2019: DISSERTAÇÃO DE MESTRADO
Mestrado Profissional em Matemática - PROFMAT, Universidade Federal de São João Del-Rei - UFSJ
Sociedade Brasileira de Matemática - SBM

Aplicações de Álgebra Linear na Compressão de Imagens.

Guilherme Fernandes Carias Barbosa [1]
Humberto Cesar Fernandes Lemos[2]

Resumo: Este trabalho tem como intuito ilustrar algumas ideias básicas da Álgebra Linear aplicadas ao processo de compressão de imagens com perdas. A título de exemplo, o processo descrito utiliza principalmente a transformada discreta do cosseno (DCT, do inglês *discrete cosine transform*), além disso utiliza-se a codificação de Huffman e é descrito os passos para a compressão em formato JPEG. Trataremos da compressão com perdas, pois a nossa ênfase está nos princípios básicos da compressão e a Álgebra Linear no processo, lembrando que nosso interesse não é melhorar o armazenamento nem conservar a qualidade da imagem.

Palavras-chave: Transformada discreta do cosseno, Aplicação da DCT, compressão JPEG, codificação de Huffman, compressão de imagens com perda.

Abstract: This work aims to illustrate some basic ideas of Linear Algebra applied to the lossy image compression process. As an example, we describe a process that mainly uses the discrete cosine transform (DCT). In addition, Huffman encoding is used and the steps for compression in JPEG format are described. We will deal with lossy compression, as our emphasis is on the basic principles of compression and Linear Algebra topics used in the process, remembering that our interest is not to improve storage or preserve image quality.

Keywords: Discrete cosine transform, DCT, JPEG, Huffman encoding, lossy image compression.

1 Introdução

Com o passar do tempo tudo evolui e cria alguma necessidade, a matemática, a vida, o corpo humano, a informática e etc. Nos dias atuais, notamos um alto crescimento na busca por melhores imagens digitais e melhoramento dos dispositivos de captura que proporcionam o máximo de qualidade, com isso surge a necessidade de otimizar o compartilhamento de imagens. Com isso, métodos de compressão vêm surgindo ao longo dos

[1]Aluno de Mestrado do PROFMAT, Turma 2018, Universidade Federal de São João Del-Rei - UFSJ, guilherme_fcb@yahoo.com.br

[2]Professor orientador, Departamento de Estatística, Física e Matemática - DEFIM, UFSJ, humbertolemos@ufsj.edu.br

anos de acordo com a necessidade para o armazenamento e agilidade na transmissão de imagens.

Quando usamos algum algoritmo para compressão de imagens devemos levar em consideração alguns fatores de extrema importância, como diminuir de forma substancial o espaço ocupado, o quanto de qualidade é possível perder sem que note alguma alteração em sua visualização, o método de compressão mais viável devido a otimização de tempo, qualidade oferecida e melhor taxa de compressão oferecida. Os métodos por transformada são os mais usados, pela boa capacidade de compressão e pela qualidade oferecida ao final, esses métodos convertem dados de domínio espacial em domínio de frequência.

Aqui, neste trabalho, abordaremos o método de compressão para o formato JPEG, porém nosso foco é na Álgebra Linear por trás desse processo, onde em cada um dos passos descritos é possível notar alguns tópicos, contudo ao longo do texto serão expostos de forma clara.

Na seção 2 introduzimos alguns conceitos de Álgebra Linear que consideramos básicos e fundamentais para se tratar dos processos que aqui serão descritos. Na seção 3 falamos sobre os estudos de Fourier, em particular sobre a série de Fourier. Usando a série de Fourier e suas propriedades, definimos tanto a transformada de Fourier, quanto suas versões discretas, a saber a transformada Discreta de Fourier e, finalmente, a transformada discreta do cosseno, que é de grande importância nesse processo de compressão. Já na seção 4 tratamos brevemente a respeito de noções de programação, onde abordamos um pouco sobre o formato JPEG e também sobre a codificação de Huffman. Na seção 5, é possível notar a ligação entre as seções anteriores e aplicação no processo de compressão quando apresentamos as principais ideias do processo de compressão, fazendo alguns exemplos que mostram o passo a passo, e chamamos a atenção para o uso da Álgebra Linear neste processo. Na seção 6 sugerimos três atividades contextualizadas e interdisciplinares, que podem ser desenvolvidas no ensino básico, com isso mostrando aos alunos que por trás das fotos que eles tiram e enviam existe matemática envolvida. E por último, na seção 7, temos as considerações finais.

2 Noções de Álgebra Linear

Nesta seção serão apresentado e definidos alguns itens importantes de álgebra linear, que possuem aplicação direta no processo de compressão de imagens com perdas no formato JPEG. Não pretendemos apresentar um conteúdo completo de Álgebra Linear, para isso recomendamos a consulta em material no assunto em especifico, como por exemplo os que aqui usamos [1], [2], [3] e [5].

2.1 Espaço Vetorial

Definimos espaço vetorial como qualquer conjunto não-vazio $\mathbb{V}$, munido das operações de adição entre dois vetores do espaço $(+)$ e multiplicação $(\cdot)$ sobre um corpo $\mathbb{K}$, os elementos do corpo $\mathbb{K}$ são denominados *escalares*. As operações de adição de vetores e multiplicação por escalar devem obedecer aos seguintes axiomas:

A: Axiomas da adição: dados $u, v, w \in \mathbb{V}$, tem-se:

$$\begin{aligned} A_1 &: u + v = v + u; \\ A_2 &: u + (v + w) = (u + v) + w; \\ A_3 &: \exists\ u^* \text{ tal que } u + u^* = u^* + u = u; \\ A_4 &: \exists - u \text{ tal que } u + (-u) = (-u) + u = u^*. \end{aligned}$$

M: Axiomas da multiplicação: dados $u, v, \in \mathbb{V}$ e $\alpha, \beta \in \mathbb{K}$, tem-se:

$$\begin{aligned} M_1 &: \alpha(\beta u) = (\alpha\beta)u; \\ M_2 &: \alpha(u + v) = \alpha u + \alpha v; \\ M_3 &: (\alpha + \beta)u = \alpha u + \beta u; \\ M_4 &: \exists\ 1 \in \mathbb{K} \text{ tal que } 1u = u. \end{aligned}$$

Nesse texto, consideraremos sempre $\mathbb{K} = \mathbb{R}$, ou seja, trabalharemos sobre espaços vetoriais reais. Porém existem espaços vetoriais sobre outros corpos, um exemplo comum são espaços vetoriais complexos, que são aqueles que atuam sobre sobre $\mathbb{C}$. Podemos citar como exemplos de espaços vetoriais reais:

- o plano $\mathbb{R}^2$ e o espaço $\mathbb{R}^3$;
- os espaços $\mathbb{R}^n$ em geral;
- as matrizes $M_{nm}(\mathbb{R})$ de entradas reais;
- os polinômios $\mathcal{P}_n$;
- as funções reais de uma variável f.

A demonstração de que alguns desses conjuntos acima listados são espaços vetoriais pode ser vista em [22]

Em todos os contextos, $u \in \mathbb{V}$ é chamado vetor do espaço $\mathbb{V}$. Relembraremos a seguir mais alguns conceitos importantes no estudo de espaços vetoriais.

Definição 2.1 *Sejam $\mathbb{V}$ um espaço vetorial e $v_1, ..., v_n \in \mathbb{V}$. Dizemos que o conjunto $\{v_1, ..., v_n\}$ é **linearmente independente** (LI), ou que os vetores $c_1, ..., v_n$ são LI, se a equação $a_1v_2 + ... + a_nv_n = 0$ implica que $a_1 = ... = a_n = 0$. No caso em que exista algum $a_i \neq 0$ dizemos que $\{v_1, ..., v_n\}$ é **linearmente dependente** (LD), ou que os vetores $v_1, ..., v_n$ são LD*

A definição de independência linear é importante para podermos definir a base de um espaço vetorial. O conceito de base será importante em todo o nosso trabalho.

Definição 2.2 *Um conjunto $\{v_1, ..., v_n\}$ de vetores de $\mathbb{V}$ é dito uma **base** de $\mathbb{V}$ se:*

i) O conjunto $\{v_1, ..., v_n\}$ *é LI,*

ii) o conjunto $\{v_1, ..., v_n\}$ *gera o espaço vetorial* $\mathbb{V}$*. Isso significa que para todo* $x \in \mathbb{V}$*, existem números reais* $x_1, \ldots, x_n$ *tais que*

$$x = x_1 v_1 + x_2 v_2 + \ldots + x_n v_n.$$

Os números $x_1, \ldots, x_n$ *são as* ***componentes*** *do vetor* x *na base.*

Podemos definir também a **dimensão** de um espaço vetorial como o número de vetores em uma base. Por exemplo, se $\{v_1, ..., v_n\}$ é uma base de $\mathbb{V}$, então dizemos que $\dim \mathbb{V} = n$.

Um importante conceito e que será brevemente abordado em nosso trabalho é o de subespaço vetorial. No entanto não iremos nos alongar demais aqui, o leitor interessado pode consultar [1], [2], [3] e [5].

Definição 2.3 *Dado um espaço vetorial* $\mathbb{V}$ *sobre* $\mathbb{R}$*, um subconjunto* $\mathbb{W}$*, não vazio, será um* ***subespaço vetorial*** *de* $\mathbb{V}$ *se:*

i) Para quaisquer $u, v \in \mathbb{W}$ *tem-se* $u + v \in \mathbb{W}$*.*

ii) Para quaisquer $\alpha \in \mathbb{R}$ *e* $u \in \mathbb{W}$ *tem-se* $\alpha u \in \mathbb{W}$*.*

A noção de subespaço nos permite, entre outras coisas, trabalhar com apenas a parte do espaço vetorial que nos será relevante para o problema. No caso do nosso texto, que fala sobre o processo de compressão, essa é uma maneira simples de economizar recursos computacionais.

2.2 Produto Interno

Definiremos agora o importante conceito de produto interno, por vezes conhecido também como produto escalar. Embora a definição de produto interno possa ser dada em um espaço vetorial $\mathbb{V}$ sobre qualquer corpo numérico, iremos usar a definição dada sobre o corpo dos reais, pois só iremos trabalhar este caso ao longo do texto.

Definição 2.4 *Seja* $\mathbb{V}$ *um espaço vetorial sobre o corpo dos reais* $\mathbb{R}$*. Um aplicação* $\langle \cdot, \cdot \rangle : \mathbb{V} \times \mathbb{V} \mapsto \mathbb{R}$ *é um* ***produto interno*** *caso ela satisfaça às propriedades a seguir:*

i) $\langle u, u \rangle \geq 0$*, para todo* $u \in \mathbb{V}$*. Além disso temos que* $\langle u, u \rangle = 0$ *se, e somente se* u*, for o vetor nulo.*

ii) $\langle v, u \rangle = \langle u, v \rangle$*,* $\forall\ u, v \in \mathbb{V}$*.*

iii) $\langle u, \alpha v \rangle = \alpha \langle u, v \rangle$*,* $\forall\ u, v \in \mathbb{V}$*, e* $\forall\ \alpha \in \mathbb{R}$*.*

iv) $\langle u, v + w \rangle = \langle u, v \rangle + \langle u, w \rangle$*,* $\forall\ u, v, w \in \mathbb{V}$*.*

Um importante exemplo é o produto interno canônico em $\mathbb{R}^n$.

Exemplo 2.1 *Sejam dois vetores* $x = (x_1, ..., x_n)$ *e* $y = (y_1, ..., y_n) \in \mathbb{R}^n$*.*

(a) O ***produto interno canônico*** *de x e y é dado por*

$$x \cdot y = x_1y_1 + x_2y_2 + ... + x_ny_n = \sum_{i=1}^{n} x_iy_i.$$

(b) Definimos a ***norma*** *de um vetor $x = (x_1, ..., x_n) \in \mathbb{R}^n$ por*

$$\|x\| = \sqrt{x \cdot x} = \sqrt{x_1^2 + ... + x_n^2} = \sqrt{\sum_{i=1}^{n} x_i^2} \quad .$$

Escrevendo dois vetores $x, y \in \mathbb{R}^n$ como matrizes colunas

$$x = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} \quad \text{e} \quad y = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix},$$

o produto interno desses vetores pode ser escrito em termos do produto de matrizes como

$$x \cdot y = x^t y,$$

onde x^t é a matriz transposta de x.

OBS.: Optamos por definir o produto interno canônico pelo símbolo $x \cdot y$, e reservamos $\langle x, y \rangle$ para um produto interno arbitrário.

Exemplo 2.2 *Sejam $v = (3, -4, 1, 0, -1, 5)$ e $w = (2, 2, -1, 6, 1, -3)$ vetores do $\mathbb{R}^6$. O produto escalar entre v e w é dado por*

$$v \cdot w = (3)(2) + (-4)(2) + (1)(-1) + (0)(6) + (-1)(1) + (5)(-3) = -19.$$

As normas de v e w são respectivamente dadas por

$$\|v\| = \sqrt{3^2 + (-4)^2 + 1^2 + 0^2 + (-1)^2 + 5^2} = \sqrt{53} \ ,$$
$$\|w\| = \sqrt{2^2 + 2^2 + (-1)^2 + 6^2 + 1^2 + (-3)^2} = \sqrt{55}.$$

Definição 2.5 *Seja $\mathbb{V}$ um espaço vetorial com produto interno. Dizemos que dois elementos $u, v \in \mathbb{V}$ são* ***ortogonais*** *se, e somente se, $\langle u, v \rangle = 0$.*

Observe que, no caso em que $\mathbb{V} = \mathbb{R}^2$ ou $\mathbb{V} = \mathbb{R}^3$, o produto interno canônico entre dois vetores $u, v \in \mathbb{V}$ pode ser expresso como

$$u \cdot v\rangle = \|u\| \, \|v\| \cos\theta,$$

onde $0 \leq \theta \leq \pi$ é o ângulo entre os vetores u e v. Supondo u e v não nulos, eles são ortogonais se, e somente se, temos $\theta = \pi/2$.

Proposição 2.1 *Se $v_1, ..., v_k$ são vetores* ***não nulos de*** $\mathbb{R}^n$ ***ortogonais****, isto é, $v_i \cdot v_j = 0$, para $i \neq j$, então o conjunto $\{v_1, ..., v_k\}$ é LI.*

Demonstração: Considere a equação vetorial, lembrando que $\overline{0}$ = (0,...,0) é o vetor nulo e com $x_i \in \mathbb{R}$.

$$x_1 v_1 + \ldots + x_k v_k = \overline{0}. \tag{1}$$

Fazendo o produto escalar de ambos os membros da equação (1) com v_i, $i = 1, ..., k$ e aplicando as propriedades do produto escalar, obtemos

$$x_1(v_1 \cdot v_i) + ... + x_i(v_i \cdot v_i) + ... + x_k(v_k \cdot v_i) = 0. \tag{2}$$

Por hipótese temos $v_i v_j = 0$, se $i \neq j$. Assim de (2) obtemos que

$$x_i \|v_i\|^2 = 0.$$

Como $v_i \neq \overline{0}$, então $\|v_i\| \neq 0$, logo $x_i = 0$ para todo $i = 1, \ldots, k$.

2.3 Bases Ortogonais e Ortonormais

Definição 2.6 *Seja $\{v_1, ..., v_k\}$ uma base de um subespaço de $\mathbb{R}^n$.*

(a) Dizemos que $\{v_1, ..., v_k\}$ é uma ***base ortogonal****, se $\langle v_i, v_j \rangle = 0$, para $i \neq j$, ou seja, se quaisquer dois vetores da base são ortogonais.*

(b) Dizemos que $\{v_1, ..., v_k\}$ é uma ***base ortonormal****, se além de ser uma base ortogonal, $\|v_i\| = 1$, ou seja, o vetor v_i é* ***unitário****, para $i = 1, ..., k$.*

Exemplo 2.3 *A* ***base canônica*** *do $\mathbb{R}^n$, que é formada pelos vetores $e_1 = (1, 0, ..., 0), e_2 = (0, 1, 0, .., 0), ..., e_n = (0, 0, 0, \ldots, 1)$ é uma base ortonormal do $\mathbb{R}^n$.*

Exemplo 2.4 *A base $V \in \mathbb{R}^3$, composta pelos vetores $v_1 = (-1, 2, 1)$, $v_2 = (2, 1, 0)$ e $v_3 = (-1, 2, -5)$, é uma base ortogonal mas não é ortonormal.*

Podemos verificar tal afirmação fazendo o produto interno entre os vetores.

$$v_1 \cdot v_2 = (-1, 2, 1) \cdot (2, 1, 0) = -2 + 2 + 0 = 0$$
$$v_1 \cdot v_3 = (-1, 2, 1) \cdot (-1, 2, -5) = 1 + 4 - 5 = 0$$
$$v_2 \cdot v_3 = (2, 1, 0) \cdot (-1, 2, -5) = -2 + 2 + 0 = 0$$

$$v_1 \cdot v_1 = (-1, 2, 1) \cdot (-1, 2, 1) = 1 + 4 + 1 = 6 \neq 1$$
$$v_2 \cdot v_2 = (2, 1, 0) \cdot (2, 1, 0) = 4 + 1 + 0 = 5 \neq 1$$
$$v_3 \cdot v_3 = (-1, 2, -5) \cdot (-1, 2, -5) = 1 + 4 + 25 = 30 \neq 1.$$

Caso desejemos transformar a base do exemplo anterior em ortonormal, basta multiplicar cada vetor pelo inverso de sua norma.

$$u_1 = \frac{v_1}{\| v_1 \|} = \frac{(-1,2,1)}{\sqrt{(-1)^2+(2)^2+(1)^2}} = \frac{(-1,2,1)}{\sqrt{6}} = \left(-\frac{\sqrt{6}}{6}, \frac{\sqrt{6}}{3}, \frac{\sqrt{6}}{6}\right)$$

$$u_2 = \frac{v_2}{\| v_2 \|} = \frac{(2,1,0)}{\sqrt{(2)^2+(1)^2+(0)^2}} = \frac{(2,1,0)}{\sqrt{5}} = \left(\frac{2\sqrt{5}}{5}, \frac{\sqrt{5}}{5}, 0\right)$$

$$u_3 = \frac{v_3}{\| v_3 \|} = \frac{(-1,2,-5)}{\sqrt{(-1)^2+(2)^2+(-5)^2}} = \frac{(-1,2,-5)}{\sqrt{30}} = \left(-\frac{\sqrt{30}}{30}, \frac{\sqrt{30}}{15}, -\frac{\sqrt{30}}{6}\right).$$

2.4 Transformação Linear

Um importante conceito em Álgebra Linear é o de transformação linear, com aplicações em diversas áreas do conhecimento. Iremos apenas enunicá-lo e usar para falar sobre mudança de bases.

Definição 2.7 *Dados dois espaços vetoriais $\mathbb{V}$ e $\mathbb{W}$, uma transformação (ou função) entre esses espaços é uma lei que associa a todo vetor $v \in \mathbb{V}$, um único vetor $w \in \mathbb{W}$, tal que $w = T(v)$. O vetor $T(v) \in \mathbb{W}$ é chamado imagem de v pela transformação T.*

Definição 2.8 *Sejam $\mathbb{V}$ e $\mathbb{W}$ dois espaços vetoriais. A aplicação $T : \mathbb{V} \to \mathbb{W}$ é uma transformação linear se:*

i) $T(u+v) = T(u) + T(v), \forall\ u, v \in \mathbb{V}$,

ii) $T(\alpha v) = \alpha T(v), \forall\ v \in \mathbb{V},\ \forall\ \alpha \in \mathbb{R}$.

Chamamos essas duas propriedades de linearidade da transformação T.

Obs: A propriedade da linearidade pode também ser testada em uma única análise, a saber T é linear se $T(\alpha u + v) = \alpha T(u) + T(v)$, $\forall u, v \in \mathbb{V}$, $\forall \alpha \in \mathbb{R}$.

Exemplo 2.5 *$T : \mathbb{R}^2 \to \mathbb{R}^3$, dada por $T(x,y) = (x+y, x, x-y)$ é uma transformação linear, pois dados $u = (x_1, y_1), v = (x_2, y_2) \in \mathbb{R}^2$, temos $(u+v) = (x_1 + x_2, y_1 + y_2)$ e $\alpha u = (\alpha x_1, \alpha y_1)$, logo:*

$$\begin{aligned} T(u+v) &= (x_1 + x_2 + y_1 + y_2, x_1 + x_2, x_1 + x_2 - y_1 - y_2) \\ &= (x_1 + y_1, x_1, x_1 - y_1) + (x_2 + y_2, x_2, x_2 - y_2) \\ &= T(u) + T(v), \end{aligned}$$

$$\begin{aligned} T(\alpha u) &= (\alpha x_1 + \alpha y_1, \alpha x_1, \alpha x_1 - \alpha y_1) \\ &= \alpha(x_1 + y_1, x_1, x_1 - y_1) \\ &= \alpha T(u). \end{aligned}$$

Exemplo 2.6 *$T : \mathbb{R} \to \mathbb{R}$, dada por $T(x) = 2x + 1$ não é uma transformação linear. De fato, para $u = (x_1), v = (x_2) \in \mathbb{R}$, temos:*

$$\begin{aligned} T(u+v) = T(x_1 + x_2) &= 2(x_1 + x_2) + 1 \\ &\neq (2x_1 + 1) + (2x_2 + 1) \\ &= T(u) + T(v). \end{aligned}$$

Exemplo 2.7 *$T : \mathbb{R}^2 \to \mathbb{R}^3$, definida por $T(v) = Av$ (ou $T_A(v) = Av$), com $A_{3,2}$ uma matriz qualquer, é uma transformação linear, pois, da álgebra matricial:*

$$T_A(\alpha u + v) = A(\alpha u + v) = A(\alpha u) + A(v) = \alpha A(u) + A(v) = \alpha T_A(u) + T_A(v).$$

Em um caso particular, podemos ter a matriz

$$A = \begin{bmatrix} 0 & 1 \\ -1 & 4 \\ 2 & 3 \end{bmatrix},$$

e então

$$T_A(v) = \begin{bmatrix} 0 & 1 \\ -1 & 4 \\ 2 & 3 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \end{bmatrix} = (y, -x + 4y, 2x + 3y)^T.$$

De fato, é possível mostrar que para toda transformação linear entre espaços vetoriais $T : \mathbb{V} \to \mathbb{W}$, com $\dim \mathbb{V} = m$ e $\dim \mathbb{W} = n$, existe uma matriz $A_{n,m}$ que representa tal transformação nas respectivas bases. Reciprocamente, toda matriz $A_{n,m}$ define uma transformação linear $T : \mathbb{V} \to \mathbb{W}$, com $\dim \mathbb{V} = m$ e $\dim \mathbb{W} = n$.

2.5 Mudança de Base

Sejam A e B bases de um espaço vetorial $\mathbb{V}$. Chamamos matriz de mudança da base A para a base B, $[I]_B^A$, como uma matriz que expressa as coordenadas de um vetor v em relação a base A numa outra base qualquer B.

Por exemplo, consideremos o caso[3] em que $\dim \mathbb{V} = 3$. Sejam $A = \{v_1, v_2, v_3\}$ e $B = \{w_1, w_2, w_3\}$ duas bases arbitrárias de $\mathbb{V}$. Dado um vetor $v \in \mathbb{V}$, este pode ser expresso como combinação linear dos vetores das bases A e B. Comecemos escrevendo v em termos da base A

$$v = x_1 v_1 + x_2 v_2 + x_3 v_3, \tag{3}$$

ou simplesmente

$$v_A = (x_1, x_2, x_3).$$

Já para a base B temos

$$v = y_1 w_1 + y_2 w_2 + y_3 w_3, \tag{4}$$

[3]O problema para os espaços de dimensão finita n qualquer é análogo.

ou então

$$v_B = (y_1, y_2, y_3).$$

Por sua vez os vetores da base A podem ser escritos em relação à base B, isto é

$$\begin{aligned} v_1 &= a_{11}w_1 + a_{21}w_2 + a_{31}w_3, \\ v_2 &= a_{12}w_1 + a_{22}w_2 + a_{32}w_3, \\ v_3 &= a_{13}w_1 + a_{23}w_2 + a_{33}w_3, \end{aligned} \tag{5}$$

Substituindo (5) em (3), obtemos

$$v = x_1(a_{11}w_1+a_{21}w_2+a_{31}w_3)+x_2(a_{12}w_1+a_{22}w_2+a_{32}w_3)+x_3(a_{13}w_1+a_{23}w_2+a_{33}w_3). \tag{6}$$

Reorganizando os termos, podemos reescrever a equação acima como

$$v = (a_{11}x_1+a_{12}x_2+a_{13}x_3)w_1+(a_{21}x_1+a_{22}x_2+a_{23}x_3)w_2+(a_{31}x_1+a_{32}x_2+a_{33}x_3)w_3, \tag{7}$$

e comparando (7) com (4), segue

$$y_1 = a_{11}x_1 + a_{12}x_2 + a_{13}x_3,$$

$$y_2 = a_{21}x_1 + a_{22}x_2 + a_{23}x_3,$$

$$y_3 = a_{31}x_1 + a_{32}x_2 + a_{33}x_3,$$

que na forma matricial pode ser reescrito como

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix},$$

ou, mais simplesmente, pela equação:

$$[v]_B = [I]_B^A \, [v]_A.$$

Assim sendo, a matriz

$$[I]_B^A = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix},$$

é chamada **matriz mudança de base**, que permite escrever um vetor v qualquer, cujas componentes na base A sejam conhecidas, em termos de suas componentes da base B. Para a mudança de base de B para A, temos que

$$[I]_A^B = \left([I]_B^A\right)^{-1}.$$

Não é nosso objetivo nesse texto, mas é possível mostrar que uma matriz mudança de base sempre tem inversa.

3 Séries de Fourier

Nesta seção serão definidos itens de extrema importância, mostrando a a influência direta de Álgebra linear na transformada discreta do cosseno (DCT), para maiores informações veja [20], [16], [17], [18] e [19]. No entanto, devemos começar definindo a série de Fourier, e para falarmos sobre ela devemos entender conceitos básicos como funções, funções trigonométricas e principalmente as funções periódicas.

3.1 Funções Periódicas

Sabe-se que as funções periódicas são todas aquelas cujos os valores da função $y = f(x)$ se repetem para determinados valores da variável x, sendo assim, para cada período definido pelos valores de x, sempre teremos os valores repetidos para aquela função.

Definição 3.1 *Uma função $f : \mathbb{R} \longrightarrow \mathbb{R}$ é **periódica** se existe $T \in \mathbb{R}$, $T \neq 0$, tal que $f(x+T) = f(x)$, $\forall x \in \mathbb{R}$. O número real T é chamado um **período** para a função f. Chamamos de **período fundamental** o menor período positivo da função f.*

Claramente, se T é um período para a função f, então qualquer múltiplo inteiro de T também é um período para f : $2T, -2T, 3T, -3T, 4T, -4T$, etc. O menor valor positivo do período é chamado de período fundamental de $f(x)$. Também, a função constante é considerada periódica com qualquer período e sem período fundamental. Em geral o período fundamental de uma função periódica é referido simplesmente como *período* da função. Por que o valor de uma função periódica repete-se a cada intervalo de comprimento igual ao seu período, para conhecer uma função periódica de período T basta descrevê-la em qualquer intervalo de comprimento T; o seu gráfico é obtido repetindo-se o gráfico neste intervalo em qualquer outro intervalo de comprimento T.

Exemplo 3.1 *As funções seno e cosseno são periódicas e ambas têm período fundamental 2π. Quando tratamos das propriedades dessas funções, queremos referenciar o domínio, imagem e periodicidade das funções. No caso das funções seno e cosseno temos coincidência nas propriedades. Sobre o domínio, ambas estão definidas para qualquer valor real, sendo assim* $\mathrm{dom}(\mathrm{sen}) = \mathrm{dom}(\cos) = \mathbb{R}$*, sobre a imagem, ambas funções estão compreendidas com valores reais entre 1 e -1,* $\mathrm{Im}(\mathrm{sen}) = \mathrm{Im}(\cos) = I = \{y \in \mathbb{R} | -1 \leq y \leq 1\}$ *e ao tratarmos da periodicidade temos que ambas possuem o mesmo período de 2π para todo $x \in \mathbb{R}$ e $k \in \mathbb{Z}$.*

$$\mathrm{sen}(x) = \mathrm{sen}(x + 2\pi) = \mathrm{sen}(x + 4\pi) = ... = \mathrm{sen}(x + 2k\pi)$$

$$\cos(x) = \cos(x + 2\pi) = \cos(x + 4\pi) = ... = \cos(x + 2k\pi)$$

Exemplo 3.2 *As funções* $\mathrm{sen}(\frac{m\pi}{L}x)$ *e* $\cos(\frac{m\pi}{L}x)$ *para $m = 1, 2, 3, ...$ são periódicas com período fundamental $T = \frac{2L}{m}$.*

Para verificar essa característica lembramos que $\operatorname{sen} x$ e $\cos x$ tem como período fundamental 2π e que $\operatorname{sen} \alpha x$ e $\cos \alpha x$ tem como período fundamental $\frac{2\pi}{\alpha}$. Assim escolhendo $\alpha = \frac{m\pi}{L}$ verificamos que o período fundamental T de $\operatorname{sen}(\frac{m\pi}{L}x)$ e $\cos(\frac{m\pi}{L}x)$ é dado por $T = 2\pi(\frac{m\pi}{L})^{-1} = \frac{2L}{m}$. Adicionalmente, como todo múltiplo inteiro de um período também é um período então cada uma das funções $\operatorname{sen}(\frac{m\pi}{L}x)$ e $\cos(\frac{m\pi}{L}x)$ tem um período comum $2L$.

3.2 Séries de Fourier

A série de Fourier apareceu nos estudos de Joseph Fourier a respeito do calor. A solução da equação diferencial parcial (EDP) do calor naturalmente leva a soluções que são séries de senos e cossenos, que serão descritas na equação (8) a seguir. O leitor interessado em tais soluções pode consultar [6]. Enfatizamos que não iremos utilizar diretamente a série de Fourier no processo de compressão de imagens, mas sim a transformada discreta do cosseno que será vista a seguir. Entretanto como ela é a primeira a ser estuda em cursos básicos de EDP's. Consideremos agora uma função contínua[4] e de período fndamental $2L$. Tomando a representação da função no intervalo $[-L, L]$, a série de Forier de f é

$$f(x) = \frac{a_0}{2} + \sum_{m=1}^{\infty} \left[a_n \cos\left(\frac{n\pi x}{L}\right) + b_n \operatorname{sen}\left(\frac{n\pi x}{L}\right) \right], \tag{8}$$

onde os coeficientes podem ser calculados por

$$\begin{aligned} a_n &= \frac{1}{L}\int_{-L}^{+L} f(x)\cos\left(\frac{n\pi x}{L}\right)\,dx, \quad n = 0, 1, 2, \ldots \\ b_n &= \frac{1}{L}\int_{-L}^{+L} f(x)\operatorname{sen}\left(\frac{n\pi x}{L}\right)\,dx, \quad n = 1, 2, 3, \ldots \end{aligned} \tag{9}$$

É possível escrever a transformada de Fourier em sua forma complexa usando a identidade de Euler

$$e^{\pm i\theta} = \cos\theta \pm i \operatorname{sen}\theta,$$

onde $i^2 = -1$ é o número imaginário. Assim temos que (8) pode ser reescrito como

$$f(x) = \sum_{n=-\infty}^{+\infty} c_n \exp\left(i\frac{n\pi x}{L}\right), \tag{10}$$

onde os coeficientes agora são dados por

$$c_n = \frac{1}{2L}\int_{-L}^{+L} f(x)\exp\left(-i\frac{n\pi x}{L}\right)\,dx, \quad n \in \mathbb{Z}. \tag{11}$$

As expressões (8)-(9) e (10)-(11) são equivalentes, porém apresentam algumas diferenças: se f for uma função real, a série de Fourier (8) apresenta coeficientes a_n e b_n reais, o que é

[4]Essa hipótese é muito forte pode ser relaxada.

mais intuitivo a primeira vista. Por outro lado a série de Fourier complexa (10) apresenta uma expressão mais simétrica para seus coeficientes (11). Nosso interesse, no entanto, é que a serie de Fourier complexa nos permite uma extensão natural para a transformada de Fourier, como veremos na seção 3.4.

3.3 Série de Fourier e Álgebra Linear

Sejam u e v funções reais contínuas em um intervalo $\alpha < x < \beta$, O conjunto de tais funções forma um espaço vetorial. Ver exemplo 1 no capítulo 1 em [3]. O produto interno canônico $\langle u, v\rangle$ é definido como:

$$\langle u, v\rangle = \int_\alpha^\beta u(x)v(x)dx. \tag{12}$$

Podemos demonstrar que o produto interno definido pela equação (12) satisfaz às propriedades mostradas na seção 2.2. Com efeito, para a primeira prorpiedade temos

$$\langle u, v\rangle = \int_\alpha^\beta u(x)v(x)dx = \int_\alpha^\beta v(x)u(x)dx = \langle v, u\rangle.$$

Já para a segunda propriedade, segue

$$\langle v, v\rangle = \int_\alpha^\beta v(x)v(x)dx = \int_\alpha^\beta [v(x)]^2dx \geq 0,$$

por tratar-se de um integrando não negativo. Podemos ver que a igualdade só é válida quando v é a função identicamente nula. Continuando, vemos que as propriedades de linearidade saem diretamente do fato da integral ser linear, isto é

$$\langle u+v, w\rangle = \int_\alpha^\beta (u(x)+v(x))w(x)dx = \int_\alpha^\beta u(x)w(x)dx + \int_\alpha^\beta v(x)w(x)dx = \langle u, w\rangle + \langle v, w\rangle,$$

e

$$\langle \lambda u, v\rangle = \int_\alpha^\beta (\lambda u(x))v(x)dx = \lambda \int_\alpha^\beta u(x)v(x)dx = \lambda\langle u, v\rangle.$$

Sendo (12) um produto interno, as funções u e v são chamadas de ortogonais no intervalo $\alpha < x < \beta$ caso

$$\langle u, v\rangle = \int_\alpha^\beta u(x)v(x)dx = 0. \tag{13}$$

3.3.1 Relações de Ortogonalidade

Inicialmente, com base em trigonometria básica, vamos relembrar as formulas de seno e cosseno da soma e da diferença.

$$\operatorname{sen}(\alpha + \beta) = \operatorname{sen}\alpha\cos\beta + \cos\alpha\operatorname{sen}\beta, \tag{14a}$$
$$\operatorname{sen}(\alpha - \beta) = \operatorname{sen}\alpha\cos\beta - \cos\alpha\operatorname{sen}\beta, \tag{14b}$$
$$\cos(\alpha + \beta) = \cos\alpha\cos\beta - \operatorname{sen}\alpha\operatorname{sen}\beta, \tag{14c}$$
$$\cos(\alpha - \beta) = \cos\alpha\cos\beta + \operatorname{sen}\alpha\operatorname{sen}\beta. \tag{14d}$$

Usando essas fórmulas é possível obter as identidades a seguir, as quais terão aplicação mais adiante, somando (14c) com (14d) temos (15a), fazendo também a subtração (14a) - (14b) temos (15b) e por fim subtraindo (14d) - (14c) temos (15c). Veja os resultados a seguir,

$$2\cos\alpha\cos\beta = \cos(\alpha+\beta) + \cos(\alpha-\beta), \tag{15a}$$
$$2\cos\alpha\,\mathrm{sen}\,\beta = \mathrm{sen}(\alpha+\beta) - \mathrm{sen}(\alpha-\beta), \tag{15b}$$
$$2\,\mathrm{sen}\,\alpha\,\mathrm{sen}\,\beta = \cos(\alpha-\beta) - \cos(\alpha+\beta). \tag{15c}$$

Fazendo $\alpha = \left(\frac{m\pi}{L}x\right)$ e $\beta = \left(\frac{n\pi}{L}x\right)$, considerando $m \neq n$, temos:

$$\cos\left(\frac{m\pi}{L}x\right)\cos\left(\frac{n\pi}{L}x\right) = \frac{1}{2}\left\{\cos\left[\frac{(m+n)\pi}{L}x\right] + \cos\left[\frac{(m-n)\pi}{L}x\right]\right\}, \tag{16}$$

$$\cos\left(\frac{m\pi}{L}x\right)\mathrm{sen}\left(\frac{n\pi}{L}x\right) = \frac{1}{2}\left\{\mathrm{sen}\left[\frac{(m+n)\pi}{L}x\right] + \mathrm{sen}\left[\frac{(m-n)\pi}{L}x\right]\right\}, \tag{17}$$

$$\mathrm{sen}\left(\frac{m\pi}{L}x\right)\mathrm{sen}\left(\frac{n\pi}{L}x\right) = \frac{1}{2}\left\{\cos\left[\frac{(m-n)\pi}{L}x\right] - \cos\left[\frac{(m+n)\pi}{L}x\right]\right\}. \tag{18}$$

No caso particular em que $m = n$ as relações anteriores (16), (17) e (18), reduzem-se às seguintes relações:

$$\cos^2\left(\frac{m\pi}{L}x\right) = \frac{1}{2}\left\{1 + \cos\left[\frac{2m\pi}{L}x\right]\right\}, \tag{19}$$

$$\cos\left(\frac{m\pi}{L}x\right)\mathrm{sen}\left(\frac{m\pi}{L}x\right) = \frac{1}{2}\,\mathrm{sen}\left[\frac{2m\pi}{L}x\right], \tag{20}$$

$$\mathrm{sen}^2\left(\frac{m\pi}{L}x\right) = \frac{1}{2}\left\{1 - \cos\left[\frac{2m\pi}{L}x\right]\right\}. \tag{21}$$

Tendo isso, podemos enunciar e demonstrar o teorema a seguir:

Teorema 3.1 *(Relações de Ortogonalidade), com* $m, n \in \mathbb{Z}_+^*$

$$\int_{-L}^{L}\cos\left(\frac{m\pi}{L}x\right)\cos\left(\frac{n\pi}{L}x\right)dx = \begin{cases} 0 & se \quad m \neq n \\ L & se \quad m = n \end{cases} \tag{22}$$

$$\int_{-L}^{L}\cos\left(\frac{m\pi}{L}x\right)\mathrm{sen}\left(\frac{n\pi}{L}x\right)dx = \{\ 0 \quad \forall \ \ m \ e \ n \tag{23}$$

$$\int_{-L}^{L}\mathrm{sen}\left(\frac{m\pi}{L}x\right)\mathrm{sen}\left(\frac{n\pi}{L}x\right)dx = \begin{cases} 0 & se \quad m \neq n \\ L & se \quad m = n \end{cases} \tag{24}$$

Os resultados acima são chamados de "Relações de Ortogonalidade".

Um conjunto de funções formam um conjunto ortogonal se cada par de funções diferentes pertencentes ao conjunto é ortogonal. Assim, as funções $\text{sen}(\frac{m\pi}{L}x)$ e $\cos(\frac{m\pi}{L}x)$ para $m = 1, 2, 3, \ldots$ formam um conjunto ortogonal de funções no intervalo $-L < x < L$. Estas relações acimas podem ser provadas facilmente a partir da integração direta, usando as relações trigonométricas (19), (20) e (21) .

Demonstração: A prova da relação (22) será apresentada a seguir, pois nosso intuito é falar sobre a transformada discreta do cosseno. As demonstrações das demais relações são análogas, por exemplo consulte [17]. Se $m \neq n$ temos o seguinte:

$$P = \int_{-L}^{L} \cos\left(\frac{m\pi}{L}x\right)\cos\left(\frac{n\pi}{L}x\right)dx = \frac{1}{2}\left\{\int_{-L}^{L}\left[\cos\left[\frac{(m+n)\pi}{L}x\right] + \cos\left[\frac{(m-n)\pi}{L}x\right]\right]dx\right\},$$

$$P = \frac{1}{2}\left[\frac{L}{\pi(m+n)}\,\text{sen}\left[\frac{(m+n)\pi}{L}x\right] + \frac{L}{\pi(m-n)}\,\text{sen}\left[\frac{(m-n)\pi}{L}x\right]\right]_{-L}^{L},$$

$$P = \frac{L}{2\pi(m+n)}\left[\text{sen}\left[\frac{(m+n)\pi}{L}x\right]\right]_{-L}^{L} + \frac{L}{2\pi(m-n)}\left[\text{sen}\left[\frac{(m-n)\pi}{L}x\right]\right]_{-L}^{L},$$

$$P = \frac{L}{2\pi}\left\{\frac{1}{(m+n)}[\text{sen}(m+n)\pi - \text{sen}(-1)(m+n)\pi] + \frac{1}{(m-n)}[\text{sen}(m-n)\pi - \text{sen}(-1)(m-n)\pi]\right\},$$

$$P = \frac{L}{\pi}\left[\frac{1}{(m+n)}\,\text{sen}(m+n)\pi + \frac{1}{(m-n)}\,\text{sen}(m-n)\pi\right] = 0.$$

□

Lembrando que os múltiplos de π são tais que $\text{sen}(k\pi) = 0$ para $k \in \mathbb{Z}$. Para $m = n$, temos:

$$P = \int_{-L}^{L} \cos^2\left(\frac{m\pi}{L}x\right)dx = \frac{1}{2}\left\{\int_{-L}^{L}\left[1 + \cos\left(\frac{2m\pi}{L}x\right)\right]dx\right\},$$

$$P = \frac{1}{2}\left[x + \text{sen}\left(\frac{2m\pi}{L}x\right)\frac{L}{2m\pi}\right]_{-L}^{L},$$

$$P = \frac{1}{2}\left[L - (-L) + \frac{L}{2m\pi}\,\text{sen}(2mr) - \frac{L}{2m\pi}\,\text{sen}[(-1)(2m\pi)]\right],$$

$$P = L + \frac{L}{2m\pi}\,\text{sen}(2m\pi) = L.$$

□

O nome *relações de ortogonalidade* deve-se ao fato de que as expressões citadas acima significam que as funções $\text{sen}(n\pi x/L)$ e $\cos(n\pi x/L)$ são ortogonais no espaço vetorial das funções quadrado-integráveis definidas no intervalo $[-L, L]$. De fato, no espaço

$$L^2([a,b]) = \left\{u : [a,b] \to \mathbb{R} : \int_a^b u^2(x)dx < \infty\right\},$$

das funções definidas no intervalo $[a, b]$ cujo quadrado é integrável, podemos definir o produto interno por

$$\langle u, v \rangle = \int_a^b u(x)v(x)dx.$$

Por serem as funções quadrado-integráveis, a integral acima está bem definida.

3.4 Transformada de Fourier

A equação (10) tem sua aplicação voltada a representação de funções periódicas de período 2L, já quando as funções são não periódicas é necessário calcular o limite tendendo ao infinito que acaba gerando a Transformada de Fourier, não iremos mostrar o desenvolvimento desse limite e podendo ser visto em [6]. A transformada de Fourier de uma função $f : \mathbb{R} \longrightarrow \mathbb{C}$ é denotada pelo simbolo $\hat{f}$ e tem a seguinte definição:

$$\hat{f}(\xi) = \frac{1}{2\pi} \int_{-\infty}^{\infty} e^{-ix\xi} f(x)dx. \tag{25}$$

com $\xi \in \mathbb{R}$. Logo temos que se $f : \mathbb{R} \longrightarrow \mathbb{R}$ e usando a fórmula de Euler, temos:

$$\hat{f}(\xi) = \frac{1}{2\pi} \left(\int_{-\infty}^{\infty} \cos(\xi x) f(x)dx - i \int_{-\infty}^{\infty} \text{sen}(\xi x) f(x)dx \right). \tag{26}$$

Já a transformada inversa é dada pela equação:

$$f(x) = \int_{-\infty}^{\infty} \hat{f}(\xi) e^{2\pi i \xi x} dx. \tag{27}$$

A identidade de Euler, uma exponencial complexa, pode ser vista na Transformada Discreta de Fourier onde seus coeficientes são, quase sempre, números complexos. Por outro lado, os coeficientes da DCT (discrete cosine transform serão mostrados no seção seguinte) são sempre números reais.

3.4.1 Transformada discreta de Fourier

Se um sinal for considerado discreto e sendo ele digital, aplicamos a transformada discreta de Fourier (DFT, do inglês *Discrete Fourier Transform*), assim como as outras analises de Fourier também é baseada na decomposição de sinais em senóides, é utilizada para sinais digitalizados e com grande aplicação na área computacional em processamento digital de sinais, pois trabalha com sinais discretos e periódicos assim como os computadores. Lembrando que Fourier sempre considera os sinais tem duração infinita, mas ao trabalhar com um computador os sinais devem ter necessariamente duração finita. Um modelo da transformada para funções discretas é dado pela expressão abaixo, na ordem é apresentado a transformada e sua inversa:

- DFT unidimensional

$$F_k = \frac{1}{\sqrt{n}} \sum_{k=0}^{n-1} f_j e^{-\frac{2\pi i}{n} jk} \qquad k = 0, ..., n-1 \tag{28}$$

$$f_j = \frac{1}{\sqrt{n}} \sum_{j=0}^{n-1} F_k e^{\frac{2\pi i}{n} jk} \qquad j = 0, ..., n-1 \tag{29}$$

- DFT bidimensional

$$F_{(w_x,w_y)} = \frac{1}{\sqrt{nm}} \sum_{x=0}^{n-1} \sum_{y=0}^{m-1} f_{(x,y)} e^{-2i\pi(\frac{w_x x}{n} + \frac{w_y y}{m})} \tag{30}$$

Para $w_x = 1, 2, \ldots, n-1$ e $w_y = 1, 2, \ldots, m-1$

$$f_{(x,y)} = \frac{1}{\sqrt{nm}} \sum_{w_x=0}^{n-1} \sum_{w_y=0}^{m-1} F_{(w_x,w_y)} e^{2i\pi(\frac{w_x x}{n} + \frac{w_y y}{m})} \tag{31}$$

Para $x = 1, 2, \ldots, n-1$ e $y = 1, 2, \ldots, m-1$

3.5 Transformada discreta do cosseno (DCT)

Nessa seção serão apresentados duas formas de Transformada discreta do Cosseno (Ou DCT do inglês *Discrete Cosine Transform*), DCT 1D e DCT 2D, contudo daremos mais enfase a DCT bidimensional, a qual tem maior importância nesse trabalho. A DCT tem sua aplicação voltada a compressão de dados e processamento digital de imagens, assim como em outras transformadas ela tem por objetivo descorrelacionar os dados da imagem ou video que esta sendo tratado. As imagens que tiveram seus dados transformados podem ser facilmente restauradas ao seu formato original sem perdas. Esta transformada tem seu funcionamento baseado na escolha de uma base onde seus primeiros elementos tem baixa frequência de variação, ou seja um bloco separado de uma imagem natural onde os valores de cada pixel são próximos, e os últimos elementos tem alta frequência de variação. O primeiro valor transformado na sequência ou na matriz é conhecido como valor médio, referimos a ele como coeficiente DC, os demais valores serão chamados de AC e sua maioria é zero e guarda as variações do valor médio. Com referência na Álgebra linear, o que acontece quando usamos a transformada é uma mudança de base, da base canônica para uma base ortonormal e os coeficientes transformados são vistos como a discretização do cosseno de uma determinada frequência. Os processos descritos aqui serão explicados a seguir parte a parte com mais detalhes para maior compreensão.

3.5.1 DCT 1D

A DCT unidimensional é aplicada a vetores em uma dimensão

$$c(u) = \alpha(u) \sum_{x=0}^{N-1} f(x) \cos\left[\frac{\pi(2x+1)u}{2N}\right] \tag{32}$$

Para $x = 0, 1, 2, ..., N-1$.
E a DCT-1D inversa é dada pela equação:

$$f(x) = \sum_{u=0}^{N-1} \alpha(u)c(u) \cos\left[\frac{\pi(2x+1)u}{2N}\right] \tag{33}$$

Para $u = 0, 1, 2, ..., N-1$.
Para ambas equações $\alpha(u)$ será dado como:

$$\alpha(u) = \begin{cases} \frac{1}{\sqrt{N}} & \text{para} \quad u = 0 \\ \sqrt{\frac{2}{N}} & \text{para} \quad u \neq 0 \end{cases}$$

3.5.2 DCT 2D

Como o intuito desse trabalho é a compressão de imagens, temos que a DCT 2D quem apresenta mais eficácia no processo de compactação de energia, a DCT bidimensional é basicamente uma extensão da DCT unidimensional e será dada pela seguinte forma, com $N \in \mathbb{N}$:

$$C(i,j) = \alpha(i)\alpha(j) \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(x,y) \cos\left(\frac{(2x+1)i\pi}{2N}\right) \cos\left(\frac{(2y+1)j\pi}{2N}\right) \tag{34}$$

Para $i, j = 0, 1, ..., N-1$, e

$$\alpha(x) = \begin{cases} \frac{1}{\sqrt{N}} & \text{se} \quad x = 0 \\ \sqrt{\frac{2}{N}} & \text{se} \quad x = 1, 2, ..., N-1 \end{cases}.$$

Onde $C(i, J)$ é a Transformada Discreta do Cosseno, $f(x, y)$ são os valores da imagem original em escalas de cinza e N é a dimensão da imagem (matriz ou bloco).

Com o intuito de maior eficácia da aplicação da DCT nos cálculos é necessário dividir a imagem em matrizes quadradas (blocos), podem ter ordem 4, 8, 16, 32 e até 64. Normalmente são usados blocos de ordem 8 pela eficiência. A inversa dessa transformada é dada por:

$$f(x,y) = \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} \alpha(i)\alpha(j)c(i,j) \cos\left(\frac{(2x+1)i\pi}{2N}\right) \cos\left(\frac{(2y+1)j\pi}{2N}\right). \tag{35}$$

Até aqui foi introduzido a parte de Matemática envolvida que será aplicada no processo, na próxima seção será abordado um pouco de noções básicas de programação que também são necessárias na compressão, lembrando que ao falamos de programação falamos também de matemática.

4 Noções de Programação

4.1 JPEG

O padrão JPEG é um método de compressão muito usado e também é considerado como um formato de arquivo. Esse método foi desenvolvido a quase duas décadas pelo grupo *Joint Photographics Experts Group*, cujo acrônimo cria sua sigla. Nos dias atuais, ainda é considerado um dos melhores métodos de compressão pois, consegue juntar redução de dados no armazenamento com qualidade de imagem satisfatória mesmo com a perdas no processo. Para uma leitura complementar com mais detalhes veja [13], [9].

4.1.1 Características

O tamanho do arquivo final pode ser manipulado, quanto maior a compressão menor será o arquivo a ser armazenado e de forma proporcional será a qualidade final. Possui maior aplicação em imagens de tons contínuos, imagens naturais com muitas cores repetidas e sem grande variação de uma cor para outra, onde seu desempenho é ótimo. No caso de imagens com baixos níveis de cor, imagens monocromáticas ou escritas, sua aplicação deixa de ser efetiva, pois torna visível a distorção da imagem mesmo em baixos índices de compressão.

O formato JPEG é considerado uma das técnicas de compressão com perdas pois ele é baseado na transformada discreta do cosseno, porém essas perdas se tornam mais aceitáveis já que são imperceptíveis e há um grande benefício no armazenamento.

O algoritmo para compressão no formato JPEG explora e força a ocorrência de redundâncias entre os pixels (dados repetidos), que é de onde vem seu sucesso pois as variações de cor são menos perceptíveis do que as variações de brilho aos olhos humanos. Nas imagens naturais e de cores continuas as perdas não são notadas facilmente se as taxas de compressão estiverem entre 10:1 e 20:1, se as taxas passarem disso poderão ser notadas pequenas distorções, imagens monocromáticas as distorções são notadas com taxas de compressão acima de 5:1, é onde não recomenda se aplicar esse método.

4.2 Codificação de Huffman

A codificação de huffman foi desenvolvida na década de 50 quando o seu criador, David Huffman, ainda cursava o doutorado no MIT (*Massachusetts Institute of Technology*). Esse modelo de codificação usa modelos estatísticos, pela frequência ou pela probabilidade de ocorrência, e permite criar diferentes códigos binários para cada pixel. Cada pixel recebe um código de acordo com o número de vezes que aparece, se aparece várias vezes terá um código menor e se aparece pouco seu código será maior, dessa maneira a codificação final fica menor que uma codificação onde todos os elementos tem a mesma dimensão. Codificação de tamanho variável, (em inglês VLC, para *variable length code*), é a nomeação que esse código recebe. Ver maiores detalhes em [7] e [8].

4.2.1 Árvore de codificação

A codificação de Huffman cria uma árvore onde os elementos aparecem ordenados de forma decrescente por ordem de frequência ou probabilidade de ocorrência, e suas ramificações surgem a partir da ligação dos caracteres sempre em ordem crescente de frequência, onde são ligados dois a dois e tem suas frequências somadas criando no final um ponto com a frequência acumulada . Com a árvore pronta, cada ramificação recebe um código, a esquerda recebe o código 0, o da direita recebe 1, dessa forma até estar toda completa, levando em consideração que cada ramo se torna um novo código e acaba quando junta todos os caminhos em um só ponto que chamaremos de raiz. Sendo assim, cada caractere com maior frequência recebe uma codificação menor e na medida que a frequência diminui vai aumentando a codificação. Quando falamos de codificação normalmente é lembrando da codificação binaria fixa onde cada caractere recebe uma codificação de 8 bits, com a codificação de Huffman, que é uma codificação de tamanho variável, cada caractere recebe uma codificação de tamanho diferente de acordo com sua frequência. Observe o exemplo a seguir:

Exemplo 4.1 *Vejamos a frase de Pitágoras: "**Educai as crianças para que não seja necessário punir os adultos**".*

Antes de tudo serão definidos alguns detalhes só para facilitar a compreensão: Veremos todas as letras como se fossem maiúsculas, os acentos não as tornam diferentes umas das outras, os espaços serão diferenciados com um traço.

Observe que o exemplo será feito por frequência, podendo ser feito em outros casos por probabilidade de ocorrência – o processo será o mesmo. Primeiramente separamos as letras, por frequência, e as colocamos em ondem decrescente como mostra a tabela 1, onde é possível notar o numero de vezes que cada caractere aparece e somando temos a quantidade total de caracteres da frase usada como exemplo. Caso algumas letras apareçam com frequência repetida, elas podem ser listadas em qualquer ordem – optamos por deixar em ordem alfabética, por exemplo como acontece com as letras J,L,Q e T, todas de frequência 1.

Uma vez construída nossa tabela de frequência, vamos iniciar a construção da árvore de Huffman. Para melhor compreensão deste processo, ilustramos cada passo com as figuras de 1 a 9. Nessa parte do processo, vamos juntando os caracteres dois a dois e somando suas frequências de modo que vamos criando os caminhos, lembrando que começamos sempre juntado das menores para as maiores e repetindo essa ligação a cada passo. É possível ver na figura 1 que juntamos as letras Q e T somando suas frequências e criando uma ramificação. Para economizar um passo de compressão, o que significa economia de recursos computacionais em um processo real, já juntamos também o J e o L, conforme mostrado na mesma imagem. Já olhando a figura 2 é possível notar que dessa vez iremos juntar as frequências menores novamente, nesse caso a junção de Q+T e J+L bem como também temos D e o P de mesma frequência, e assim seguem os passos seguintes, sempre juntando as menores frequências primeiro. Observando o imagem 5 podemos notar uma ramificação de valor 12 e duas ramificações de valor 16, temos que usar a ramificação 12 primeiro e posteriormente escolhi a ramificação 16 mais a direita como

mostra a imagem 6, continuando o processo até juntar todas as ramificações montando a árvore. Aqui é possível ver que estamos usando um subespaço pois não usamos todas as letras do alfabeto e ele seria considerado o espaço total. Fazendo esse caminho ate juntar todos os caracteres em um único ponto com soma no valor total de caracteres, assim como podemos ver nas imagens de 1 a 8. Lembrando que a árvore é construída dessa forma juntando primeiro os caracteres de menor frequência para que os de maior frequência recebam códigos menores, caso contrário a codificação não seria tão eficaz quanto o esperado. Com isso acabamos de criar a árvore de codificação de Huffman, logo entramos no parte em que daremos um código a cada caractere, na árvore cada ramificação a esquerda recebe código 0 (zero) e cada ramificação a direita recebe código 1 (um), veja como fica a árvore com códigos para seus caminhos na imagem 9. Passado esse processo de construção da árvore de códigos vamos criar a codificação para cada caractere seguindo o caminho das ramificações da frequência total até a frequência de cada caractere individual, assim podemos ver o código pronto de cada caractere na tabela 2.

A	-	S	E	C	I	N	O	R	U	D	P	J	L	Q	T
10	10	7	5	4	4	4	4	4	4	2	2	1	1	1	1

Tabela 1: Caracteres em ordem decrescente de frequência

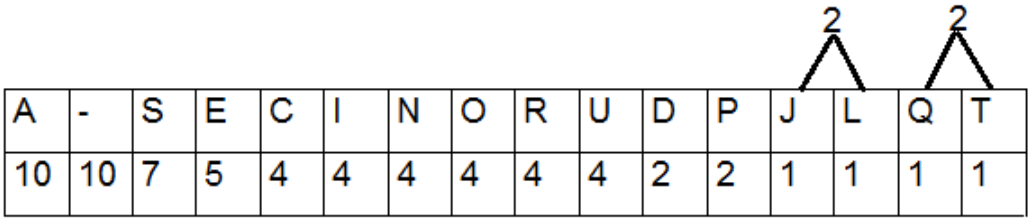

Figura 1: Imagem 1

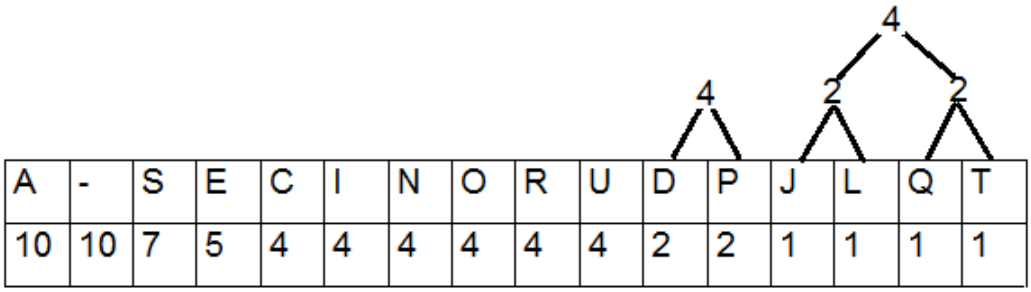

Figura 2: Imagem 2

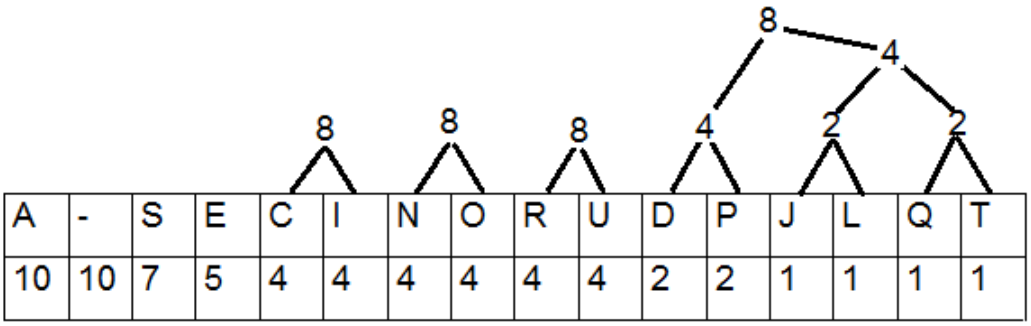

Figura 3: Imagem 3

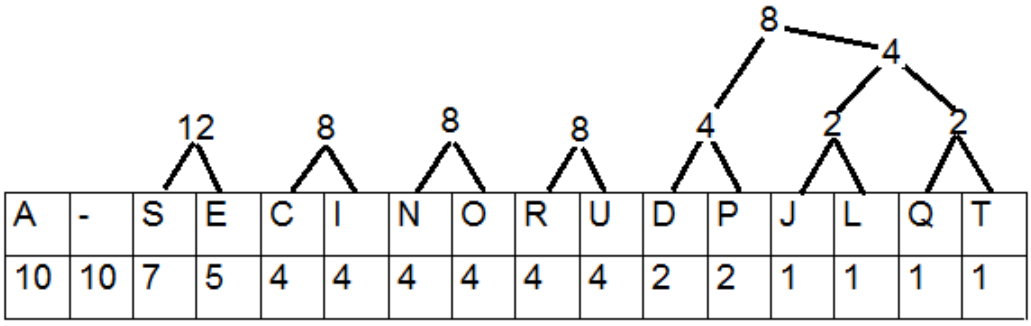

Figura 4: Imagem 4

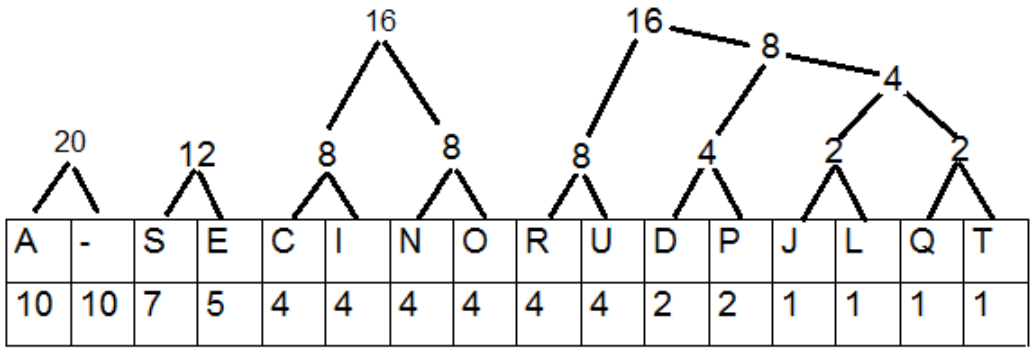

Figura 5: Imagem 5

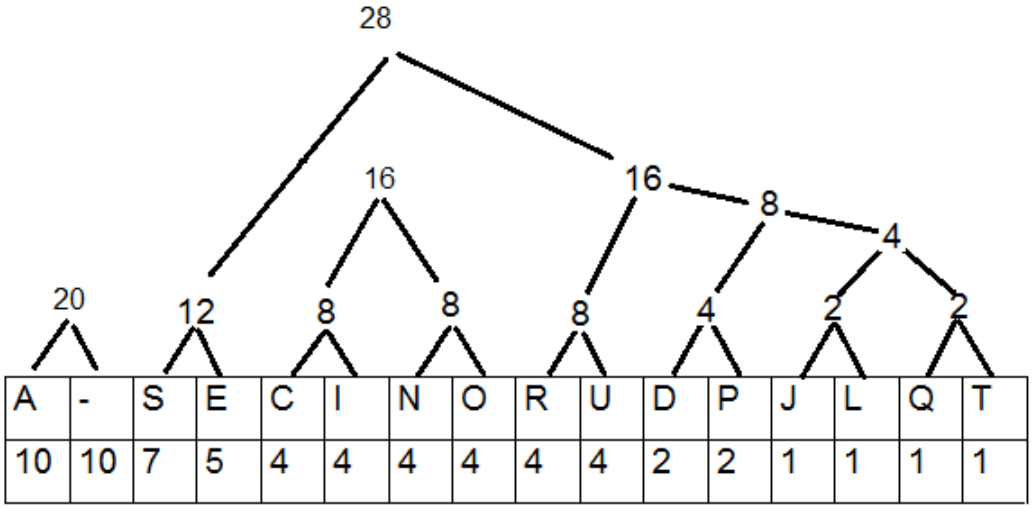

Figura 6: Imagem 6

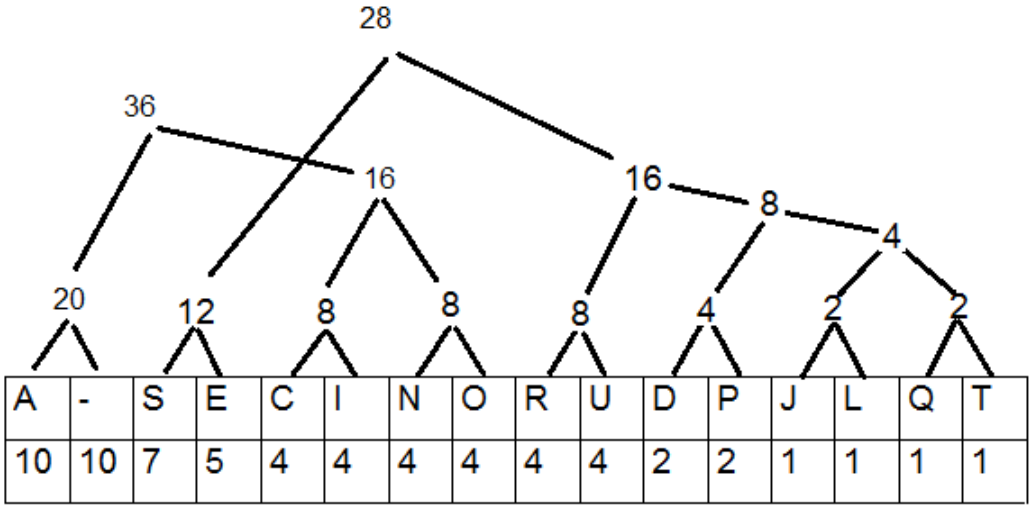

Figura 7: Imagem 7

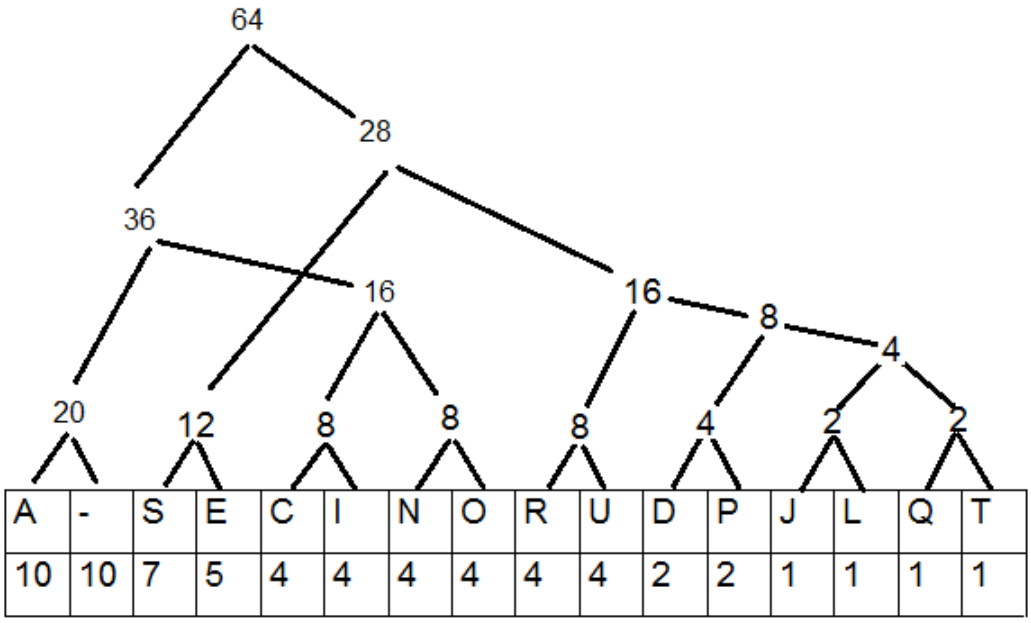

Figura 8: Imagem 8

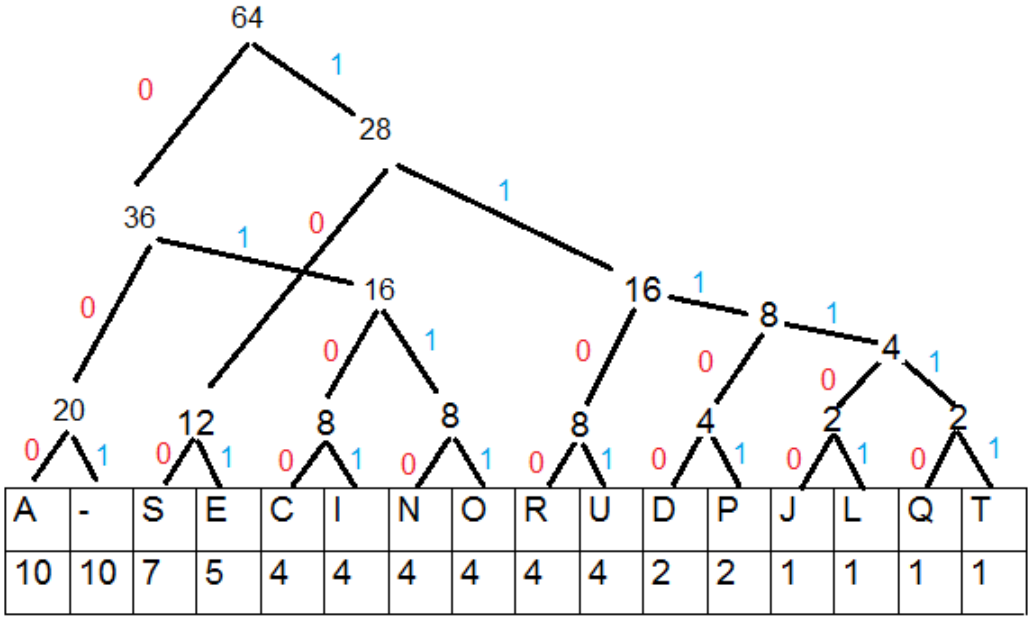

Figura 9: Imagem 9

A	-	S	E	C	I	N	O
000	001	10	101	0100	0101	0110	0111
R	U	D	P	J	L	Q	T
1100	1101	11100	11101	111100	111101	111110	111111

Tabela 2: Códigos Caracteres

Em código binário variável, codificação de Huffman, a frase pode ser escrita com 236 bits, e é representada numericamente assim:

101 11100 1101 0100 000 0101 001 000 100 001 0100 1100 0101 000 0110 0100 000 100 001 11101 000 1100 000 001 111110 1101 101 001 0110 000 0111 001 100 101 111100 000 001 0110 101 0100 101 100 100 000 1100 0101 0111 001 11101 1101 0110 0101 1100 001 0111 100 001 000 11100 1101 111101 111111 0111 100

Em código binário fixo a frase seria escrita com 512 bits, e seria representado da seguinte forma numérica:

01000101 01100100 01110101 01100011 01100001 01101001 00100000 01100001 01110011 00100000 01100011 01110010 01101001 01100001 01101110 11100111 01100001 01110011 00100000 01110000 01100001 01110010 01100001 00100000 01110001 01110101 01100101 00100000 01101110 11100011 01101111 00100000 01110011 01100101 01101010 01100001 00100000 01101110 01100101 01100011 01100101 01110011 01110011 01100001 01110010 01101001 01101111 00100000 01110000 01110101 01101110 01101001 01110010 00100000 01101111 01110011 00100000 01100001 01100100 01110101 01101100 01110100 01101111 01110011

Ao comparar as duas codificações podemos notar que a codificação de Huffman se tonar mais efetiva no intuito de redução de dados.

5 Compressão de Imagens

5.1 Passos para compressão de imagens

A compressão de imagens no padrão JPEG usando a DCT (Transformada discreta do cosseno) e também muito de álgebra linear tem o seguintes passos como mostra a imagem abaixo e cada um dos passos são descritos logo em seguida, conforme pode ser visto em [12].

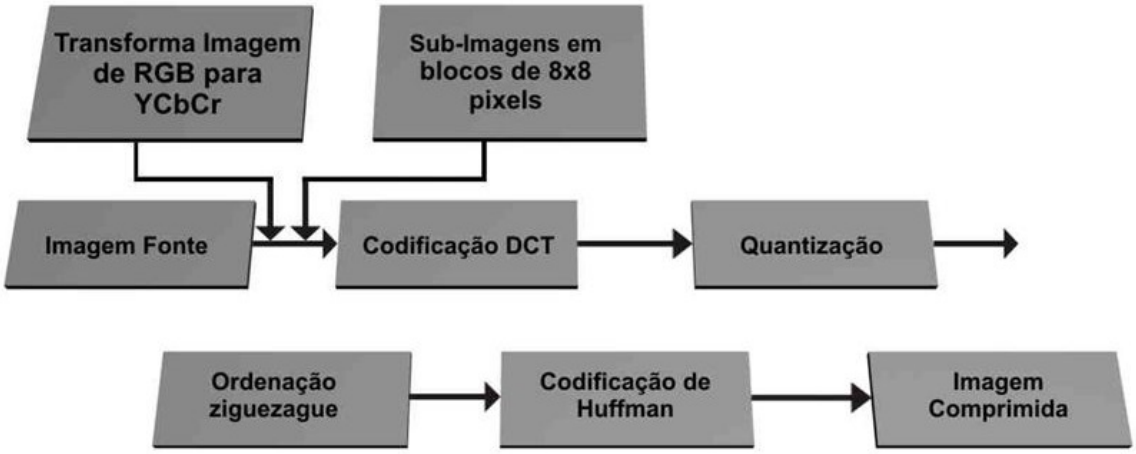

Figura 10: Passos compressão JPEG, Google imagens

- 1º passo. Converter as cores do sistema RGB para YC_bC_r.
- 2º passo. Divisão da imagem em blocos.
- 3º passo. Aplicação da Transformada Discreta do Cosseno (TDC).
- 4º passo. Quantização dos blocos.
- 5ª passo: Sequenciamento zig-zag.
- 6º passo. Codificação da imagem.

Inicialmente temos que RGB é a sigla do sistema de cores aditivas formado pelas iniciais das cores em inglês Red, Green e Blue, que significa em português, respectivamente, Vermelho, Verde e Azul. O sistema de cores luminosas RGB (também designado por cor-luz) é usado nos objetos que emitem luz como, por exemplo, os monitores de computador e televisão, as câmeras digitais, o scanner, entre outros. As cores são obtidas através das misturas das três cores primárias, em quantidades determinadas. Cada uma das cores obtidas estão enquadradas numa escala que varia de 0 a 255. Quando a mistura das três cores está no valor mínimo (0, 0, 0), o resultado é a cor preta. Quanto está no máximo (255, 255, 255), resulta na cor branca. A variação entre valores mínimos corresponde a tons escuros e entre os valores máximos, os tons são mais intensos, mais claros.

Já o sistema YC_bC_r, muito usado em formatos digitais, como fotos e vídeos, consiste em uma componente de brilho Y e duas componentes de cores C_b e C_r (b e r de blue e red, respectivamente).

No passo 2, onde acontece a divisão dos blocos, é feito antes de aplicar DCT (Transformada Discreta do Cosseno) a imagem deve ser separada em blocos (matrizes) para uma maior otimização de tempo e uma taxa mais aceitável de compressão. Normalmente

a imagem é separada em blocos de 8x8 pixels, blocos disjuntos, mas a imagem pode ser dividida em blocos de 4x4, 8x8, 16x16, 32x32 e até 64x64.

No terceiro passo temos que a cada bloco criado no segundo passo, aplicamos o algoritmo da transformada discreta do cosseno, considerado como o passo mais importante no processo de compressão, ela toma um conjunto de dados inciais (dados espaciais) e os converte em dados de frequência. Uma das características mais notáveis na transformada é que os dados resultantes exigem uma quantidade maior de espaço para armazenamento.

A transformada deve eliminar as correlações pois ao observar os pixels em uma imagem natural pode se notar um nível muito alto, ou seja, os pixels próximos são semelhantes e muitas vezes redundantes, com isto necessitamos de representar somente um desses valores, sem perda aparente para a imagem. Uma outra característica importante ao trabalhar com domínio de frequência é que as distorções acontecem onde nossos olhos não percebem, que seriam nas arestas onde ocorre uma troca mais suave de tons.

É importante ressaltar que o trabalho da transformada consiste em converter um bloco de um espaço a um espaço de frequência, ela não desempenha o papel de perdas na compressão, não salva nenhuma informação. As perdas acontecem quando quantizamos os blocos transformados e as informações serão salvas no processo de codificação por entropia, ou seja uma codificação que comprimi sem perdas.

No quarto passo,nota-se que a matriz transformada ocupa mais espaço de armazenamento que a matriz inicial pois ela transforma os elementos inteiros em elementos reais e é onde entra o processo de quantização para diminuir esses dados, ou seja, reduzir o número de bits para armazenar.

Na compressão de imagens as perdas significantes acontecem no processo de quantização, isso pelo fato de que descartamos alguns valores irrelevantes que posteriormente, no processo de decodificação, não poderão ser mais recuperados. Uma imagem que passou por esse processo jamais poderá voltar a seu formato original.

Este processo ocorre logo após obter os blocos transformados, os elementos dos blocos transformados sofrem uma divisão por elementos correspondentes de uma matriz de quantificação seguido por um arredondamento, os elementos reais do bloco transformado perdem a parte decimal e voltam a ser inteiros com isso muitos são convertidos a zeros, onde ocorre as perdas e também ocorre a redução de espaço para armazenamento.

De uma forma bem sutil, o processo de quantizar passa os blocos transformados por uma função de arredondamento, pois é melhor salva zero ou um bit do que algum intervalo infinito entre zero e um.

veja maiores informações sobre quantização em [9] e [10].

No quinto passo ocorre o sequenciamento Zig-Zag, isso antes de passar pelo processo de codificação por entropia, processo sem perdas nesse caso a codificação de Huffman, a matriz sofre uma reordenação em zig-zag e seus elementos perdem a forma matricial de duas dimensões e tomam a forma de sequência em uma dimensão. Existem outras formas de reordenação porém o formato JPEG usa o sequenciamento de zig-zag que se sobressai aos demais, isso pois como os valores importantes ficam no canto esquerdo superior da matriz transformada e esse método os coloca na frente e em ordem crescente de frequência da sequência criada, facilitando a aplicação da codificação e aumentando sua otimização de tempo.

Aqui no sexto e ultimo passo, após passar os passos anteriores, os blocos resultantes são sequências de 64 elementos onde os dados relevantes se encontram no final, e é notável que as sequências derivadas das matrizes esparsa (matriz onde a maioria dos elementos valem zero)possuem grande quantidade de elementos nulos, aumentando a eficacia da codificação. Esses dados são compactados mais uma vez e usando a codificação de Huffman, dessa vez com um algoritmo sem perdas, logo em seguida são salvos no arquivo JPEG.

5.2 Álgebra linear nos passos de compressão

Nessa seção será descrita toda álgebra envolvida na compressão, notável que alguns passos possuem mais álgebra envolvida que outros porém todos tem grande importância. Algumas matrizes aqui descritas podem ser vistas com outros detalhes em [15].

- 1^{o} passo. Converter as cores do sistema RGB para YC_bC_r.
- 2^{o} passo. Divisão da imagem em blocos.
- 3^{o} passo. Aplicação da Transformada Discreta do Cosseno (TDC).
- 4^{o} passo. Quantização dos blocos.
- 5^{a} passo: Sequenciamento zig-zag.
- 6^{o} passo. Codificação da imagem.

Em seguida iremos descrever cada passo aqui citado,

No primeiro passo é feita a conversão do sistema RGB para YC_bC_r, uma das formas é dada pelo sistema abaixo que representa uma transformação linear e tem como função um produto matricial, essa conversão é feita pois o sistema RGB tem seus dados muito bem agrupados e fica difícil a aplicação do DCT já no espectro de crominância é possível trabalhar melhor a transformada, temos esse sistema de conversão logo abaixo que é tipico e já é pré definido para o padrão. Lembrando que nessa transformação vamos considerar as bases canônicas do espaço $\mathbb{R}^3$ e com isso no sistema RGB temos (1,0,0) a cor vermelha, (0,1,0) a cor verde e (0,0,1) a cor azul.

$$\begin{bmatrix} Y \\ C_b \\ C_r \end{bmatrix} = \begin{bmatrix} 0,299 & 0,587 & 0,114 \\ -0,169 & -0,331 & 0,500 \\ 0,500 & -0,419 & -0,081 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

No segundo passo a imagem é separada em matrizes (blocos) quadradas de ordem 8 onde seus elementos são as escalas de cinza de cada pixel.

Exemplo 5.1 *A imagem mostrada vista na figura 11 foi capturada pela câmera NIKON 7200, com sua qualidade máxima possui dimensões de* 6000×4000 *pixels. Para ser feita a compressão dessa foto, ela deve ser separada em blocos (matrizes) de* 8×8 *disjuntos, temos no entanto* 750×500 *blocos, num total de 375.000 matrizes de 64 elementos. Ao compararmos a mesma foto salva em RAW e em JPEG é possível notar a diferença no espaço ocupado, onde uma ocupa um espaço de 31,6 MB e a outra ocupa 19,1 MB, nesse caso uma economia de 40% no armazenamento.*

Figura 11: Imagem acervo pessoal

Exemplo 5.2 *Aqui no trabalho vamos usar matrizes na sua forma mais simples, como matrizes de imagens em preto e branco, apenas para dar uma ideia de como funciona o processo lembrando que imagens coloridas possuem 3 componentes na entrada de cada pixel e o processo é aplicado a cada umas dessas componentes. Porém, um bloco* 8×8 *de uma imagem colorida seria algo parecido com*

$$\begin{bmatrix} (5,11,243) & (237,44,158) & (23,115,51) & (120,68,170) & \dots \\ (116,196,159) & (61,255,36) & (149,205,28) & (245,246,27) & \dots \\ (147,80,229) & (13,207,100) & (247,214,77) & (0,142,142) & \dots \\ (128,163,101) & (96,193,255) & (185,23,66) & (158,111,122) & \dots \\ (120,25,93) & (176,57,45) & (34,211,211) & (110,178,103) & \dots \\ (9,98,152) & (45,169,230) & (44,206,188) & (179,248,205) & \dots \\ (240,213,80) & (0,75,98) & (71,0,255) & (122,111,23) & \dots \\ (154,106,69) & (106,70,201) & (70,23,45) & (69,255,0) & \dots \end{bmatrix}.$$

Notem que na matriz acima, por falta de espaço, representamos apenas as primeiras quatro colunas da mesma. Cada elemento dessa matriz é um vetor de 3 componentes, seja ele na escala RGB, ou depois da conversão na escala YC_bC_r*. Entretanto, como ja'dito anteriormente, para demonstrar os príncipios da compressão de imagens basta*

considerarmos um bloco de mesmo tamanho representando uma imagem em escala de cinzas. Temos então algo como a matriz abaixo

$$\begin{bmatrix} 52 & 55 & 61 & 66 & 70 & 61 & 64 & 73 \\ 63 & 59 & 55 & 90 & 109 & 85 & 69 & 72 \\ 62 & 59 & 68 & 113 & 144 & 104 & 66 & 73 \\ 63 & 58 & 71 & 122 & 154 & 106 & 70 & 69 \\ 67 & 61 & 68 & 104 & 126 & 88 & 68 & 70 \\ 79 & 65 & 60 & 70 & 77 & 68 & 58 & 75 \\ 85 & 71 & 64 & 59 & 55 & 61 & 65 & 83 \\ 87 & 79 & 69 & 68 & 65 & 76 & 78 & 94 \end{bmatrix}.$$

No terceiro passo, como estamos comprimindo imagens e seus dados estão em matrizes usaremos a TDC bidimensional na compressão, com isso será usada seguinte expressão.

$$G(i,j) = \alpha(i)\alpha(j) \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(x,y) \cos\left(\frac{(2x+1)i\pi}{2N}\right) \cos\left(\frac{(2y+1)j\pi}{2N}\right),$$

para $i, j = 0, 1, ..., N-1$, com

$$\alpha(x) = \begin{cases} \dfrac{1}{\sqrt{N}} & \text{se} \quad x = 0, \\ \sqrt{\dfrac{2}{N}} & \text{se} \quad x = 1, 2, ..., N-1, \end{cases}$$

onde $G_{i,j}$ é um elemento da matriz Transformada Discreta do Cosseno, $f(x,y)$ Valores da imagem original (escalas de cinza) e N Dimensão da imagem (ou bloco).

Exemplo 5.3 *Seja a matriz A, um bloco retirado de uma imagem qualquer, nele será aplicado o processo da DCT.*

$$A = \begin{bmatrix} 52 & 55 & 61 & 66 & 70 & 61 & 64 & 73 \\ 63 & 59 & 55 & 90 & 109 & 85 & 69 & 72 \\ 62 & 59 & 68 & 113 & 144 & 104 & 66 & 73 \\ 63 & 58 & 71 & 122 & 154 & 106 & 70 & 69 \\ 67 & 61 & 68 & 104 & 126 & 88 & 68 & 70 \\ 79 & 65 & 60 & 70 & 77 & 68 & 58 & 75 \\ 85 & 71 & 64 & 59 & 55 & 61 & 65 & 83 \\ 87 & 79 & 69 & 68 & 65 & 76 & 78 & 94 \end{bmatrix}$$

Para a aplicação da DCT os coeficientes são normalizados com valores entre -128 e 127, tendo que os valores iniciais estão variando de 0 a 255 numa escala de cinza, essa normalização é feita subtraindo 128 de cada componente. Fazendo então $N_{i,j} = A_{i,j} - 128$ para cada elemento da matriz inicial A, criamos uma matriz normalizada como mostra a seguir:

$$N = \begin{bmatrix} -76 & -73 & -67 & -62 & -58 & -67 & -64 & -55 \\ -65 & -69 & -73 & -38 & -19 & -43 & -59 & -56 \\ -66 & -69 & -60 & -15 & 16 & -24 & -62 & -55 \\ -65 & -70 & -57 & -6 & 26 & -22 & -58 & -59 \\ -61 & -67 & -60 & -24 & -2 & -40 & -60 & -58 \\ -49 & -63 & -68 & -58 & -51 & -60 & -70 & -53 \\ -43 & -57 & -64 & -69 & -73 & -67 & -63 & -45 \\ -41 & -49 & -59 & -60 & -63 & -52 & -50 & -34 \end{bmatrix}$$

Aplicando a DCT a cada elemento da matriz N, obtemos a matriz G como resultado, será citada como matriz transforma:

$$G = \begin{bmatrix} -415,38 & -30,19 & -61,20 & 27,24 & 56,13 & -20,10 & -2,39 & 0,46 \\ 4,47 & -21,86 & -60,76 & 10,25 & 13,15 & -7,09 & -8,54 & 4,88 \\ -46,83 & 7,37 & 77,13 & -24,56 & -28,91 & 9,93 & 5,42 & -5,65 \\ -48,53 & 12,07 & 34,10 & -14,76 & -10,24 & 6,30 & 1,83 & 1,95 \\ 12,12 & -6,55 & -13,20 & -3,95 & -1,88 & 1,75 & -2,79 & 3,14 \\ -7,73 & 2,91 & 2,38 & -5,94 & -2,38 & 0,94 & 4,30 & 1,85 \\ -1,03 & 0,18 & 0,42 & -2,42 & -0,88 & -3,02 & 4,12 & 0,66 \\ -0,17 & 0,14 & -1,07 & -4,19 & -1,17 & -0,10 & 0,50 & 1,68 \end{bmatrix}$$

Entrando no quarto passo, o bloco transformado é passado por um processo de quantização. Aqui é onde temos as perdas permanentes, uma vez que os elementos transformados são divididos por elementos correspondentes de uma matriz de quantização e em seguida são arredondados para o inteiro mais próximo. O processo de quantização pode ser descrito pela expressão:

$$F_{i,j} = round\left(\frac{G_{i,j}}{Q_{i,j}}\right),$$

para $i, j = 0, 1, 2, ..., 7$, onde $G_{i,j}$ é a matriz transformada, $Q_{i,j}$ é a matriz de quantização típica e $F_{i,j}$ é a matriz resultante já quantizada e pronta para o sequenciamento zig-zag.

Os elementos da matriz de quantização $Q_{i,j}$, são criados através de uma formula própria descrita por

$$Q_{i,j} = 1 + (1 + i + j) * \text{fator de qualidade},$$

sendo que o Fator de Qualidade pode variar entre 2 e 25, pois valores acima de 25 afetam muito a qualidade final da imagem, então se deseja ter uma boa taxa de compressão com as perdas insignificantes é necessário que o fator de qualidade esteja no intervalo sugerido, lembrando que quanto menor maior qualidade e menor compressão .

Exemplo 5.4 *Veja abaixo uma matriz de quantização típica usada pelo padrão JPEG. Outras matrizes de quantização poderão ser criadas desde que se especifique. Utilizando a*

matriz G descrita no 3ª passo e passando ela pela quantização, obtemos a matriz F abaixo:

$$Q = \begin{bmatrix} 16 & 11 & 10 & 16 & 24 & 40 & 51 & 51 \\ 12 & 12 & 14 & 19 & 26 & 58 & 60 & 55 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 14 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\ 18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\ 24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\ 72 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{bmatrix}$$

$$F = \begin{bmatrix} -26 & -3 & -6 & 2 & 2 & -1 & 0 & 0 \\ 0 & -2 & -4 & 1 & 1 & 0 & 0 & 0 \\ -3 & 1 & 5 & -1 & -1 & 0 & 0 & 0 \\ -3 & 1 & 2 & -1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Exemplo 5.5 *Cada elemento da matriz G passa polo processo de quantização e forma os elementos de F, como será mostrado a seguir para o primeiro elemento.*

$$f_{0,0} = round\left(\frac{g_{0,0}}{q_{0,0}}\right)$$

$$f_{0,0} = round\left(\frac{-415,38}{16}\right)$$

$$f_{0,0} = round(-25,96125)$$

$$f_{0,0} = -26$$

Já no quinto passo, temos que o processo somente reordena os coeficientes da matriz, para a codificação, conforme mostra a imagem abaixo, para exemplificar de forma mais clara observe o exemplo a seguir:

Exemplo 5.6 *Considere a matriz A, uma matriz esparsa, podemos notar, que a sequencia logo abaixo foi criada com os elementos de A, reordenados segundo o sequenciamento zig-zag.*

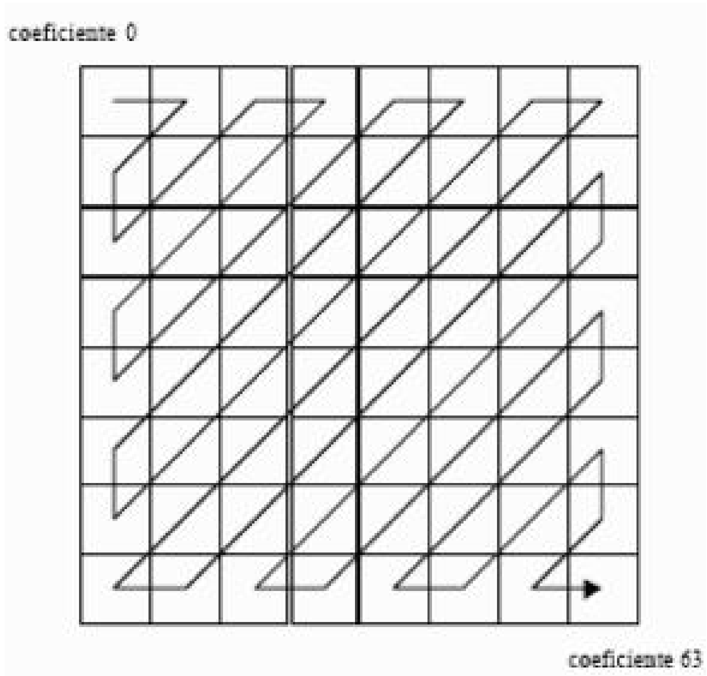

Figura 12: Sequência zig-zag,Google imagens

$$A = \begin{bmatrix} -46 & 24 & 9 & 2 & 0 & -3 & -1 & 1 \\ 26 & 18 & -5 & 4 & 1 & -2 & 0 & 0 \\ -6 & 3 & -1 & 0 & 6 & 0 & 0 & 0 \\ 4 & -8 & 1 & 1 & 0 & 0 & 0 & 0 \\ 7 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

[-46, 24, 26, -6, 18, 9, 2, -5, 3, 4, 7, -8, -1, 4, 0, -3, 1, 0, 1, -1, -1, 0, 0, 0, 1, 6, -2, -1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

Ao chegar aqui, no sexto passo, os dados transformados, quantizados e sequenciados vem em forma de sequência numérica. Essa sequência sofre um pouco de influência estatística, onde seus elementos são ordenados de forma decrescente em referência a sua frequência de ocorrência. Como é mostrado no Exemplo 4.1, que será relembrado em partes a seguir. Lembrando que no processo de compressão estaremos trabalhando com valores numéricos e não letras como mostra o exemplo, porem a codificação segue o mesmo raciocínio para codificação. Baseado no exemplo temos que em seu primeiro passo os dados são ordenados por frequência, como mostra a tabela 3. O segundo passo é criar a árvore de Huffman, ver imagem 13. Por fim montamos os códigos referentes aos caracteres, escrevemos a sequência codificada e está pronto a ser salvo no arquivo JPEG, assim

A	-	S	E	C	I	N	O	R	U	D	P	J	L	Q	T
10	10	7	5	4	4	4	4	4	4	2	2	1	1	1	1

Tabela 3: Caracteres em ordem decrescente de frequência

como mostra a figura 14.

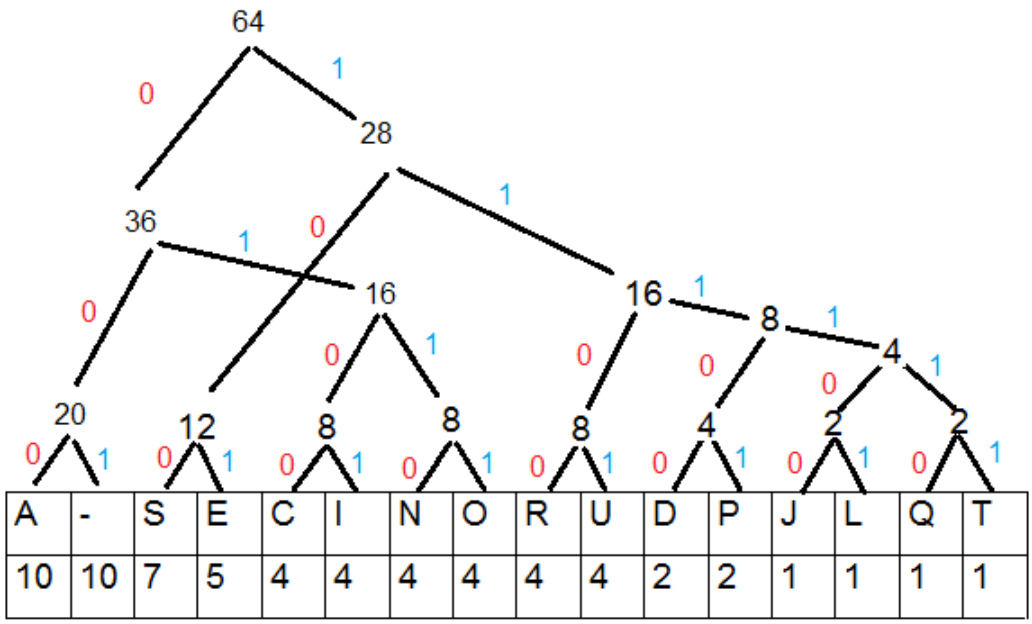

Figura 13: Árvore de Huffman

A	-	S	E	C	I	N	O
000	001	100	101	0100	0101	0110	0111
R	**U**	**D**	**P**	**J**	**L**	**Q**	**T**
1100	1101	11100	11101	111100	111101	111110	111111

Figura 14: Códigos caracteres

Em código binário variável, codificação de Huffman, a frase pode ser escrita com 236 bits, e é representada numericamente assim:

101 11100 1101 0100 000 0101 001 000 100 001 0100 1100 0101 000 0110 0100 000 100 001 11101 000 1100 000 001 111110 1101 101 001 0110 000 0111 001 100 101 111100 000 001 0110 101 0100 101 100 100 000 1100 0101 0111 001 11101 1101 0110 0101 1100 001 0111 100 001 000 11100 1101 111101 111111 0111 100.

6 Aplicação no Ensino Básico

Antes de inciar alguma atividade em qualquer turma devemos mostrar aos alunos o que é apresentado aqui e posteriormente introduzir o pensamento crítico em algumas situações, por exemplo: Como a combinação de cores monta uma imagem? Códigos usados na informática devem ser aplicados? Álgebra linear aplicada a compressão de imagens, onde? O que é uma matriz? Quais operações envolvendo matrizes eu irei aplicar?

Ainda que de modo visionário, este trabalho tem assuntos que podem ser trabalhados no ensino médio e também no ensino fundamental, o ensino da matemática de forma interdisciplinar com a área tecnológica do tratamento de imagens e vídeos. Em alguns casos, quando ensinamos matemática no ensino médio, introduzimos a noção de matrizes e suas operações, sempre temos aqueles alunos que questionam "onde eu poderei aplicar esse conhecimento?", aqui é possível ver com clareza a grande aplicação. No ensino fundamental já é possível introduzir a noção de codificação binaria mostrando quantidades e formas de relacionar a escrita binaria na matemática com a computação. Este trabalho pode ser usado como apoio a professores e alunos que desejam deixar mais interessante a obtenção e domínio do conteúdo de matrizes, mostrando a vasta aplicação no cotidiano.

Exemplo 6.1 *Cada aluno escolher uma foto em seu acervo pessoal e analisar o formato em que foi salvo, tamanho que a imagem ocupa, verificar se existe a necessidade e possibilidade de compressão, quantas matrizes 8x8 existem naquela imagem.*

Exemplo 6.2 *Comparar uma foto antes e depois da compressão, tamanho para armazenamento, formato, verificar se aquela é uma compressão aceitável.*

Exemplo 6.3 *Pegar uma matriz 8x8 qualquer e fazer o sequenciamento zig-zag com seus elementos, escrevendo os mesmos em linha, sempre olhando a posição dos elementos. É possível fazer a atividade ao contrário com uma sequência de 64 elementos, escrevendo os mesmos em uma matriz 8x8.(Pode ser aplicado no ensino fundamental dando introdução a noção de sequência numérica.)*

Considerações Finais

Com este trabalho abordamos uma pequena parte da aplicação de Álgebra Linear que existe por trás de um "simples"processo, que é a compressão de uma imagem, ou melhor dizendo a mudança do formato de uma imagem para reduzir o tamanho do armazenamento, facilitando a transmissão via rede e mantendo uma qualidade aceitável. Dentro desse processo de cada passo está permeado pelo uso de ferramentas vindas da Álgebra Linear. Por exemplo, ao usarmos a Transformada Discreta do Cosseno, lidamos com conceitos como as bases ortogonais e ortonormais, e também com mudanças de bases.

Citando outros exemplos, temos uma transformação linear no produto entre matrizes para mudança de um sistema de cor a outro, que também pode ser visto também como mudança de base, separação de um bloco grande de pixels (imagem) em matrizes quadradas 8×8 para a aplicação da transformada discreta do cosseno, uso de subespaços vetoriais

quando trabalhamos apenas com a base necessária para codificação da mensagem, entre outros. Em resumo, existe matemática em praticamente tudo que trabalhamos. No entanto, o trabalho além de mostrar a Álgebra Linear por trás de uma situação que muitas vezes passa despercebida, alcança uma maior proporção dado que se faz necessário chamar a atenção para o ensino de matemático em tempos atuais, em todos os níveis de estudo, para que isso busque e cresça o interesse dos alunos dentro das salas.

Abordamos também alguns conceitos de programação, como o padrão JPEG que é considerado uma excelente técnica de compressão e formato de armazenamento, onde sua aplicação tem maior eficiência como em imagens naturais de tons contínuos, e também onde não é recomendado como imagens em preto e branco, textos ou gráficos. Ainda falando sobre programação temos a codificação de Huffman que mostra grande auxilio no processo de compressão ao codificar os dados com códigos binários de tamanhos variáveis, a qual também usa conceitos matemáticos de probabilidade. Ressaltamos aqui que o processo mostrado é uma versão muito simplificada do que acontece realmente, onde técnicas cada vez melhores visam melhorar a compressão de arquivos para seu crescente uso. Entretanto as ideias principais estão presentes aqui no texto, e esperamos ter alcançado nosso principal objetivo de mostrar o recorrente uso de Matemática, em particular da Álgebra Linear, em processos presentes no cotidiano de nossos alunos.

Espero que este trabalho possa ajudar aos professores do ensino básico, oferecendo a eles uma forma diferente de trabalhar alguns conceitos matemáticos, contextualizado com a área de computação, abordando, com didática, assuntos onde a Álgebra Linear tem grande aplicação como a compressão de imagens.

Dedicatória

Dedico esse trabalho as pessoas que amo, em especial a meu filho, Luís Guilherme, que é para quem desejo ser um bom exemplo de pessoa, esforço e conhecimento. Dedico também a mim mesmo, pois está sendo uma conquista e parte da realização de um sonho.

Agradecimentos

Agradeço, primeiramente, a mim mesmo por nunca desistir e ser meu maior apoio sempre.

A todos que me apoiaram, deram suporte e sempre incentivaram a ser melhor e continuar melhorando.

A mãe de meu filho e amiga, que sempre me apoiou nos estudos e me ajudou demais com a criação de nosso filho, muitas vezes sozinha.

Aos meus colegas de mestrado, que sempre ajudaram nos estudos e dúvidas, tanto na turma de 2014 quando na turma de 2018.

Ao Wálmisson Régis de Almeida, que sempre me ajudou a ser alguém melhor, que me ajudou demais nas viagens a Ouro Branco, e mostrou ser um ótimo amigo.

Aos professores que me ensinaram demais, mostraram que devo estudar com mais dedicação e tiveram grande influência no meu crescimento.

A meus grandes amigos Daniela Maia e Patrick Fonseca, que sempre me ajudaram em momentos difíceis e sempre buscando o meu melhor.

À CAPES pelo apoio financeiro, possibilitou mais tempo para os estudos e com mais tranquilidade.

E a todos que me deram apoio, mesmo de forma subliminar.

Referências

[1] BOLDRINI, J. L. Álgebra Linear. 3ªed., São Paulo - Brasil: Editora Harbra, 1986.

[2] SANTOS, Reginaldo J. Introdução à Álgebra Linear. 1ªed., Belo Horizonte - Brasil: Imprensa Universitaria da UFMG, 2013.

[3] HEFEZ, Abramo; FERNANDEZ, Cecília de Souza. Introdução à Álgebra Linear. 2ªed., Rio de Janeiro - Brasil: Editora SBM - Coleção Profmat, 2016.

[4] HEFEZ, Abramo. Aritmética. 3ªed., Rio de Janeiro - Brasil: Editora SBM - Coleção Profmat, 2013.

[5] STEINBRUCH, Alfredo; WINTERLE, Paulo. Álgebra Linear. 2ªed., São Paulo - Brasil: Editora PEARSON, 1987.

[6] BOYCE, William E.; DIPRIMA, Richard C. Equações Diferenciais Elementares e Problemas de Valores de Contorno. 9ªed., Rio de Janeiro - Brasil: Editora LTC, 2010.

[7] Codificação de Huffman 1. Disponível em: <http://www.inf.unioeste.br/∼adair/PID/Notas%20Aula/Codificacoes %20Huffman%20e%20Aritmetica.pdf> Acesso em: 25 de Setembro de 2016

[8] Codificação de Huffman 2. Disponível em: <http://www.inf.ufes.br/∼pdcosta/ensino/2009-1-estruturas-de-informacao/material/CodificacaoHuffman.pdf> Acesso em: 25 de Setembro de 2016

[9] SILVA, Francisco Assis da, Compressão de imagens usando a Transformada discreta do cosseno. Dissertação de Graduação, Unoeste, FIPP, Presidente Prudente, 1998.

[10] LIMA, Nilson Teixeira, Compressão de imagens JPEG. Trabalho de Graduação, Processamento digital de sinais, UFPR, Presidente Prudente, 2007.

[11] GROLLI, Wellingtom Luiz Julio Rodrigues, Compressão de imagens (JPEG). Trabalho de Graduação, Processamento digital de sinais, UFPR, Presidente Prudente, 2007.

[12] FRASSON, Miguel, Transformada Discreta de Cosseno: uma aplicaçãao da Álgebra Linear na compressão de imagens do formato JPEG, 2013.

[13] VASCONCELOS; CARTAXO; RIBEIRO; ANDRYÊ, Transformada Discreta do Cosseno Aplicada ao Padrão JPEG, IFPB.

[14] Significado JPEG. Disponível em: <https://www.significados.com.br/jpeg/> Acesso em 13 de Novembro de 2016

[15] SILVA, Felipe A.; DOMINIQUINI, Rafael B. Técnicas para identificação de manipulações baseadas em insconsistências de compressão. Ttrabalho de graduação, 2010.

[16] As Séries de Fourier. Disponível em: <https://seara.ufc.br/pt/tintim-por-tintim/matematica/as-series-de-fourier/> Acesso em:04 de Janeiro de 2017

[17] Séries de Fourier Disponível em: <http://www.feis.unesp.br/Home/departamentos /engenhariaeletrica/mcap03.pdf> Acesso em:04 de Janeiro de 2017

[18] BIEZUNER, Rodney j. Introdução às Equações Diferenciais Parciais. Notas de Aula, UFMG, 2007.

[19] SANTOS, Reginaldo J. Séries de Fourier. UFMG, 2002.

[20] BOYCE, William E. e DIPrIMA, Richard. Equações Diferenciais Elementares e Problemas de Valores de Contorno. 8ªed., Rio de Janeiro - Brasil: Editora LTC, 2005.

[21] SANTOS, Reginaldo J. Transformada de Fourier,2010. Disponível em: <http://www.mat.ufmg.br/∼regi/eqdif/transfourier.pdf> Acesso em:05 de Janeiro de 2017

[22] Espaços vetoriais, exemplos. Disponível em: <https://www.ime.unicamp.br/ marcia/AlgebraLinear/Arquivos%20PDF/exemplos_espacos.pdf> Acesso em: 21 de Agosto de 2020