \mu_2 & \mu_3 & \mu_4 & \mu_1 + \mu_2 + \mu_4 & \mu_1 + \mu_2 + \mu_3 + \mu_4 \end{bmatrix} \end{aligned}$$

Assim, por exemplo, o vector $(1, 0, 1, 0)$ é codificado como 101010 e o vector $(0, 1, 0, 0)$ como 010011.

Exemplo 2.2.2. Consideremos a matriz geradora

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

Esta matriz gera um código-$[4, 2]$ linear. De facto, ao vector (μ_1, μ_2) associamos a palavra de código dada por

$$[\mu_1 \ \mu_2] \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix} = [\mu_1 \ \mu_2 \ \mu_2 \ \mu_1] .$$

Assim os vectores $(0, 0)$, $(0, 1)$, $(1, 0)$ e $(1, 1)$ são codificados respectivamente como 0000, 0110, 1001 e 1111. Obtemos portanto o código

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

2.3 Descodificar com um código linear

Suponhamos que queremos enviar a palavra código $x_1 x_2 \cdots x_n$ e é recebida a palavra de código $y_1 y_2 \cdots y_n$. Definimos o vector erro e por $e = y - x = e_1 e_2 \cdots e_n$.

Dado C, um código-$[n,k]$, e $a \in F_q^n$ podemos definir o conjunto

$$a + C = \{a + x \colon x \in C\}$$

que designamos por *classe de equivalência (de a).*

Lema 2.3.1. Seja $a + C$ uma classe de equivalência e $b \in a + C$. Temos que

$$b + C = a + C.$$

Demonstração. Suponhamos que $b \in a + C$. Então existe $x \in C$ tal que $b = a + x$. Assim, dado $y \in C$, temos que $x + y \in C$ uma vez que C é linear e

$$b + y = (a + x) + y = a + (x + y) \in a + C \Rightarrow b + C \subseteq a + C$$

Por outro lado, se $a + z \in a + C$ então $z - x \in C$ uma vez que C é linear e

$$a + z = (b - x) + z = b + (z - x) \in b + C$$

pelo que $a + C \subseteq b + C$.

Podemos então concluir que $b + C = a + C$. □

O próximo teorema mostra que um código linear pode ser decomposto numa reunião de classes de equivalência disjuntas com q^k palavras cada uma.

Teorema 2.3.2. Seja C um código-$[n,k]$, então:

i) Todas as palavras do código estão em alguma classe de equivalência de C;

ii) Todas as classes de equivalência contêm exactamente q^k palavras;

iii) Quaisquer duas classes de equivalência distintas são disjuntas.

Demonstração. Se $a \in F_q^n$ então $a = a + 0 \in a + C$, o que mostra i).

Cada classe de equivalência é obtida somando uma palavra a cada palavra do código. Portanto, como o código tem k palavras, cada classe de equivalência tem k palavras. Assim, existem tantas classes de equivalência como palavras do código e portanto todas as classes de equivalência têm q^k palavras. Fica estabelecido ii).

Sejam $a + C$ e $b + C$ duas classes de equivalência de C que não são disjuntas nem coincidentes. Portanto, existe uma palavra $d \in (a + C) \cap (b + C)$, ou seja, para alguns $x, y \in C$ tem-se $d = a + x = b + y$. Ora, podemos concluir que $a = b + (y - x) \in b + C$. Pelo Lema 2.3.1 sabemos que se $a \in b + C$ então $a + C = b + C$, o que contradiz a hipótese. Concluímos que $a + C$ e $b + C$ ou são disjuntas ou são coincidentes e demonstramos iii). □

Designamos um vector com peso mínimo dentro de uma classe de equivalência por *líder da classe de equivalência*. Podemos portanto escrever

$$F_q^n = (\ell_1 + C) \cup \cdots \cup (\ell_{q^{n-k}} + C),$$

onde as classes de equivalência são disjuntas duas a duas e $\ell_1, \cdots, \ell_{q^{n-k}}$ são líderes das classes de equivalência.

Abordaremos o problema de descodificar com um código linear com um exemplo. Esta opção tem a vantagem de não exigir notação demasiado pesada, ficando suficientemente claro como abordar o caso geral. Consideremos o código linear do Exemplo 2.2.2:

$$C = \{0000, 0110, 1001, 1111\} \subset F_2^4.$$

e a matriz geradora associada

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

Temos para C as seguintes classes de equivalência:

$$\begin{aligned} 0000 + C &= \{0000, 0110, 1001, 1111\} = C \\ 1000 + C &= \{1000, 1110, 0001, 0111\} \\ 0100 + C &= \{0100, 0010, 1101, 1011\} \\ 1010 + C &= \{1010, 1100, 0011, 0101\}. \end{aligned}$$

Escolhemos um líder em cada classe de equivalência. Podemos, por exemplo, designar 1000 como o líder da classe de equivalência $1000 + C$. Como vimos, de acordo com o Teorema 2.3.2, podemos escrever

$$F_2^4 = (0000 + C) \cup (1000 + C) \cup (0100 + C) \cup (1010 + C),$$

sendo as classes de equivalência disjuntas duas a duas.

O objectivo seguinte é construir uma *matriz de Slepian* para C, a qual nos permitirá descodificar com o código linear. Ora, uma matriz de Slepian de um código-$[n, k]$ linear é uma matriz $q^{n-k} \times q^k$ onde as classes de equivalência são escritas linha a linha, começando sempre pela classe de equivalência que contém a palavra $00 \cdots 0$, sendo a primeira coluna constituída pelos líderes das classes de equivalência do código, e sendo cada classe de equivalência disposta pela ordem que se obtém somando ao respectivo líder (colocado na

primeira coluna) os elementos da classe de equivalência de $00\cdots0$ pela ordem que foram colocados na matriz. A seguinte matriz

$$\begin{bmatrix} 0000 & 0110 & 1001 & 1111 \\ 1000 & 1110 & 0001 & 0111 \\ 0100 & 0010 & 1101 & 1011 \\ 1010 & 1100 & 0011 & 0101 \end{bmatrix}$$

é uma matriz de Slepian do código C. Vamos usá-la para descodificar uma palavra recebida. Suponhamos que é enviada a palavra x e é recebida a palavra $y = 1101$. Esta palavra não pertence ao código, detectamos portanto que ocorreram erros.

Para descodificar temos de seguir os seguintes passos:

1. descobre-se a posição de y na *matriz de Slepian*:

2. o erro $e = y - x$ ocorrido é exactamente igual ao líder dessa classe de equivalência (atendendo à forma como foi construída a matriz);

3. descoberto o erro resta fazer $x = y - e$.

No caso concreto que estávamos a considerar, a palavra recebida $y = 1101$ está na terceira linha e terceira coluna. O líder dessa classe de equivalência é a palavra 0100 que consideramos ser o erro que ocorreu. Assim, a palavra é descodificada como $x = y - e = 1101 - 0100 = 1001$. Note-se que a palavra 1101 não está no código mas está à distância 1 de duas palavras de código (a palavra 1001 e a palavra 1111). O algoritmo de descodificação permite-nos decidir pela palavra 1001, no entanto, tal escolha não pode ser feita atendendo à distância de Hamming entre a palavra recebida e as palavras de código. Note-se que este código tem distância mínima 2 e portanto, de acordo com o Teorema 1.1.5 permite detectar 1 erro de substituição mas não permite corrigir erros.

Capítulo 3

Códigos perfeitos

Este capítulo pretende, sem detalhar todos os aspectos, visitar alguns resultados sobre códigos perfeitos. Nomeadamente: na Secção 3.1 apresentamos o conceito de código dual que se revela um instrumento fundamental para definir alguns dos mais importantes códigos perfeitos; na Secção 3.2 mostramos que os códigos de repetição binários de tamanho ímpar e o conjunto F_q^n são exemplos elementares de códigos perfeitos; nas Secções 3.3 e 3.4 introduzimos respectivamente os códigos de Hamming binários e os códigos de Golay que se contam entre os códigos perfeitos mais famosos; finalmente na Secção 3.5 enunciamos um resultado que caracteriza os códigos perfeitos sobre alfabetos q-ários onde q é uma potência de um número primo.

3.1 O código dual

Seja q uma potência de um número primo. Dados elementos de F_q^n, $x = (x_1, \ldots, x_n)$ e $y = (y_1, \ldots, y_n)$, é fácil verificar que a aplicação $\langle \cdot, \cdot \rangle \colon F_q^n \times F_q^n \to F_q$ dada por

$$\langle x, y \rangle = x_1 y_1 + \cdots + x_n y_n \mod q$$

é um produto interno em F_q^n, isto é, para todos os $x, y, z \in F_q^n$ e $\alpha \in F_q$ temos

i) $\langle x + y, z \rangle = \langle x, z \rangle + \langle y, z \rangle$;

ii) $\langle \alpha x, y \rangle = \alpha \langle x, y \rangle$;

iii) $\langle x, y \rangle = \langle y, x \rangle$;

iv) $\langle x, x \rangle \geq 0$.

O conceito de produto interno será aqui usado para introduzir o conceito de código dual de um código linear. Naturalmente diremos que duas palavras $x, y \in F_q^n$ são *ortogonais* se $\langle x, y\rangle = 0$.

Seja C um código-$[n, k]$ linear. Define-se o *código dual de* C, que se denota por $C^\perp$, por

$$C^\perp = \{x \in F_q^n \colon \langle x, y\rangle = 0,\ y \in C\}.$$

Assim o código dual de um código linear é o código formado por todas as palavras de F_q^n que são ortogonais a C.

O seguinte resultado caracteriza o código dual de um código linear.

Teorema 3.1.1. Seja q uma potência de um número primo e C um determinado código-$[n, k]$ linear sobre F_q com uma matriz geradora G. Então

a) para cada $x \in F_q^n$ temos

$$x \in C^\perp \ \Leftrightarrow\ x\,G^T = 0,$$

onde G^T designa a matriz transposta de G.

b) o código $C^\perp$ é um código-$[n, n-k]$ linear sobre F_q.

c) temos $(C^\perp)^\perp = C$.

Demonstração. Sejam $l_1, \ldots l_n$ as linhas de G e $x = x_1 \ldots x_n \in F_q^n$ uma palavra tal que $x\,G^T = 0$ ou seja tal que $\langle x, l_i\rangle = 0$, $i = 1, \ldots, n$. Dada uma palavra $c = c_1 \ldots c_n \in C$ existem constantes $\lambda_1, \ldots, \lambda_n$ tais que $c = \sum_{i=1}^n \lambda_i l_i$ uma vez que C é um código linear. Deste modo

$$\langle x, c\rangle = \sum_{i=1}^n \lambda_i \langle x, l_i\rangle = \sum_{i=1}^n \lambda_i 0 = 0.$$

Assim x é ortogonal a todas as palavras de C e portanto temos que $x \in C^\perp$.

Por outro lado, se $x \in C^\perp$, temos que x é ortogonal às linhas da matriz G uma vez que estas linhas são palavras do código C. Fica demonstrado a).

Para demonstrar b), comecemos por provar que $C^\perp$ é um código linear. Para tal vamos mostrar que $C^\perp$ é um subespaço linear de F_q. Dados $x, y \in C^\perp$ e $\lambda \in F_q$ tem-se, para todo o $c \in C$,

$$\langle x + \lambda y, c\rangle = \langle x, c\rangle + \lambda\langle y, c\rangle = 0 + \lambda 0 = 0,$$

pelo que $x + \lambda y \in C^\perp$. Conclui-se que $C^\perp$ é subespaço linear de F_q e consequentemente que $C^\perp$ é um código linear.

Vamos agora mostrar que $C^\perp$ é um código com dimensão $n-k$. Seja G a matriz geradora do código C. Ora, atendendo a que G tem então dimensão k e à alínea a), temos que

$$x = (x_1, \ldots, x_n) \in C^\perp \;\Leftrightarrow\; x\,G^T = 0,$$

ou seja,

$$\sum_{j=1}^{n} g_{ij}x_j = 0, \quad i = 1, \ldots, k,$$

onde g_{ij}, $i \in \{1, \ldots, k\}$, $j \in \{1, \ldots, n\}$, são as entradas da matriz G. Obtemos portanto um sistema de k equações linearmente independentes com n incógnitas. Naturalmente, o código $C^\perp$ que é o conjunto das soluções deste sistema (que corresponde ao núcleo da matriz $G^\perp$) é um subespaço linear de dimensão $n-k$.

Demonstramos agora a alínea c). Como todas as palavras de C são ortogonais a todas as palavras de $C^\perp$, concluímos que C está contido em $(C^\perp)^\perp$. Por outro lado, pela alínea b), temos

$$\dim(C^\perp)^\perp = n - (n-k) = k = \dim C.$$

Conclui-se portanto que $(C^\perp)^\perp = C$. □

Seja C um código-$[n,k]$ linear. Uma matriz geradora do código $C^\perp$ designa-se por *matriz de paridade* do código C.

Exemplo 3.1.2. A matriz paridade do código de repetição $R_{3,2}$ é a matriz H geradora do código $R_{3,2}^\perp$. Como $R_{3,2} = \{000, 111\}$, temos

$$x_1x_2x_3 \in R_{3,2}^\perp \quad \Leftrightarrow \quad \langle x_1x_2x_3, 111\rangle = x_1 + x_2 + x_3 = 0.$$

Assim

$$\begin{aligned} R_{3,2}^\perp &= \{x_1x_2x_3 \in (F_2)^3 : \langle x_1x_2x_3, 111\rangle = 0\} \\ &= \{x_1x_2x_3 \in (F_2)^3 : x_1 + x_2 + x_3 = 0\}. \end{aligned}$$

Como $R_{3,2}^\perp$ tem dimensão 2 e as palavras 101, 011 $\in R_{3,2}^\perp$ são linearmente independentes, uma matriz geradora para $R_{3,2}^\perp$ é uma matriz cujas linhas sejam estas palavras de código:

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

3.2 Códigos perfeitos triviais

Designamos por *código perfeito trivial* qualquer um dos seguintes códigos:

- os códigos de repetição binários de tamanho ímpar;
- os códigos consistindo em todo o F_q^n.

Temos o seguinte resultado:

Teorema 3.2.1. Os códigos perfeitos triviais são códigos perfeitos.

Demonstração. Os códigos de repetição binários de tamanho ímpar só possuem duas palavras. De acordo com o teorema binomial, temos

$$2^n = (1+1)^n = \binom{n}{0} + \binom{n}{1} + \cdots + \binom{n}{n}.$$

Portanto, atendendo a que para n ímpar e cada $i = 1, \ldots, (n+1)/2$ temos um número par de termos ($n+1$ termos) e notando que

$$\binom{n}{i} = \binom{n}{n-i},$$

conclui-se que

$$2\left(\binom{n}{0} + \cdots + \binom{n}{(n+1)/2}\right) = 2^n.$$

Assim, os códigos de repetição binários de tamanho ímpar são códigos perfeitos.

Os códigos formados por todo o F_q^n tem q^n palavras de tamanho n à distância mínima de 1. Deste modo

$$q^n \binom{n}{0} = q^n \cdot 1 = q^n,$$

pelo que também estes códigos são perfeitos. □

3.3 Códigos de Hamming binários

Um *código de Hamming binário* é um código linear binário cuja matriz de paridade é uma matriz $k \times (2^k - 1)$ cujas colunas são todos os vectores não nulos de $(F_2)^k$. Denota-se um tal código por $\mathrm{Ham}(k, 2)$.

Exemplo 3.3.1. O código linear com matriz de paridade

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

é um código $\mathrm{Ham}(2, 2)$. De acordo com o Exemplo 3.1.2, este é o código de repetição $\mathcal{R}_{3,2}$.

Exemplo 3.3.2. O código de Ham$(3,2)$ tem como matriz de paridade uma matriz cujas colunas sejam todos os vectores não nulos de F_2^3, por exemplo a matriz:

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

O código Ham$(4,2)$ tem como matriz de paridade uma matriz cujas colunas sejam todos os vectores não nulos de F_2^4, por exemplo a matriz:

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

É imediato que existem $P_7 = 7! = 5040$ matrizes de paridade distintas para o código Ham$(3,2)$ e $P_{15} = 15! = 1307674368000$ matrizes de paridade distintas para Ham$(4,2)$.

Teorema 3.3.3. Dado $k \geq 2$, o código Ham$(k,2)$ é um código-$[2^k - 1, 2^k - 1 - k]$ linear com distância mínima 3.

Demonstração. Por definição, um código de Hamming binário Ham$(k,2)$ é um código linear binário cuja matriz de paridade, H, é constituída por todos os vectores não nulos de $(F_2)^k$. Naturalmente, H tem dimensões $k \times (2^k - 1)$. Como uma matriz de paridade de Ham$(k,2)$ é uma matriz geradora de Ham$(k,2)^\perp$ e portanto tem dimensões $k \times (2^k - 1)$, concluímos que o código Ham$(k,2)^\perp$ é um código linear de dimensão k com palavras de tamanho $2^k - 1$. Isto é, Ham$(k,2)^\perp$ é um código linear-$[2^k - 1, k]$. Deste modo, de acordo com as alíneas b) e c) do Teorema 3.1.1, o código Ham$(k,2) = (\text{Ham}(k,2)^\perp)^\perp$ é um código $[2^k - 1, 2^k - 1 - k]$.

Para provar que o código Ham$(k,2)$ é um código com distância mínima 3 temos de mostrar que todas as palavras não nulas pertencentes ao código têm peso maior ou igual a 3. Para tal, provemos que, à excepção da palavra nula, o código Ham$(k,2)$ não tem palavras com peso 1 ou 2. Suponhamos por contradição que Ham$(k,2)$ tem uma palavra x com peso 1, onde o número 1 aparece na i-ésima posição (e naturalmente em todas as outras posições temos 0). Como x é ortogonal a todas as linhas da matriz H, a i-ésima entrada de todas as linhas da matriz H é o vector 0. Mas isto contradiz a definição de matriz H (uma vez que as colunas desta são vectores não nulos). Só falta provar que o código não tem palavras com peso 2. Suponhamos por contradição que y é uma palavra código com

peso 2. Sejam i e j as posições de y ocupadas pelo número 1 (sendo naturalmente todas as outras posições ocupadas por 0). Como y é ortogonal a todas as linhas da matriz H, as entradas i e j de cada linha têm de ser ambas 1 ou ambas 0. Assim, as colunas i e j são iguais o que contradiz, novamente, a definição de matriz de paridade (cada vector não nulo só pode surgir uma vez nas colunas da matriz de paridade). Deste modo, o código $\mathrm{Ham}(k,2)$ tem distância mínima 3. □

Exemplo 3.3.4. De acordo com o teorema anterior, como $2^3-1=7$ e $2^3-1-3=4$, um código $\mathrm{Ham}(3,2)$ é um código-$[7,4]$ linear, binário com distância mínima 3, ou seja, um código $(7,16,3)_2$.

Os códigos de Hamming binários dos Exemplos 3.3.1 e 3.3.2 são códigos perfeitos. O seguinte corolário do teorema 3.3.3 mostra que todos os códigos de Hamming binários são perfeitos.

Teorema 3.3.5. Dado $k \geq 2$, o código $\mathrm{Ham}(k,2)$ é um código perfeito corrector de um erro.

Demonstração. Seja $k \geq 2$. Para mostrar que $\mathrm{Ham}(k,2)$ é um código perfeito, basta garantir que o Limite superior de Hamming é atingido. De facto temos

$$1+\binom{2^k-1}{1}=1+\frac{(2^k-1)!}{1!(2^k-2)!}=1+2^k-1=2^k,$$

o que mostra que $\mathrm{Ham}(k,2)$ é um código perfeito.

Como $\mathrm{Ham}(k,2)$ tem distância mínima 3, concluímos, de acordo com a alínea ii) Teorema 1.1.5, que corrige um erro. □

3.4 Códigos de Golay

Nesta secção pretendermos, sem entrar com profundidade nos pormenores, fazer referência a dois códigos perfeitos muito importantes na teoria: os códigos binário e ternário de Golay. O leitor que pretenda aprofundar a teoria relativa a estes códigos pode consultar por exemplo os livros [3, 4].

3.4.1 Código de Golay binário

O *código de Golay binário*, que denotamos por $\mathcal{G}_{23}$, é o código linear com matriz geradora

$$G_{bin}=[I_{12}|A_{bin}]$$

onde I_{12} é a matriz identidade 12×12 e

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

Atendendo a que a matriz A_{bin} é uma matriz 12×11 e à forma da matriz G_{bin} concluímos que as palavras de $\mathcal{G}_{23}$ têm tamanho 23. Por outro lado, pode mostrar-se que $\mathcal{G}_{23}$ é um código com distância mínima 7 que possui 2^{12} palavras. Assim $\mathcal{G}_{23}$ é um código $(23, 2^{12}, 7)_2$. Como

$$2^{12}\left[\binom{23}{0} + \binom{23}{1} + \binom{23}{2} + \binom{23}{3}\right] = 2^{12}\left(1 + 23 + 253 + 1771\right) = 2^{12}2^{11} = 2^{23}$$

concluímos que $\mathcal{G}_{23}$ é um código perfeito.

3.4.2 Código de Golay ternário

O código de Golay ternário, que denotamos por $\mathcal{G}_{11}$, é o código linear com matriz geradora

$$G_{ter} = [I_6 | A_{ter}]$$

onde I_6 é a matriz identidade 6×6 e

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

Atendendo a que a matriz A_{ter} é uma matriz 6×5 e à forma da matriz G_{ter} concluímos que as palavras de $\mathcal{G}_{11}$ têm tamanho 11. Por outro lado, pode mostrar-se que $\mathcal{G}_{11}$ é um código com distância mínima 5 que possui 3^6 palavras. Assim $\mathcal{G}_{11}$ é um código $(11, 3^6, 5)_3$. Como

$$3^6 \left[\binom{11}{0} + \binom{11}{1} 2 + \binom{11}{2} 2^2 \right] = 3^6 \left(1 + 22 + 220\right) = 3^6\, 3^5 = 3^{11}$$

concluímos que $\mathcal{G}_{11}$ é um código perfeito.

3.5 Outros códigos perfeitos

Os códigos de Hamming possuem uma generalização imediata para alfabetos q-ários com q potência de um número primo. Daremos aqui uma breve definição destes códigos. Dado $x \in F_q^n \setminus \{0\}$, consideramos o conjunto

$$C_x = \{px \colon p \in F_q \setminus \{0\}\}.$$

Atendendo a que, como facilmente se verifica, conjuntos C_x e C_y distintos são disjuntos, a que $x \in F_q^n \setminus \{0\}$ tem $q^n - 1$ elementos e a que $F_q \setminus \{0\}$ tem $q - 1$ elementos, F_q^n pode ser visto como reunião disjunta de $\lambda = \frac{q^n-1}{q-1}$ conjuntos:

$$F_q^n \setminus \{0\} = C_{x_1} \cup \cdots \cup C_{x_\lambda}.$$

Escolhendo para cada $i = 1, \ldots, \lambda$ um elemento $x_i \in C_i$, obtemos um conjunto de $\frac{q^n-1}{q-1}$ elementos, que se pode verificar serem linearmente independentes dois a dois. Pode demonstrar-se que a matriz cujas colunas são os vectores $x_1, \ldots, x_n$ é uma matriz de paridade de um código linear-$[\frac{q^n-1}{q-1}, \frac{q^n-1}{q-1} - n]$ com distância mínima 3. O código obtido designa-se por *código de Hamming q-ário* e denota-se por $\mathrm{Ham}(n, q)$. Pode facilmente verificar-se que qualquer um destes códigos é perfeito e que os códigos de Hamming 2-ários coincidem com os códigos de Hamming binários definidos na Secção 3.3. Designamos qualquer um destes códigos por *código de Hamming*. Para mais informação sobre códigos de Hamming q-ários remetemos o leitor para [3, 4].

O seguinte resultado, com origem em trabalhos de Van Lint [8] e Tietäväinen [12] restringe consideravelmente as possibilidades para os parâmetros de um código perfeito.

Teorema 3.5.1 (Van Lint e Tietäväinen)**.** Seja q uma potência de um número primo. Então qualquer código q-ário perfeito é um código trivial ou têm os mesmos parâmetros de um dos códigos de Hamming, do código binário de Golay ou do código ternário de Golay.

Demonstração. Ver por exemplo [9]. □

Este teorema diz em particular que, ainda que existam códigos não-triviais perfeitos, com q potência de um primo, que não são de um dos tipos referidos anteriormente (e de facto existem), existe sempre um código dos que vimos anteriormente que permite detectar e corrigir o mesmo número de erros.

Capítulo 4

Códigos no Ensino Secundário

O ensino da matemática é objecto de estudo de alguns investigadores há já algum tempo. O principal objectivo deste estudo é tornar a matemática aliciante começando por entusiasmar os alunos desde cedo. Para tal, tem-se apostado, principalmente, em métodos de ensino que tenham como base programas interactivos e didácticos. É claro que, na prática, aplicando estes métodos de ensino corre-se o risco de simplificar demasiado, tornando a matemática mecanicista o que pode implicar que a aprendizagem se desenvolva por transmissão e absorção e não por construção.

Nos dias de hoje, deparamo-nos com uma comunidade moderna e cada vez mais exigente, na qual o professor deverá adquirir um papel activo, em constante mudança (actualizações), ao longo de todo o processo de ensino-aprendizagem. No decorrer deste processo complexo, compete ao docente proporcionar aulas que estimulem os alunos da disciplina em questão, permitindo um grande envolvimento por parte dos mesmos. Cabe-lhe diversificar as estratégias de abordagem dos conteúdos matemáticos através de uma linguagem simples e harmonizada. O professor de matemática deve proporcionar aos alunos novas perspectivas desta mesma disciplina, dando a conhecer a possibilidade de realização de actividades agradáveis e úteis, permitindo que os alunos percebam a aplicabilidade do que estão a aprender. Segundo Nápoles, Santos e Veloso [10], o professor deverá saber explicar com clareza todos os passos e etapas de um determinado exercício, mas também deverá conhecer as razões, isto é, saber explicar porque se faz de determinada maneira e relacionar com ideias particulares. Na minha opinião, actualmente, principalmente no ensino secundário, os professores preocupam-se mais em preparar os alunos apenas para os exames, esquecendo-se do seu verdadeiro papel. Penso que, muito por causa dos exames, os jovens neste nível de ensino procuram exercícios "tipo", decorando-os. Como tal e para

contrariar este tipo de atitude, o principal objectivo de um professor de Matemática deveria passar por conseguir que os alunos sejam capazes de construir raciocínios lógicos e coerentes, entusiasmando-os. Na minha opinião, os professores deveriam, de vez em quando, abordar temas em "aulas livres", pertencentes ou não ao currículo da Matemática, a fim de, sem a pressão de avaliações, os alunos conseguirem sentir a beleza da Matemática. Com o desenvolvimento deste trabalho senti que seria bastante interessante propor uma "aula livre"inserida no contexto da Teoria de Códigos. Depois de alguma reflexão decidi que seria melhor propor essa aula ao nível do ensino secundário para haver mais facilidade na compreensão dos conceitos utilizados. Seria interessante até considerar a possibilidade de inserção da Teoria de Códigos no currículo do ensino secundário, no 12º ano de escolaridade, logo após o capítulo Probabilidades e Combinatória. Trata-se de um tema presente em várias situações do dia a dia (tais como a transmissão de imagens via satélite, o processamento de imagens digitais e as telecomunicações), o que podia servir motivação adicional para o estudo da matemática relacionada com o tema em questão.

Começamos por introduzir, nas Secções 4.1 e 4.2, os códigos a abordar e seguidamente, na Secção 4.3, propomos uma aula sobre estes códigos.

4.1 O código ISBN

Criado no Reino Unido em 1967 pela livraria W. H. Smith, o International Standard Book Number (ISBN) é um sistema universalmente usado para identificar livros. Até 1 de Janeiro de 2007, este número é foi constituido por 10 dígitos divididos em quatro elementos. Desde essa data, o ISBN passou a ter 13 dígitos com fim a aumentar a capacidade de numeração do sistema em vigor até então. O ISBN costuma estar na contracapa dos livros junto ao código de barras.

Figura 4.1: ISBN

4.1.1 ISBN com 10 dígitos

Os dígitos do ISBN com 10 dígitos têm o seguinte significado:

- 1º dígito - Língua;
- 2º a 4ª dígitos - Número Internacional da Editora;
- 5º ao 9º dígitos - Número que a editora atribui ao livro;
- 10º dígito - Dígito de Controlo.

Por exemplo, o ISBN do livro *Ciência* de João Caraça é

$$9 - 727 - 09287 - X$$

onde os dígitos têm o seguinte significado

- 9 - Língua Portuguesa;
- 727 - Editora Difusão Cultural;
- 09287 - Número atribuído pela editora;
- X - Dígito de Controle.

Uma forma de determinar um erro numa palavra de código consiste em considerar um dígito adicional, a_{n+1}, chamado *dígito de controlo*, que conserve alguma informação sobre os dígitos anteriores.

O dígito de controlo d do ISBN de 10 dígitos é calculado do seguinte modo:

$$d = \sum_{i=1}^{9} i a_i \mod 11.$$

Assim d assume valores inteiros entre 0 e 10. No entanto, uma vez que 10 é formado por dois dígitos que já são usados, usa-se X em vez de 10. Deste modo temos $d \in \{0, 1, \ldots 9, \mathrm{X}\}$.

Exemplo 4.1.1. Consideremos o ISBN do livro [3] da bibliografia deste trabalho: $0 - 19 - 853803 - 0$. Vamos calcular o dígito de controlo.

$$\begin{aligned} d = \sum_{i=1}^{9} i a_i &= 1 \times 0 + 2 \times 1 + 3 \times 9 + 4 \times 8 + 5 \times 5 + 6 \times 3 + 7 \times 8 + 8 \times 0 + 9 \times 3 \\ &= 187 = 17 \times 11 = 0 \mod 11. \end{aligned}$$

Olhando para o ISBN considerado, verifica-se que, de facto, o dígito de controlo é 0.

Vamos verificar de seguida que o ISBN com 10 dígitos detecta erros de substituição e de transposição.

Lema 4.1.2. O ISBN com 10 dígitos detecta 1 erro de substituição.

Demonstração. Suponhamos que o ISBN correcto é $a_1a_2a_3a_4a_5a_6a_7a_8a_9d$ mas ao imprimir o livro ocorreu um erro de substituição, obtendo-se o ISBN

$$b_1b_2b_3b_4b_5b_6b_7b_8b_9d'.$$

Suponhamos que $a_i = b_i$, $i \in \{1, \cdots 9\} \setminus \{j\}$ e $a_j \neq b_j$. Suponhamos $b_j = a_j + k$ com $k \in [-9, 9] \setminus \{0\}$. Como

$$\sum_{i=1}^{9} ib_i = \sum_{i=1, i\neq j}^{9} ia_i + j(a_j + k) = \sum_{i=1, i\neq j}^{9} ia_i + ja_j + jk = \sum_{i=1}^{9} ia_i + jk,$$

para o erro não ser detectado teria de se verificar

$$\sum_{i=1}^{9} ia_i + jk = d \mod 11.$$

Como

$$\sum_{i=1}^{9} ia_i = d \mod 11,$$

concluiríamos que $jk = 0 \mod 11$, ou seja, 11 divide jk. Como 11 é primo então 11 divide j ou 11 divide k mas tal não pode acontecer pois $k \in [-9, 9] \setminus \{0\}$ e $1 \leq j \leq 9$. Concluímos que $d \neq d'$ e assim o erro é detectado. □

Lema 4.1.3. O ISBN com 10 dígitos detecta 1 erro de transposição.

Demonstração. Suponhamos que o ISBN correcto é $a_1a_2a_3a_4a_5a_6a_7a_8a_9d$ mas na impressão do livro obteve-se o ISBN $b_1b_2b_3b_4b_5b_6b_7b_8b_9d'$. Suponhamos ainda que ocorreu um erro de transposição. Seja $a_i = b_i$, $i \in \{1, \cdots, 9\} \setminus \{j, k\}$, $b_j = a_k$, $b_k = a_j$ e $a_j \neq a_k$. Atendendo a que

$$d' = \sum_{i=1}^{9} ib_i \mod 11,$$

obtemos

$$\begin{aligned}\sum_{i=1}^{9} ib_i &= \sum_{i=1, i\neq j,k}^{9} ia_i + ja_k + ka_j \\ &= \sum_{i=1, i\neq j,k}^{9} ia_i + ja_k + ka_j + ja_j + ka_k - ja_j - ka_k \\ &= \sum_{i=1}^{9} ia_i + ja_k + ka_j - ja_j - ka_k \\ &= \sum_{i=1}^{9} ia_i + (j-k)(a_k - a_j)\end{aligned}$$

Para o erro não ser detectado então, teria de se verificar

$$\sum_{i=1}^{9} ia_i + (j-k)(a_k - a_j) \mod 11.$$

Como

$$d = \sum_{i=1}^{9} ia_i \mod 11,$$

concluiríamos que

$$(j-k)(a_k - a_j) = d \mod 11 \Leftrightarrow 11 \mid (j-k)(a_k - a_j)$$

e, como 11 é primo,

$$11 \mid (j-k)(a_k - a_j) \Leftrightarrow 11 \mid (j-k) \vee 11 \mid (a_k - a_j).$$

Como $1 \leq j \leq 9$ e $1 \leq k \leq 9$ então $-8 \leq j-k \leq 8$. Uma vez que $j \neq k$ conclui-se que 11 não divide $j-k$. Analogamente, como $-9 \leq a_k - a_j \leq 9$ e $a_k \neq a_j$, 11 não divide $a_k - a_j$. Portanto $d \neq d'$ e o erro é detectado. □

4.1.2 ISBN com 13 dígitos

Os dígitos do ISBN com 13 dígitos têm o seguinte significado:

- 1º a 3º dígitos - Prefixo EAN;
- 4º e 5º dígitos - Identificador de Grupo, País ou Área Idiomática;
- 6º ao 8º dígitos - Identificador de Editor;
- 9º ao 12º dígitos - Identificador de Título;

- 13º dígito - Dígito de Controlo.

Por exemplo, o ISBN do livro *A Matemática das Coisas* de Jorge Buescu é

$$978 - 989 - 616 - 241 - 2$$

onde

- 978 - Prefixo EAN;
- 98 - Portugal;
- 961 - Editora Gradiva;
- 6241 - Identificador de Título;
- 2 - Dígito de Controlo.

O dígito de controle d do ISBN de 13 dígitos é calculado da seguinte forma:

$$d = 10 - (a_1 + 3a_2 + a_3 + \cdots + a_{11} + 3a_{12}) \mod 10.$$

Da soma módulo 10 obtém-se um número entre 0 e 9 e consequentemente, quando posteriormente se subtrai esse número a 10, obtém-se um número entre 1 e 10, o qual será o dígito de controle.

Vejamos um exemplo. Consideremos o ISBN do livro *A Matemática das Coisas* de Jorge Buescu:

$$978 - 989 - 616 - 241 - 2.$$

Vamos calcular o dígito de controlo.

$$\begin{aligned} d &= 10 - (9 + 3 \times 7 + 8 + 3 \times 9 + 8 + 3 \times 9 + 6 + 3 \times 1 + 6 + 3 \times 2 + 4 + 3 \times 1) \\ &= 10 - 128 = -118 = 2 \mod 10 \end{aligned}$$

O ISBN considerado apresenta de facto o dígito de controlo 2.

Vamos verificar que o ISBN com 13 dígitos detecta erros de substituição.

Lema 4.1.4. O ISBN com 13 dígitos detecta erros de substituição.

Demonstração. Suponhamos que o ISBN correcto é $a_1a_2a_3a_4a_5a_6a_7a_8a_9a_{10}a_{11}a_{12}d$ mas na impressão do livro obteve-se o ISBN $b_1b_2b_3b_4b_5b_6b_7b_8b_9b_{10}b_{11}b_{12}d'$. Suponhamos que

ocorreu um erro. Seja $a_i = b_i, i \in \{1, \cdots 12\} \setminus \{j\}$ e $a_j \neq b_j$. Suponhamos $b_j = a_j + k$ onde $k \in [-9, 9] \setminus \{0\}$. Como vimos

$$d' = 10 - (b_1 + 3 \times b_2 + b_3 + 3 \times b_4 + \cdots + b_{11} + 3 \times b_{12}) \mod 10.$$

Suponhamos que o erro está numa das posições ímpares. Sem perda de generalidade, suponhamos que está na posição b_1. Assim

$$d' = 10 - (a_1 + k + 3 \times a_2 + a_3 + 3 \times a_4 + \cdots + a_{11} + 3 \times a_{12}) \mod 10.$$

O erro não é detectado quando:

$$\begin{aligned} &a_1 + 3 \times a_2 + a_3 + 3 \times a_4 + \cdots + a_{11} + 3 \times a_{12} \\ &\quad = a_1 + k + 3 \times a_2 + a_3 + 3 \times a_4 + \cdots + a_{11} + 3 \times a_{12} \mod 10 \end{aligned}$$

ou seja quando $a_1 = a_1 + k \mod 10$. Como $k \in [-9, 9] \setminus \{0\}$ então $a_1 \neq a_1 + k \mod 10$ portanto o erro é detectado. De forma idêntica se prova que o erro também é detectado quando ocorre numa das posições pares. □

Ao contrário do ISBN com 10 dígitos, o ISBN com 13 dígitos nem sempre detecta erros de transposição. O exemplo seguinte comprova esta afirmação.

Exemplo 4.1.5. Consideremos novamente o ISBN do livro *A Matemática das Coisas* de Jorge Buescu:

$$978 - 989 - 616 - 241 - 2$$

e suponhamos que existe um erro de impressão que transpõe duas posições (a oitava e a nona):

$$978 - 989 - 661 - 241 - d'.$$

Calculemos d'. Temos

$$\begin{aligned} d' &= 10 - (9 + 3 \times 7 + 8 + 3 \times 9 + 8 + 3 \times 9 + 6 + 3 \times 6 + 1 + 3 \times 2 + 4 + 3 \times 1) \\ &= 10 - 138 = -128 = 2 - 13 \times 10 = 2 \mod 10 \end{aligned}$$

Obtivémos $d' = 2$ que é o mesmo dígito de controle do ISBN original. Portanto o erro não seria detectado. Isto aconteceu porque $3 \times 6 + 1 = 19$, $3 \times 1 + 6 = 9$ e temos $19 = 9 \mod 10$.

Claro que se trata de um caso particular. Noutros casos o erro seria detectado.

4.2 O Bilhete de Identidade

Também no caso do Bilhete de Identidade temos um dígito de controlo. Dado um número de Bilhete de Identidade $x_1x_2x_3x_4x_5x_6x_7x_8$, o dígito de controle calcula-se fazendo:

$$d = 11 - (9x_1 + 8x_2 + 7x_3 + 6x_4 + 5x_5 + 4x_6 + 3x_7 + 2x_8) \mod 11$$

Note-se que, se o número tiver menos de 8 dígitos adicionam-se quantos zeros forem necessários à esquerda do número de modo a obter um total de 8 dígitos.

Vejamos um exemplo.

Exemplo 4.2.1. O número do Bilhete de Identidade da autora deste trabalho é

$$12386649 - 9.$$

Vamos calcular o dígito de controlo.

$$\begin{aligned} d &= 11 - (9 \times 1 + 8 \times 2 + 7 \times 3 + 6 \times 8 + 5 \times 6 + 4 \times 6 + 3 \times 4 + 2 \times 9) \\ &= 11 - 178 = -167 = 9, \end{aligned}$$

que de facto é o dígito de controlo do número de Bilhete de Identidade considerado.

Pode verificar-se que o dígito de controle do BI permite detectar um erro de substituição em qualquer posição ou um erro de transposição entre quaisquer posições.

4.3 Uma aula sobre códigos

Nesta secção, o texto apresentado a itálico corresponde a um proposto diálogo do professor com os alunos. O texto que não se encontra em itálico contém algumas considerações sobre a aula bem como exercícios a propor aos alunos.

É do conhecimento global que sempre se transmitiram mensagens secretas entre amantes, diplomatas, políticos e militares. Além disso é generalizada a utilização de códigos nos mais variados contextos: para identificar livros e revistas (o código ISBN para livros, do qual vamos falar, e o código ISSN para revistas), para identificar pessoas (bilhete de identidade, passaporte, cartão de contribuinte), para distinguir diferentes produtos (os vulgares códigos de barras usados nos estabelecimentos comerciais), para podermos comunicar através da internet, etc.

Com a evolução da tecnologia sentiu-se a necessidade de melhorar a forma de codificar

mensagens de modo a torná-las mais seguras e fiáveis. Assim, aprofundou-se o estudo da Teoria de Códigos, a qual se tornou cada vez mais presente em situações do dia a dia.
A ideia base da Teoria de Códigos é conseguir transmitir informação de forma segura e fiável, sendo importante a capacidade de detectar e corrigir erros numa mensagem. Em Teoria de Códigos assumimos que existe uma fonte que transmite uma mensagem. De seguida a informação a transmitir passa num codificador onde, com a ajuda de uma função, é codificada, recorrendo a um certo código. Após a sua codificação, a mensagem é enviada através de um canal (onde poderão ocorrer erros) até chegar ao chamado descodificador. Nesta fase, o objectivo é que os erros sejam detectados e corrigidos para, posteriormente, a mensagem original ser transmitida ao receptor. Se conseguirmos fazer tudo isso cumprimos o nosso objectivo.

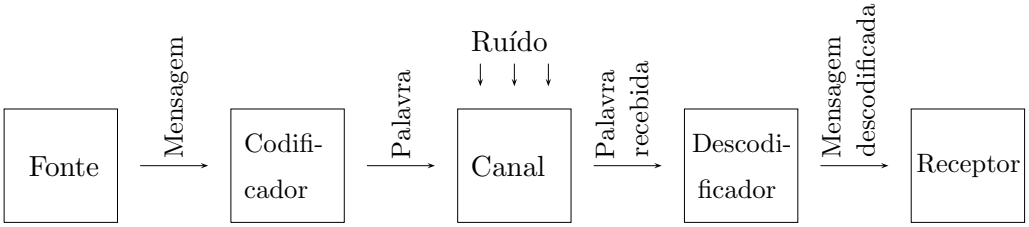

Figura 4.1: Processo de transmissão de uma mensagem

Vamos supor que queremos passar informação a um escuteiro de forma a que ele consiga seguir o caminho correcto do acampamento a uma fonte de modo a poder abastecer-se de água. Ora, as indicações correctas seriam

NNNEESEENNONEE

em que N, S, E e O representam, respectivamente Norte, Sul, Este e Oeste. Vamos usar códigos binários (ou seja códigos que apenas usam os algarismos 0 e 1) para codificar esta mensagem.

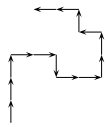

Figura 4.2: NNNEESEENNONEE

Uma codificação possível para esta mensagem seria:

00000010100110100000011001010

em que $N = 00$, $S = 01$, $E = 10$ e $O = 11$. O código pode assim ser constituído por quatro palavras cada uma com dois dígitos. Temos assim um código, ou seja um conjunto de palavras, que nos permitem escrever mensagens com N, S, E e O.

Problema a propor aos alunos 4.3.1. Pede-se a cada aluno que escolha um código, distinto do apresentado e que permita codificar a mensagem anterior usando dígitos entre 0 e 9. Pede-se de seguida aos alunos que tentem enumerar as vantagens e desvantagens do código que escolheram.

Se olharmos com um bocadinho de atenção para as palavras dadas é fácil verificar que a palavra 00 *é "mais parecida" ou "está mais próxima" da palavra* 01 *do que da palavra* 11. *Em Teoria de Códigos podemos quantificar essa parecença definindo o quão afastadas estão duas palavras. Dada uma palavra $c_1 \cdots c_n$, para cada $i = 1, \ldots, n$ dizemos que c_i é o dígito na posição i. Por exemplo na palavra* 01001, *o dígito na posição* 3 *é* 0 *e o dígito na posição* 2 *é* 1. *Vamos então definir uma forma de medir a distância entre duas palavras do mesmo tamanho de um dado código.*

Definição 4.3.2. Designa-se por distância entre duas palavras de código com o mesmo tamanho n, $c_1 \cdots c_n$ e $d_1 \cdots d_n$, e denota-se por $d(c_1 \cdots c_n, d_1 \cdots d_n)$ o número de posições em que as palavras diferem.

Por exemplo, $d(00, 01) = 1$ uma vez que as palavras 00 *e* 01 *diferem exactamente numa posição (a posição* 2*) e $d(001, 100) = 2$ uma vez que as palavras* 001 *e* 100 *diferem exactamente em duas posições (as posições* 1 *e* 3*). Convém notar que só podemos calcular a distância entre duas palavras com o mesmo tamanho.*

Problema a propor aos alunos 4.3.3. Pede-se a cada aluno que calcule as distâncias entre as várias palavras do código que escolheram.

Podemos considerar a distância mínima entre as palavras num dado código. Consideremos o código do exemplo inicial e designando este por C temos:

$$C = \{00, 01, 10, 11\}\,.$$

Podemos calcular as distâncias entre palavras de código para este código. Temos

$$d(00, 01) = 1,\ d(00, 10) = 1,\ d(00, 11) = 2,$$

$$d(01,10) = 2,\ d(01,11) = 1\ e\ d(10,11) = 1.$$

Portanto, a distância mínima deste código é igual a 1.

Depois de observar os códigos escolhidos pelos alunos, estes poderiam ser comparados tendo em conta o tamanho das palavras, a distância mínima entre as palavras e ao número de dígitos usados no código. De seguida, discutir-se-ião as vantagens e desvantagens de cada escolha com alunos, levando-os a concluir que o ideal seria ter um código com palavras de dimensão tão pequena quanto possível (para facilitar a transmissão), muitas palavras (para conseguir enviar muita informação) e distância mínima elevada (para aumentar a capacidade do código detectar e corrigir erros).

Com o fim de entendermos melhor o que significa a detecção de erros vejamos um exemplo bem conhecido de todos - o número do Bilhete de Identidade (BI). Muitos de nós já ouviram, em algum momento da nossa vida, que o número que está à direita do número do BI indica o número de pessoas que existem com o nosso nome. Nada mais falso, na verdade esse dígito é um dígito de controle que permite detectar se um dado número pode ser um número de um bilhete de identidade. Vamos ver como se calcula o dígito de controle. Para um dado número de Bilhete de Identidade $x_1x_2x_3x_4x_5x_6x_7x_8$*, o dígito de controle calcula-se da seguinte forma: primeiro calculamos a soma*

$$9x_1 + 8x_2 + 7x_3 + 6x_4 + 5x_5 + 4x_6 + 3x_7 + 2x_8;$$

de seguida calculamos o resto da divisão do resultado da soma por 11, o qual vamos designar por r*; por último calculamos o número* $d = 11 - r$*.*

Problema a propor aos alunos 4.3.4. Pede-se aos alunos que calculem o dígito de controle dos seus Bilhetes de Identidade a partir dos primeiros dígitos. Deve salientar-se o facto de que, caso o número tenha menos de oito dígitos, devemos acrescentar zeros à esquerda do número até perfazer oito dígitos.

Vamos agora falar um pouco de um outro código: o código ISBN. O ISBN é um número internacional para identificar livros. Se olharem para a parte de trás dos vossos manuais vêm um número com treze dígitos. O décimo terceiro dígito é um dígito de controle. Para calcular o digito de controle d' *de um dado ISBN com 13 dígitos*

$$a_1a_2a_3a_4a_5a_6a_7a_8a_9a_{10}a_{11}a_{12}d'$$

começa-se por calcular a soma

$$a_1 + 3a_2 + a_3 + 3a_4 + \cdots + a_{11} + 3a_{12} \tag{4.1}$$

e de seguida calculamos o resto do resultado da divisão desta soma por 10, o qual vamos designar por r'*. Por fim fazemos* $d' = 10 - r'$*.*

Problema a propor aos alunos 4.3.5. Pede-se aos alunos para calcularem vários dígitos de controle de livros conhecidos, confirmando os resultados.

O ISBN é um código que permite detectar um erro de impressão num dígito. Por exemplo, se no ISBN $978 - 989 - 616 - 241 - 2$ *o dígito* 7 *for substituído pelo dígito* 9 *temos*

$$9 + 3 \times 9 + 8 + 3 \times 9 + 8 + 3 \times 9 + 6 + 3 \times 1 + 6 + 3 \times 2 + 4 + 3 \times 1 = 134$$

e, como o resto da divisão de 134 *por* 10 *é* 4*, já não obtemos o mesmo dígito de controle (obtemos o dígito de controlo* 4 *em vez de* 2*).*

Problema a propor aos alunos 4.3.6. Pede-se aos alunos para simularem erros num dígito dos ISBN considerados anteriormente, substituindo um dos dígitos por um número distinto, e verificarem que já não obtém o mesmo dígito de controle.

Quando fazemos esta operação, ou seja, quando um dos dígitos é substituído por um dígito distinto (por exemplo em consequência de um engano no código correspondente ao editor) dizemos que estamos perante um erro de substituição. Ora, como acabaram de verificar o ISBN de 13 *dígitos detectou o erro de substituição que introduziram em cada uma da palavras de código consideradas. Será que isto acontece sempre que ocorre um (único) erro de substituição em algum dos dígitos? E, em caso afirmativo, porque é que isto acontece?*

Por exemplo, vamos supor que no dígito a_3 *temos um erro de substituição. Isto corresponde a adicionar uma constante* c *a* a_3 *obtendo um novo dígito entre 0 e 9, distinto do anterior,* $a_3 + c$*.*

Para que $a_3 + c$ *esteja entre* 0 *e* 9 *fácil verificar que essa constante pode ser um natural*

entre $-a_3$ *e* $9-a_3$*, excluindo o zero (se* $c=0$ *não seria alterado o valor).*
Suponhamos que o ISBN correcto é

$$a_1a_2a_3a_4a_5a_6a_7a_8a_9a_{10}a_{11}a_{12}d$$

mas, como ocorreu o erro de substituição referido, obtemos

$$a_1a_2(a_3+c)a_4a_5a_6a_7a_8a_9a_{10}a_{11}a_{12}d.$$

Para provar que o ISBN com 13 *dígitos corrige um erro de substituição na terceira posição, vamos mostrar que este número já não corresponde a um ISBN. Para isso vamos calcular o novo dígito de controlo,* d'*, e verificar que* $d'\neq d$*. Sabemos que para calcular* d' *temos de calcular a soma*

$$a_1+3a_2+(a_3+c)+3a_4+\cdots+a_{11}+3a_{12}=c+a_1+3a_2+a_3+3a_4+\cdots+a_{11}+3a_{12}, \quad (4.2)$$

de seguida calcular o resto do resultado da divisão desta soma por 10*, a que vamos chamar* r'*, e por fim obtemos* $d'=10-r'$*.*
Seja r *o resto da divisão da soma* $a_1+3a_2+a_3+3a_4+\cdots+a_{11}+3a_{12}$ *por* 10*. Temos, de acordo com* (4.2)*, que* $r'=c+r$*. Assim* $d'=10-r'=10-r-c=d-c$ *com* $c\in\{-a_3,9-a_3\}\setminus\{0\}$*. Como* $c\neq 0$*, concluímos que* $d\neq d'$*, o que prova que o erro é detectado. De forma semelhante se prova que o erro também é detectado quando ocorre numa das outras posições.*

De seguida podia discutir-se esta demonstração com os alunos, tentando que estes se apercebam de que no caso de qualquer posição ímpar a ideia da demonstração é exactamente a mesma e pedindo-lhes que discutissem o que tinha de ser alterado para demonstrar o resultado no caso das posições pares e mesmo quando o erro ocorre no dígito de controlo. Seguidamente poder-se-ia discutir um exemplo concreto de um tipo de erro que não se consegue detectar (usaremos o ISBN do exemplo 4.1.5).

Verificamos de seguida que o ISBN de 13 *dígitos não permite detectar todos os tipos de erro. Vejamos o seguinte exemplo onde consideremos o ISBN*

$$978-989-616-241-2$$

e suponhamos que existe um erro de impressão que transpõe duas posições (a oitava e a nona):

$$978-989-661-241-d'.$$

Calculemos d'. Ora, começando pela soma em (4.1) *tem-se*

$$9 + 3 \times 7 + 8 + 3 \times 9 + 8 + 3 \times 9 + 6 + 3 \times 6 + 1 + 3 \times 2 + 4 + 3 \times 1 = 138.$$

Seguidamente, calculando o resto da divisão deste número por 10, obtemos

$$\frac{138}{10} = \frac{10 \times 13 + 8}{10} = 13 + \frac{8}{10}.$$

Portanto, o resto é 8*. Por último fazemos*

$$d' = 10 - 8 = 2.$$

Obtivemos assim $d' = 2$ que é o mesmo dígito de controle do ISBN original. Portanto o erro não seria detectado.

Poderíamos acabar esta aula referindo alguns aspectos do problema abordado que poderiam ser estudados mais aprofundadamente bem como alguma bibliografia elementar que os alunos interessados no tema poderiam consultar. Por exemplo, relativamente ao número do bilhete de identidade, o livro [1] (*O Mistério do Bilhete de Identidade* de Jorge Buescu).

Apêndices

Álgebra

Corpo

Sejam A um conjunto não vazio e $+ : A \times A \to A$ e $\cdot : A \times A \to A$ duas operações binárias. O terno $(A, +, .)$ diz-se um *corpo* se

a) existe $0 \in A$ tal que $a + 0 = a$ para todo o $a \in A$

b) $(a + b) + c = a + (b + c)$ para todos os $a, b, c \in A$

c) para todo o $a \in A$ existe $-a \in A$ tal que $a + (-a) = 0$

d) $a + b = b + a$ para todos os $a, b \in A$

e) existe $1 \in A \setminus \{0\}$ tal que $a \cdot 1 = a$ para todo o $a \in A$

f) $(a \cdot b) \cdot c = a \cdot (b \cdot c)$ para todos os $a, b, c \in A$

g) para todo o $a \in A \setminus \{0\}$ existe $a^{-1} \in A \setminus \{0\}$ tal que $a \cdot a^{-1} = 1$

h) $a \cdot b = b \cdot a$ para todos os $a, b \in A$

i) $a \cdot (b + c) = a \cdot b + a \cdot c$

Espaço vectorial

Seja E um conjunto não vazio, K um corpo e $+ : E \times E \to E$ e $\cdot : K \times E \to E$ duas operações binárias. Diz-se que o terno $(E, +, \cdot)$ é um *espaço vectorial* ou um *espaço linear* sobre o corpo K se verifica as propriedades seguintes:

a) existe $0_E \in E$ tal que $a + 0_E = a$ para todo o $a \in E$

b) $(a + b) + c = a + (b + c)$ para todos os $a, b, c \in E$

c) para todo o $a \in E$ existe $-a \in E$ tal que $a + (-a) = 0_E$

d) $a + b = b + a$ para todos os $a, b \in E$

e) para todos os $a, b \in A$ e $\alpha \in K \alpha \cdot (a + b) = \alpha \cdot a + \alpha \cdot b$;

f) para todo o $a \in A$ e $\alpha, \beta \in K$ temos $(\alpha + \beta) \cdot a = \alpha \cdot a + \beta \cdot a$;

g) para todo o $a \in A$ e $\alpha, \beta \in K$ temos $\alpha \cdot (\beta \cdot a) = (\alpha \cdot \beta) \cdot a$;

h) para todo o $a \in A$ temos $1 \cdot a = a$, em que 1 é a identidade de K.

Seja $(E, +, \cdot)$ um espaço vectorial sobre um corpo K e sejam $x_1, \ldots, x_n \in E$. Um vector v diz-se uma *combinação linear* dos vectores $x_1, \ldots, x_n$ se existirem escalares $\lambda_1, \ldots, \lambda_n \in K$ tais que

$$v = \lambda_1 x_1 + \ldots + \lambda_n x_n$$

Sejam E um espaço vectorial sobre um corpo K e $x_1, \ldots, x_n \in E$. Diz-se que os vectores $x_1, \ldots, x_n$ são *linearmente independentes* se a *única* combinação linear nula de $x_1, \ldots, x_n$ é a trivialmente nula, isto e, se

$$\lambda_1 x_1 + \cdots + \lambda_n x_n = 0_E \Rightarrow \lambda_1 = \cdots = \lambda_n = 0.$$

Seja E um espaço vectorial sobre um corpo K e C um subconjunto não vazio de E. Diz-se que C é um *conjunto gerador* de E ou que C gera E se qualquer vector de E se pode escrever como combinação linear de vectores pertencentes a C. Um conjunto de vectores de E diz-se uma *base (de Hamel)* de E se gera E e é constituído por vectores linearmente independentes.

Congruências

Sejam $q \in \mathbb{N}$ e $a, b \in \mathbb{Z}$ dizemos que a é *congruente* com b módulo q, e denota-se $a = b \mod q$, se existe $k \in \mathbb{Z}$ tal que $a = b + kq$, ou seja se q divide $a - b$. Naturalmente, dado $q \in \mathbb{N}$, existe um e um só $k \in \{0, \ldots, q - 1\}$ tal que $a = k \mod q$.

A relação de congruência módulo q define uma relação de equivalência em $\mathbb{Z}$, isto é, dados $a, b, c \in \mathbb{Z}$, tem-se:

a) Reflexividade: $a = a \mod q$;

b) Simetria: se $a = b \mod q$ então $b = a \mod q$;

c) Transitividade: se $a = b \mod q$ e $b = c \mod q$ então $a = c \mod q$.

A relação de congruência módulo q divide $\mathbb{N}$ em q conjuntos que designamos por *classes equivalência*. De facto, dado $k \in \{0, \ldots, q-1\}$ designamos por *classe de equivalência de* k o conjunto

$$[k] = \{p \in \mathbb{Z} : p = k \mod q\}$$

e temos $\mathbb{N} = [0] \cup \cdots \cup [q-1]$ e $[i] \cap [j] = \emptyset$ sempre que $i, j \in \{0, \ldots, q-1\}$, $i \neq j$.

Sejam $q \in \mathbb{N}$ e $a, b, c, d \in \mathbb{Z}$. Se $a = b \mod q$ e $c = d \mod q$ temos

a) $a + c = b + d \mod q$;

b) $a - c = b - d \mod q$;

c) $ac = bd \mod q$.

Combinatória

Dado um conjunto com n elementos, o número de sequências de k elementos que podemos extrair do conjunto, sem reposição, sendo relevante a ordem por que são extraídos os elementos, designa-se por *arranjos de n elementos k a k* e denota-se por A_k^n. Em particular, se $k = n$ estamos a usar todos os elementos disponíveis, e A_n^n corresponde a todas as ordenações possíveis de um conjunto com n elementos. Estes arranjos de todos os n elementos do conjunto são comummente referidos como *permutações*, e em lugar de A_n^n escreve-se P_n. Obviamente,

$$P_n = n! = n \times (n-1) \times \ldots \times 1.$$

O número de subconjuntos com k elementos extraíveis de um conjunto com n elementos ($n \geq k$) é denotado por $\binom{n}{k}$, símbolo que se designa por *combinações de n, k a k*. Pode mostrar-se que

$$\binom{n}{k} = \frac{n!}{k!(n-k)!}.$$

Teorema 4.3.7 (Teorema Binomial)**.** Se $x, y \in \mathbb{R}$ e $n \in N$ é válida a fórmula

$$(x+y)^n = \sum_{k=0}^{n} \binom{n}{k} x^k y^{n-k}.$$

Bibliografia

[1] J. Buescu, *O Mistério do Bilhete de Identidade e Outras Histórias*, Gradiva, 2002.

[2] R. L. Fernandes, M. Ricou, *Introdução à Álgebra*, IST Press, 2004.

[3] R. Hill, *A first course in coding theory*, Oxford University Press, 1986.

[4] J. H. van Lint, *Introduction to coding theory*, 3^{rd} edition, Graduate Texts in Mathematics 86, Springer, 1999.

[5] E. Giraldes, H. Fernandes, M. Smith, *Curso de Álgebra Linear e Geometria Analítica*, Mc Graw Hill, 1995.

[6] M. J. Golay, *Notes on Digital Coding*, Proc. of the Inst. of Radio Engineers **37** (1949) 657.

[7] R. W. Hamming, *Error detecting and error correcting codes*, Bell Syst. Tech. J. **29** (1950) 147–160.

[8] J.H. van Lint, *On the nonexistence of certain perfect codes, in Computers in Number Theory (Atkin and Birch, Eds.)*, Acad. Press, New York, 1971.

[9] F. J. MacWilliams, N. J. Sloan, *The theory of error-correcting codes*, North-Holland, Amsterdam, 1977.

[10] S. Nápoles, L. Santos, E. Veloso, *A Matemática na formação inicial de professores*, Educação e Matemática **91** (2007) 90-95.

[11] C. E. Shannon, *A mathematical theory of communication*, Bell Syst. Tech. J. **27** (1948) 379-423.

[12] A. Tietäväinen, *On the nonexistence of perfect codes over finite fields*, SIAM J. Appl. Math. **24** (1973) 88-96.