2
Conceitos Básicos

A seguir serão explicados alguns conceitos necessários para o entendimento deste trabalho. Por ser um trabalho que envolve diferentes áreas de conhecimento, julgou-se necessário uma breve explicação de cada uma das áreas em que este trabalho se referencia.

Este trabalho tem como base a natureza dos dados sísmicos, o que justifica um capítulo introdutório sobre a aquisição do mesmo. Como este trabalho propõe a utilização de atributos sísmicos, é feita uma explicação sobre como estes são calculados e qual a sua utilização.

A transformada *wavelets* se encaixa na área de processamento de sinais. Será descrito como este tipo de transformada funciona e será feita uma comparação com a transformada de Fourier (que é a mais conhecida e utilizada). Ainda dentro da área de processamento de sinais, será descrito o processo de quantização de um sinal, qual a sua utilização e seus benefícios.

Por fim, será tratada a parte de algoritmos de compressão, como estes funcionam e quais os principais algoritmos conhecidos.

2.1.
Sísmica

Este trabalho foi desenvolvido através de uma parceria entre a Petrobras e o laboratório Tecgraf da PUC-Rio. Esta parceria já acorre há alguns anos e tem apresentado bons resultados, como mostram as publicações [5], [12], [13], [14], [15], [16].

A exploração de hidrocarbonetos está divida em três etapas. A primeira consiste na aquisição do dado. Nesta etapa, uma fonte de energia, que pode ser um explosivo, no caso de aquisições terrestres, ou um canhão de ar comprimido, na aquisição marítima, produz uma onda que viaja até uma camada abaixo da superfície terrestre. Ao encontrar a interface entre duas camadas com impedância acústicas diferentes, parte da energia, da onda sísmica, é refletida e outra parte

segue viajando na nova camada. Um exemplo de uma aquisição marítima pode ser visto na figura .

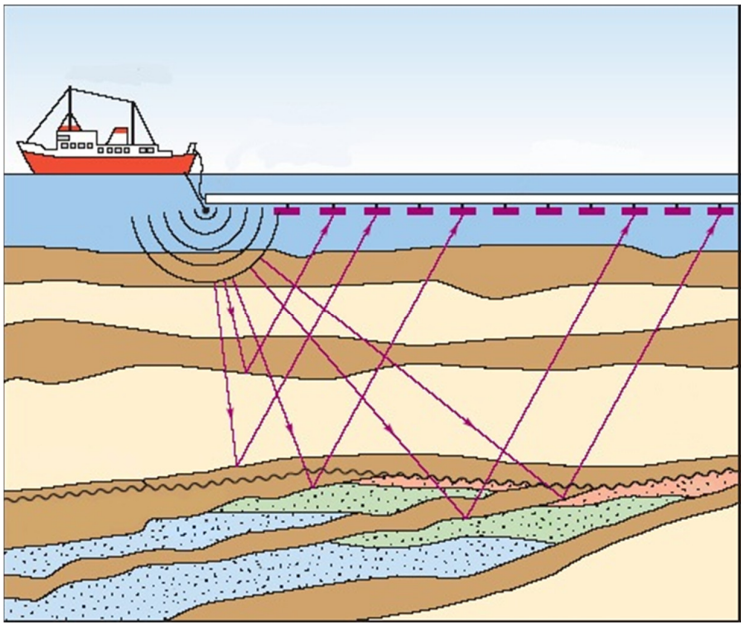

Figura 1 Aquisição de um dado sísmico marítimo. Adaptado de [17]

O resultado do levantamento sísmico é um conjunto de pontos que possuem a posição da fonte no momento do disparo, a posição do receptor no instante em que a onda de energia foi captada, o tempo de viagem da onda (o tempo gasto para a onda de energia ir até uma determinada camada e voltar até o receptor) e a amplitude da onda recebida pelo receptor.

Na aquisição sísmica, assume-se que ao ser refletida, a onda possui um ângulo de reflexão igual ao ângulo de incidência. Neste caso, determina-se que a posição da camada onde existiu o choque com a onda que foi refletida é o ponto médio entre a fonte e o receptor. Este ponto é chamado de CMP (*commom midpoint*). Este tipo de dado é chamado de dados pré-empilhado.

A segunda etapa do processo de exploração sísmica consiste no processamento do dado pré-empilhado. Aqui, os dados são tratados, ou seja, os ruídos são identificados e atenuados. Determinados eventos sísmicos têm suas posições deslocadas pela correção da perda de energia da onda, e outros processamentos são executados.

Outro processo aplicado é o empilhamento do dado. Este processo consiste em gerar uma grade regular para a área de aquisição. Para cada cela da grade, identificar todos os traços sísmicos pertencentes a esta cela e fazer a média entre os traços para reduzir o ruído. Ao fim do processamento, é gerado um novo dado

sísmico, que é chamado de pós-empilhado. A quantidade de informação no dado pós-empilhado é muito menor do que no dado pré-empilhado.

A figura abaixo mostra a visualização de um volume pós-empilhado.

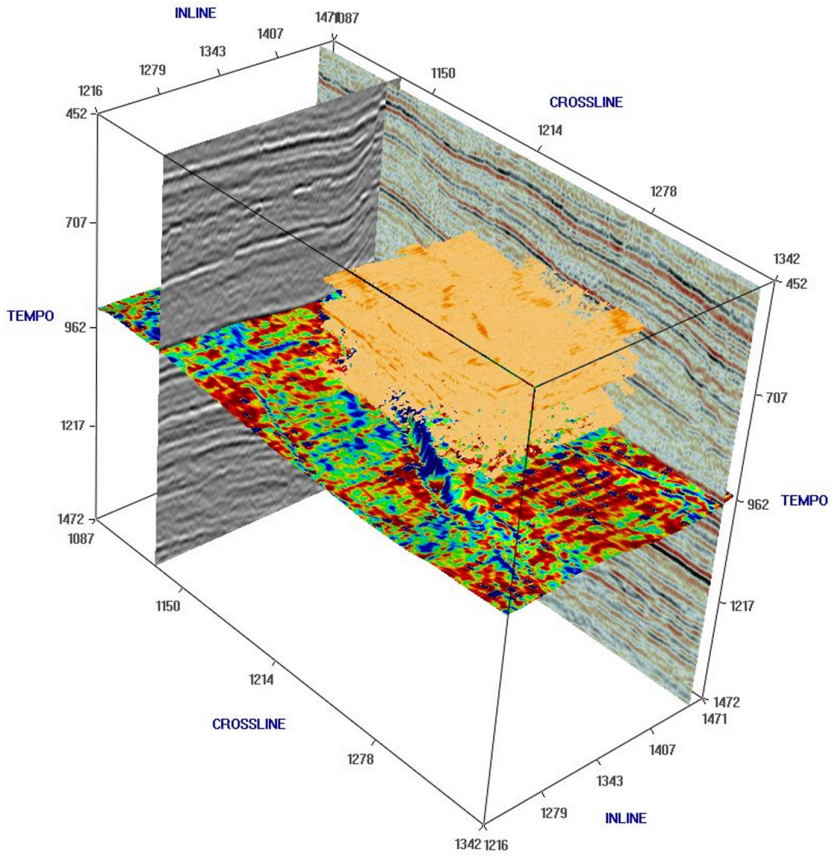

Figura 2 Volume post-stack com fatias, horizonte e probe (sub-região de interesse).

Por último há a etapa de interpretação, que é a etapa em que o intérprete (geólogo ou geofísico) estuda o dado adquirido e processado a procura de reservatórios de hidrocarbonetos. Nesta etapa, são utilizadas ferramentas para ajudar o intérprete a visualizar informações relevantes do dado.

Estas ferramentas vão desde os atributos sísmicos utilizados para identificar características do dado, a modernas técnicas de computação gráfica utilizadas para garantir uma maior interatividade entre o dado e o intérprete.

2.1.1. Atributos Sísmicos

Este trabalho será divido em duas partes. Na primeira parte é apresentada os benefícios do uso da transformada *wavelet* 3D na compressão de dados sísmicos volumétricos. Já a segunda parte, apresenta um estudo sobre a utilização do traço

sísmico complexo na compressão do dado sísmico. Neste ponto serão utilizadas algumas combinações de atributos para a compressão do dado.

De acordo com Chopra [18], o atributo sísmico é uma medida quantitativa de uma característica sísmica de interesse. Chopra afirma que os atributos sísmicos são utilizados desde 1930, quando o intérprete procurava por eventuais coerências no "tempo de trânsito" das ondas refletidas. O avanço do uso dos atributos sísmicos está diretamente relacionado com o avanço da tecnologia disponível. Por exemplo, quando surgiram as impressoras coloridas na década de 70, passou-se a usar cores para representar as amplitudes sísmicas, assim, à cor passou a ser um atributo que representa a amplitude da onda sísmica.

Taner [19], por outro lado, define o atributo sísmico como sendo toda informação obtida através do dado sísmico, seja por medidas diretas ou por raciocínio lógico, ou de forma experimental.

Ambos os autores associam o avanço e a popularização dos atributos sísmicos à introdução do traço complexo na década de 70. Taner *et al* [20] definem o traço complexo como sendo a transformada de Hilbert para o traço sísmico. Esta transformada determina uma função $f^*(t)$ como sendo a convolução da função $f(t)$ com $1/t$. Uma demonstração de como calcular a transformada de Hilbert utilizando a transformada de Fourier pode ser encontrada em [20] e [5]. Neste momento é necessário apenas saber que o traço sísmico ao ser adquirido, pode ser representado como sendo uma função $f(t)$ no domínio dos números reais e o traço complexo será representado por:

$$F(t) = f(t) + if^*(t) \quad \textbf{(2.1)}$$

onde $f^*(t)$ representa a parte imaginária do traço complexo. Esta componente é calculada unicamente em função de $f(t)$ e representa o resultado da transformada de Hilbert aplicado no traço $f(t)$.

Chopra [18] e Taner [20] divergem sobre a classificação dos atributos sísmicos mas, aparentemente, esta divergência ocorre por não existir uma única classificação correta. Para Taner, os atributos são divididos de acordo com as características do domínio do dado sísmico, ou seja, em atributo pré-empilhamento e atributo pós-empilhamento.

Nos atributos pré-empilhados, encontram-se os *CDP* (*common depth point)*, azimute, *offset*, dentre outros. Os atributos pré-empilhados geram uma grande

quantidade de informações, porém sem muita utilidade para a interpretação. Essas informações serão utilizadas no processo de "migração", gerando os dados pós-empilhados. Com estes novos dados, os atributos são subdivididos de acordo com suas características computacionais, ou seja, se eles são calculados amostra a amostra, se levam em consideração a onda de reflexão, dentre outras características. Estes atributos podem ser classificados como: atributos instantâneos, atributos físicos, atributos geométricos e outros.

Os atributos instantâneos são os atributos calculados amostra a amostra. Estes atributos podem ser calculados utilizando o traço complexo. Os mais utilizados são o envelope, a fase instantânea e a freqüência instantânea. Já os atributos físicos estão relacionados a características físicas do dado. Por exemplo, pode-se calcular a magnitude do envelope do traço baseado na impedância acústica do material. Quanto aos atributos geométricos, são aqueles que têm como objetivo evidenciar uma determinada feição do dado sísmico, seja uma falha ou um problema de aquisição do dado.

Um atributo sísmico é utilizado quando o intérprete deseja evidenciar alguma característica do dado a ser estudado. Estas características podem variar desde realçar uma coerência lateral até agrupar determinados tipos de materiais. Cada atributo sísmico tem um objetivo e cabe ao intérprete escolher o que melhor se adéqua ao que se pretende estudar no dado sísmico em questão.

O envelope é um atributo que tem como características discriminar o contraste entre as impedâncias acústicas, realçar os limites de camadas geológicas, eliminar os efeitos de *tuning* entre camadas muito finas, dentre outras. O envelope é calculado como sendo

$$\boldsymbol{A(t) = \sqrt{f^2(t) + f^{*2}(t)}} \tag{2.2}$$

onde $A(t)$ representa o envelope do traço $f(t)$, conforme definido em [20].

A figura 3 mostra uma seção de um determinado dado sísmico. O atributo que está sendo apresentado é a amplitude sísmica, um atributo proveniente da aquisição, ou seja, um atributo medido.

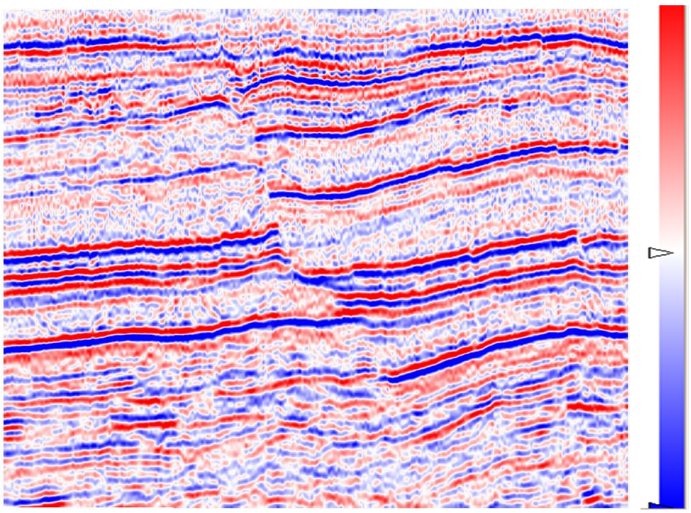

Figura 3 Amplitude sísmica de um dado.

A figura 4 é a representação do envelope que foi calculado, utilizando o traço complexo, sobre a fatia sísmica representada na figura 3. Percebe-se que camadas com altas amplitudes seguidas de camadas de baixa amplitude foram unidas, formando uma única camada mais espessa.

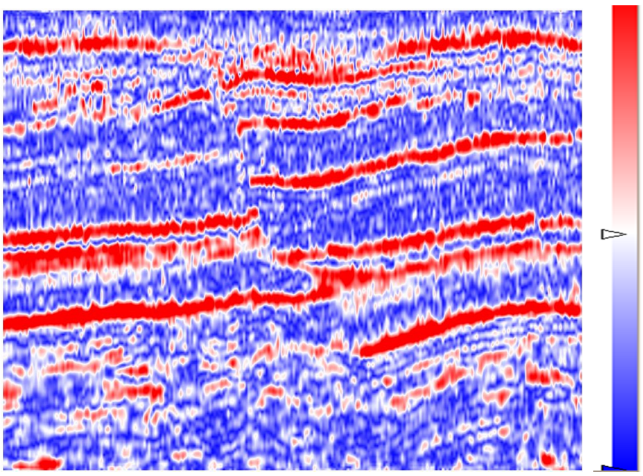

Figura 4 Envelope calculado sobre a seção sísmica representada na figura .

O atributo de fase instantânea é um bom indicador de falhas, pois este tende a realçar as descontinuidades do dado sísmico. Este atributo pode ser calculado através da equação

$$\emptyset(\boldsymbol{t}) = \boldsymbol{arc\ tg}\left(\frac{\boldsymbol{f^*(t)}}{\boldsymbol{f(t)}}\right) \tag{2.3}$$

onde $\emptyset(t)$ representa a fase instantânea calculada sobre o traço $f(t)$, conforme definido em [20].

A figura 5 representa a fase instantânea calculada sobre a amplitude mostrada na figura 3.

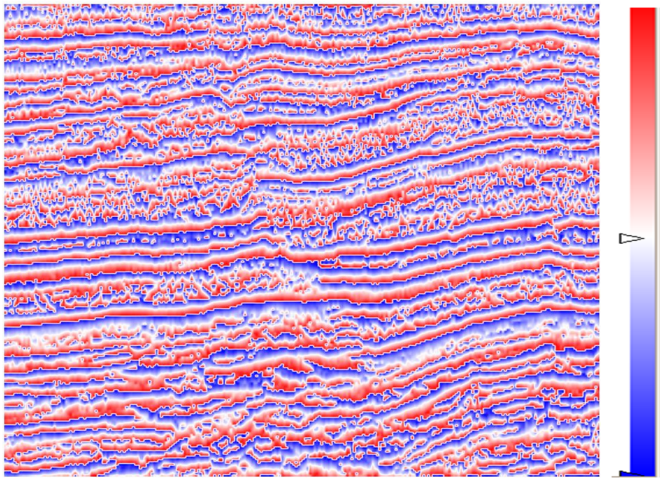

Figura 5 Fase calculada sobre seção sísmica representada na figura .

Outro atributo muito utilizado é a freqüência instantânea. Uma de suas maiores características é a de ser um indicador de áreas de fratura, pois estas fraturas tendem a aparecer em regiões de baixa frequência [19]. Outra característica é a indicação de hidrocarbonetos em regiões anômalas de baixa freqüência. A freqüência instantânea pode ser calculada pela fórmula

$$\boldsymbol{\omega(t) = \frac{d\phi(t)}{dt}} \tag{2.4}$$

onde $\omega(t)$ representa a freqüência instantânea e o segundo termo da equação representa a derivada da fase instantânea, conforme definido em [20].

A figura 6 mostra a frequência instantânea, calculada sobre a seção sísmica representada na figura .

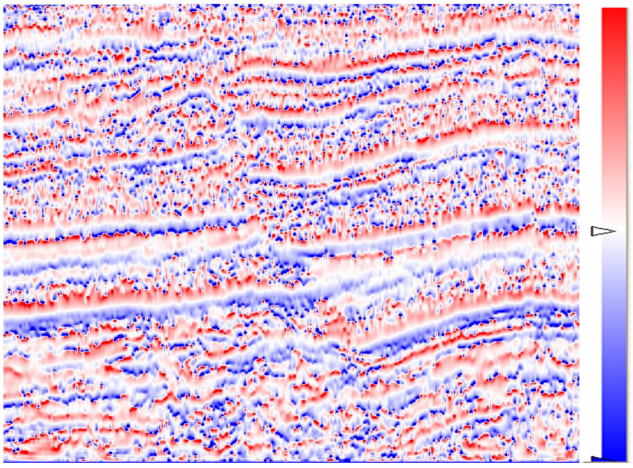

Figura 6 Frequência calculada sobre a seção sísmica representada na Figura .

Os tipos de atributos, aqui citados, são conhecidos como atributos 1D, pois levam em consideração apenas o traço sísmico, que possui variação no tempo (ou

profundidade). Existem os atributos 2D e 3D, que levam em consideração a vizinhança do traço. Neste trabalho não serão comentados os atributos 2D e 3D, pois estes não tem relevância para o que foi desenvolvido.

2.2. Transformada Wavelet

No dia-a-dia, sinais são usados para se comunicar, seja por meio da voz, por meio de gestos ou através da escrita. De acordo com Gesualdi [21], um sinal pode ser definido como uma função de uma ou mais variáveis, a qual vincula informações sobre a natureza de um fenômeno físico. No caso dos sinais definidos por funções de uma única variável, o sinal é denominado sinal unidimensional. Um exemplo deste tipo de sinal é a voz. O sinal representado por duas ou mais variáveis é chamado de sinal multidimensional. Um exemplo são as imagens, que depende de um par de coordenadas para localizar um pixel no plano e o valor do pixel.

As variáveis do sinal podem ser contínuas ou discretas, definindo funções contínuas ou discretas para representar o sinal. Em geral, no estudo de processamento de sinais, os sinais contínuos são transformados em sinais discretos. Dentre as possíveis transformações existentes, as mais utilizadas são as transformações de Fourier ([22], [23]) e *wavelet* ([23], [24], [25], [26]).

Neste trabalho será utilizada a transformada *wavelet* para a decomposição de sinais adquiridos de hidrofones/geofones. Uma forma usual de explicar a transformada *wavelet* é fazer um comparativo com a transformada de Fourier e explicitar as diferenças entre elas.

A transformada de Fourier tem por objetivo decompor um sinal em uma soma de senos e cossenos. Esta decomposição faz com que o domínio do sinal de entrada seja transformado do domínio do tempo para o domínio da frequência. Em sinais periódicos, a transformada de Fourier apresenta excelentes resultados, porém, em sinais não periódicos este tipo de transformada não se mostra muito adequado, pois com a decomposição de senos e cossenos a informação de tempo é perdida. Para tentar minimizar este problema, foi criada a transformada de Fourier Janelada [27].

Na transformada de Fourier Janelada, cria-se uma janela de um determinado tamanho e calcula-se a transformada de Fourier no trecho do sinal que pertence a esta janela. Ao alterar a posição da janela, calcula-se a nova transformada de Fourier para o sinal que se encontra dentro da nova janela. Desta forma, a relação entre o tempo e a frequência não é totalmente perdida e o resultado da decomposição passa a ser representado por um conjunto de transformadas de Fourier. O tamanho da janela pode ser alterado de acordo com as características do sinal que se está analisando, e do período em questão, dependendo se deseja um maior detalhamento ou não desta parte do sinal.

A transformada *wavelet* também decompõe um sinal em um conjunto de funções, porém estas muitas vezes são mais complexas que os senos e cossenos utilizados na transformada de Fourier. Estas funções devem obedecer alguns requisitos, tais como, a integral da função deve ser zero, a função deve ser bem localizada e deve possuir uma função inversa [23].

Na transformada de Fourier, uma pequena modificação na freqüência irá produzir mudanças em todo o domínio do tempo. Uma vantagem da transformada *wavelet* é que esta mantém a relação entre o tempo e a frequência, ou seja, uma pequena mudança no domínio da frequência gera pequenas modificações no tempo em pontos bem definidos. Esta relação é mantida através do suporte compacto definido na função *wavelet*.

Para o caso das transformadas discretas, a implementação da transformada de Fourier mais utilizada é a FFTW (*Fast Fourier Transform in the West*, [22]) desenvolvida nos laboratórios do *MIT*. Ela é chamada de *fast* pois nos testes de desempenho a que foi submetida, teve resultados melhores que a dos concorrentes de código aberto.

Computacionalmente, a transformada *wavelet* é mais rápida que a transformada de Fourier. A ordem de complexidade da *FFTW* é de $O(n \cdot log_2(n))$, enquanto que na transformada *wavelet* a ordem de complexidade diminui para $O(n)$, como mostra [23]. Por isso, a transformada wavelet discreta também é conhecida como *Fast Wavelet Transform*.

As funções de escala e dilatação definidas na transformada *wavelet* formam um conjunto de bases ortogonais e, por isso, são conhecidas como funções bases. Já as funções ortogonais às funções bases, e que não pertencem ao espaço gerado pelas funções bases, são chamadas de funções *wavelet*. Estas funções são capazes

de representar tanto a escala como as translações no espaço. A função que representa a translação no espaço de maior resolução é conhecida como *"wavelet mãe"*, enquanto que todas as funções de translações nos espaços de menor resolução são chamadas de *"wavelets filhas"*.

As *wavelets* mais conhecidas são: a *wavelet* de Haar ([1], [2], [6]) por sua simplicidade e por ser de fácil implementação, a de Daubechies [6], por possuir o filtro de decomposição simétrico ao filtro de recomposição.

A *wavelet* de Haar consiste em fazer uma sucessão de médias e diferenças. Utilizando o exemplo descrito em [1], seja uma "imagem" unidimensional, de quatro pixels de resolução, dados pelos coeficientes $[\,9\ \ 7\ \ 3\ \ 5\,]$. Pode-se obter uma imagem de menor resolução fazendo a média dos coeficientes da imagem dois a dois. A imagem resultante, seria uma imagem de menor resolução, cujos coeficientes tem valores $[\,8\ \ 4\,]$.

Somente com esses dois coeficientes, não é possível reconstruir a imagem original. Para resolver este problema, armazenam-se, também, os chamados coeficientes de detalhes. Estes coeficientes são calculados como sendo a diferença entre o primeiro elemento, do par utilizado para calcular a média e a própria média. A decomposição acima resulta em coeficientes de detalhe com valores $[\,1\ \ -1\,]$. O resultado é a imagem $[\,8\ \ 4\ \ 1\ \ -1\,]$. Diz-se que esta imagem possui um nível de decomposição.

A decomposição pode ser feita até que exista apenas um coeficiente proveniente de médias. No caso da imagem utilizada como exemplo, o resultado final seria uma imagem com valores $[\,6\ \ 2\ \ 1\ \ -1\,]$ e o nível de decomposição seria o segundo nível.

A reconstrução da imagem segue o caminho inverso. Para reconstruir um par da imagem do passo anterior, deve-se utilizar a média que o par gerou e o coeficiente de detalhe gerado. Para reconstruir o primeiro elemento do par, basta fazer a soma dos coeficientes (de média e de detalhe). O segundo elemento do par é calculado fazendo a diferença entre os coeficientes.

O ponto principal da *wavelet* de Haar é a reordenação dos dados ao fazer a média e a diferença. Com isso, os coeficientes agrupados na parte inicial da imagem representam a imagem original com uma menor resolução.

Por ser aplicada a pares de coeficientes, a *wavelet* de Haar possui uma limitação quanto ao número de coeficientes que a "imagem" deve ter para que a

mesma seja aplicada. A cada passo da decomposição, a resolução da imagem gerada é reduzida a metade. Para que o limite da decomposição seja uma imagem (de baixa resolução) de um único pixel, é necessário que a imagem original tenha tamanho igual a uma potência de dois. O total de níveis de decomposição que a imagem pode chegar é dado pelo valor da potência.

A transformada *wavelet* é largamente utilizada na compressão de imagens. Neste trabalho será descrito apenas como aplicar a transformada *wavelet* de Haar para a decomposição de uma imagem cujos lados têm as mesmas medidas e são potência de dois.

A transformada de Haar 2D consiste em aplicar a decomposição, que é 1D, em ambas as direções. Existem dois tipos de decomposição 2D utilizando a transformada de Haar: a decomposição padrão e a não-padrão.

Na decomposição não padrão, aplica-se a transformada em uma direção até atingir o último nível de decomposição. Em seguida aplica-se a transformada na outra direção até o último nível. A figura 7 mostra a decomposição sendo aplicada em todas as linhas da imagem. A imagem mais a esquerda é a imagem original de 512 x 512 pixels. A segunda imagem representa um nível de decomposição aplicado em todas as linhas da imagem original. Por último, encontra-se a imagem com nove níveis de decomposição.

Figura 7 Decomposição de Haar padrão aplicada as linhas da imagem.

Na figura 8 é aplicada a transformação em todas as colunas da imagem resultante da decomposição das linhas mostrada ultima imagem da figura 7.

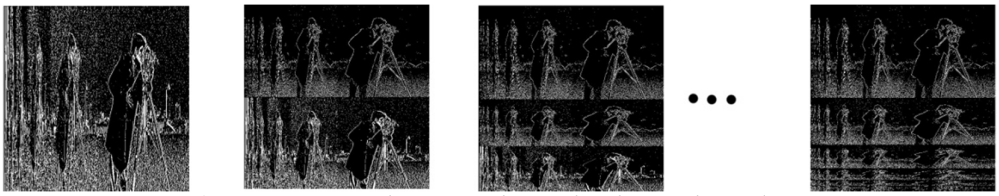

Figura 8 Decomposição de Haar padrão aplicada as colunas da imagem.

Para reconstruir a imagem original, deve-se aplicar a recomposição no sentido inverso do que foi aplicada a decomposição, ou seja, se a decomposição foi feita primeiramente nas linhas e em seguida nas colunas, deve-se reconstruir primeiro as colunas e por último as linhas.

Na decomposição não padrão a decomposição é feita em duas etapas. Primeiro aplica-se a decomposição em um sentido e, em seguida, no outro sentido. Neste tipo de decomposição, a cada nível de transformação obtém-se uma imagem de baixa resolução da imagem original. A figura 9 mostra como é feita a decomposição não padrão. A imagem (a) representa a imagem original. O resultado do primeiro nível de decomposição está representado pela imagem (b). A imagem que aparece entre (a) e (b) é somente para ilustrar os passos dessa decomposição. A imagem (c) representa dois níveis de decomposição, enquanto que a imagem (d) representa todos os níveis de decomposição possíveis para esta imagem.

Uma vantagem da decomposição não padrão é que a cada nível de decomposição, obtém-se uma representação em menor resolução da imagem original. Na figura 9 estas representações encontram-se no quadrante inferior esquerdo.

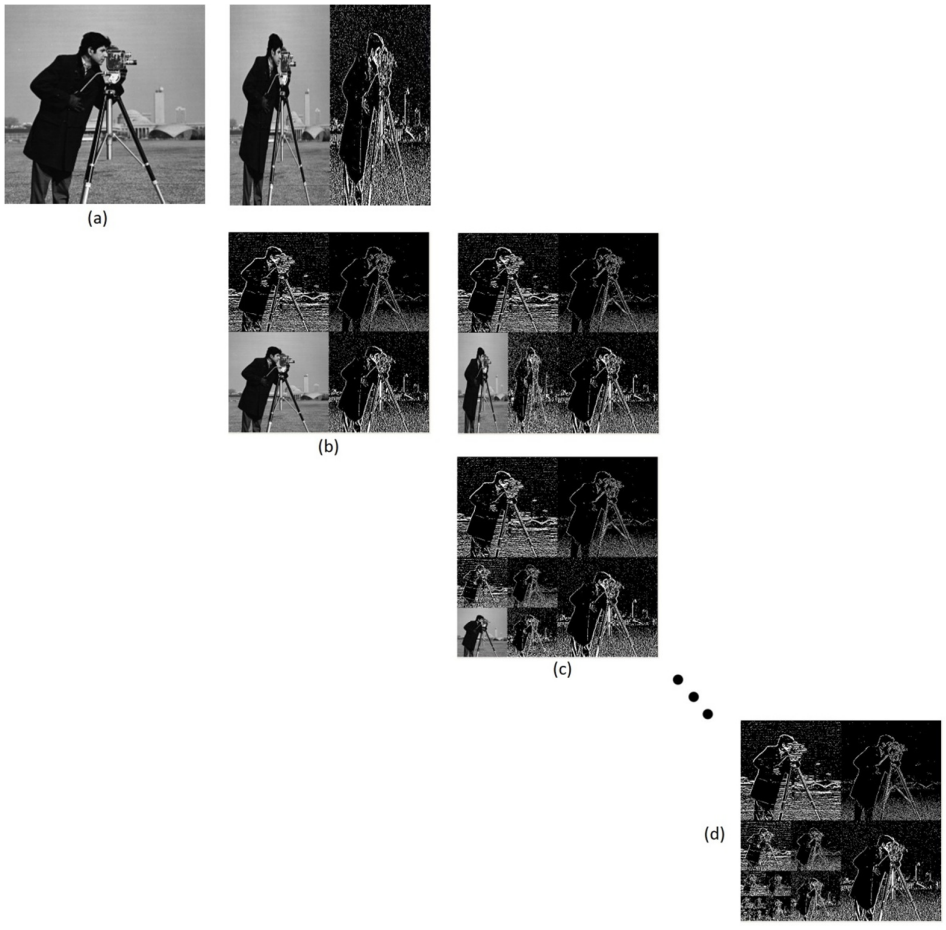

Figura 9 Decomposição de Haar não padrão.

Estas imagens parciais podem ser utilizadas para fazer uma pré-visualização da imagem original, em situações onde a mesma não precisa ser representada com perfeição, como por exemplo, as miniaturas que os *softwares* de visualização de imagem apresentam no momento de seleção de uma imagem para manipulação, as chamadas *thumbnails*. A figura 10 mostra algumas resoluções da imagem decomposta a cima.

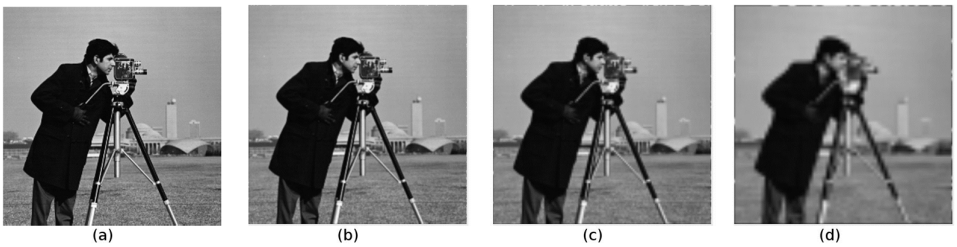

Figura 10 Diferentes níveis de resolução da imagem. (a) representa a imagem original com 512 x 512 pixel. (b) representa um nível de decomposição, sendo a imagem de 256 x 256 pixels. (c) representa a imagem com 128 x 128 pixels. (d) representa a imagem com 64 x 64 pixels.

Toda *wavelets* tem suporte compacto de energia. Em matemática, ao falar em suporte de uma função, refere-se ao menor subconjunto fechado do domínio da função, onde esta não é nula. Já o termo compacto, em topologia representa o quão pequeno é um determinado conjunto. Todo conjunto finito é compacto. Logo, uma *wavelet* com suporte compacto de energia representa uma função que possui uma pequena área de atuação, sendo nula em todo o resto do domínio.

Esta característica é de extrema importância para a aplicação da decomposição *wavelet* em séries temporais, pois é o suporte compacto que define que a transformada *wavelet* mantém a relação entre o domínio do tempo e da frequência ([23]).

Dentre esta família de *wavelets*, existem as que apresentam bases biortogonais e as que apresentam bases ortonormais. A vantagem das *wavelets* que apresentam bases biortogonais é que os filtros de decomposição e o filtro de reconstrução podem ser simétricos. O mesmo não ocorre para as *wavelets* de bases ortonormais, com exceção da *wavelet* de Haar.

2.3. Quantização

É chamada de quantização a técnica que visa mapear um determinado conjunto de dados em outro conjunto de interesse. Em geral, essa técnica é utilizada quando se deseja representar um sinal contínuo por um sinal discreto.

A quantização pode ser escalar ou vetorial. Na quantização escalar, cada elemento do sinal de entrada é analisado separadamente e levado em um valor quantizado. A figura 11 mostra um sinal sendo quantizado através da quantização escalar.

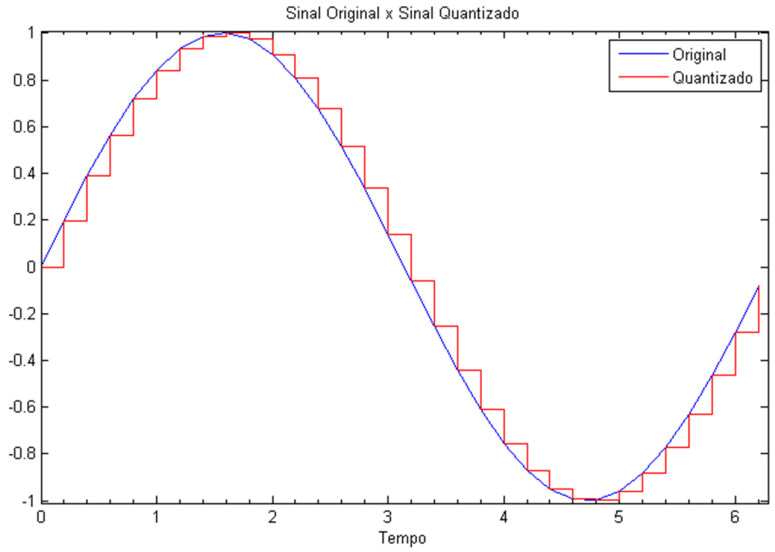

Figura 11 Quantização escalar de um sinal.

Na quantização vetorial os valores são agrupados em vários vetores de mesmo tamanho. Cada vetor é analisado e cria-se uma base de quantização (também chamada de *Codebook*) que é formado por vários vetores de mesmo tamanho e de diferentes configurações. A partir desta base, quantizam-se os vetores originais ([28]).

Ao quantizar um sinal, é inserido um erro, pois a representação do mesmo passa a ser aproximada e não exata. No processo de quantização, a informação original é descartada e se guarda apenas o necessário para poder fazer a operação inversa da quantização. No caso da quantização escalar, é armazenado uma tabela que mantém uma relação do intervalo do dado original e o valor quantizado das amostras desse intervalo. Para desquantizar o dado, basta substitui o valor

quantizado por um valor do intervalo. Em geral, este valor é representado pelo ponto médio do intervalo.

A figura 12 mostra o exemplo de uma imagem sendo quantizada através da quantização vetorial, e em seguida, sendo desquantizada.

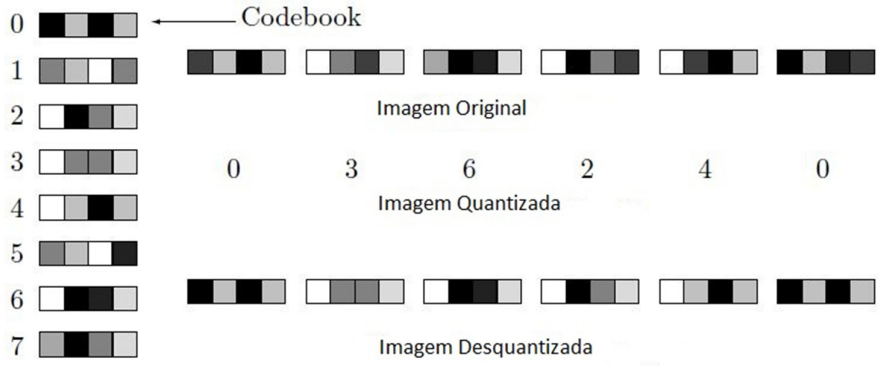

Figura 12 Quantização vetorial. Adaptado de [28].

Na figura 12 percebe-se que a imagem original de vinte e quatro pixels, foi segmentada em seis pedaços de quatro pixels consecutivos. Tendo como base de quantização (*Codebook*) os vetores representados à esquerda da imagem, foi possível quantizar a imagem gerando os códigos [0 3 6 2 4 0]. Aplicando o processo inverso (substituindo os códigos pelo vetor de pixels), obtém-se a imagem desquantizada. É fácil ver que a imagem desquantizada é diferente da imagem original, mostrando que um erro foi introduzido no processo de quantização/desquantização.

2.4. Algoritmos de Compressão

Os algoritmos de compressão têm por objetivo reduzir as redundâncias no dado que está sendo comprimido e com isso reduzir o número de *bytes* necessários para armazenar este dado.

Existem dois tipos de compressão: a compressão sem perda e a compressão com perda. Na compressão sem perda, ao descomprimir o dado, este volta para a sua forma de origem, ou seja, nenhum erro é adicionado ao dado. Já a compressão com perda adiciona um erro ao dado. Porém, na compressão com perda, as taxas de compressão tendem a serem maiores que as taxas apresentadas na compressão sem perda.

Para saber o quanto que o dado foi comprimido é necessário definir algumas grandezas para comparação entre o dado original e o dado comprimido. A primeira a ser definida é a razão de compressão. Esta grandeza é definida como sendo a razão entre o dado comprimido e o dado original, como mostra a equação abaixo:

$$razão\ de\ compressão = \frac{tamanho\ do\ dado\ comprimido}{tamanho\ do\ dado\ original} \tag{2.5}$$

Quando a razão de compressão é menor que um quer dizer que o dado comprimido é menor que o dado original. Há algumas situações indesejadas onde a razão de compressão é maior que um, indicando que o dado gerado é maior que o dado original.

A razão de compressão também pode ser representado na forma percentual. Por exemplo, se a razão de compressão foi de 0,4, este pode ser representado como sendo 40%, e quer dizer que o dado comprimido representa 40% do dado original. Outra forma de representar a razão de compressão é dizer o quanto que este é menor que o dado original, para isso basta representar a grandeza como sendo 1 – razão de compressão. Este é o chamado fator de compressão. No exemplo a cima, teríamos que o fator de compressão seria de 60% do dado original, ou seja, que o dado comprimido é 60% menor que o dado original.

Outra grandeza utilizada na compressão de dados é a taxa de compressão. Neste caso representa-se o tamanho de um dado processado em relação ao tamanho do dado original. Quando escreve-se uma taxa de compressão de 4:1 pretende-se dizer que no espaço utilizado pelo dado original é possível ter quatro dados comprimidos, ou seja, que o fator de compressão foi 25% ou 0.25. A taxa de compressão pode ser calculada como sendo:

$$taxa\ de\ compressão = \frac{tamanho\ do\ dado\ comprimido}{tamanho\ do\ dado\ original} \tag{2.6}$$

Existem vários algoritmos de compressão de dados, tanto com perda como sem perda. Dentre os algoritmos com perda, podemos destacar o JPEG2000 para imagens, o mp3 para arquivos de áudio e MPEG-4 para arquivos de vídeos. Já nos algoritmos sem perda, destacam-se os algoritmos de Huffman ([28], [29]), *Run-Length Encode* (RLE) ([28]), zip ([28]) e o método de codificação aritmética ([28]).

Neste trabalho não foram utilizados algoritmos de compressão com perda. Toda a perda de informação obtida no trabalho foi em função da quantização. Por isso, serão descritos a seguir apenas os algoritmos de Huffman e RLE, utilizados neste trabalho. Para maiores informações sobre os outros métodos é recomendada a leitura de [28].

2.4.1. Algoritmo de Huffman

Este método se baseia na probabilidade de ocorrência de um determinado símbolo. Ao símbolo que tiver uma maior freqüência de ocorrência será criada uma representação que seja a menor possível, enquanto que, para os que possuem uma menor freqüência de ocorrência será criada uma representação maior.

O Algoritmo de Huffman é um algoritmo de duas passadas. Na primeira passada, são calculadas as frequências em que os símbolos aparecem e é criada a codificação. Já na segunda passada esses símbolos são codificados de acordo com a representação que foi criada para os mesmos.

No primeiro passo, ao calcular as frequências que os símbolos aparecem, estes são organizados em uma árvore binária, em que o nó mais acima representa o símbolo de maior frequência e os nós mais abaixo representam os símbolos de menor frequência. Isto é, os símbolos são organizados em ordem decrescente de probabilidade de ocorrência. Feito isso, unem-se os dois últimos símbolos e somam-se as suas probabilidades. Os símbolos são reorganizados de acordo com as novas probabilidades e agrupam-se os dois símbolos de menor probabilidade. Repete-se este processo até que a probabilidade de ocorrência do símbolo criado seja um.

Por exemplo, sejam, os símbolos e as probabilidades de ocorrência destes símbolos definidos por $\mathrm{p1} = 0{,}20$, $\mathrm{p2} = 0{,}18$, $p3 = 0{,}10$, $p4 = 0{,}10$, $p5 = 0{,}10$, $p6 = 0{,}06$, $p7 = 0{,}06$, $p8 = 0{,}04$, $p9 = 0{,}04$, $p10 = 0{,}04$, $p11 = 0{,}04$, $p12 = 0{,}03$ e $p13 = 0{,}01$ (exemplo definido em [29]). Ao combinar os símbolos $p12$ e $p13$, é criado um novo símbolo que tem como probabilidade a soma das probabilidades de $p12$ e $p13$. Este novo símbolo será chamado de $A = 0{,}04$. Como a probabilidade de A ainda é menor, ou igual, que a dos outros símbolos, este novo símbolo entra no final da lista. Repetindo a

combinação entre os elementos de menor probabilidade, tem-se que os elementos $p11$ e A serão combinados, formando um novo símbolo B de probabilidade 0,08. Neste momento, a lista deve ser reordenada, pois a probabilidade de B é maior que a de alguns símbolos.

A nova lista fica sendo: $p1 = 0{,}20, p2 = 0{,}18, p3 = 0{,}10, p4 = 0{,}10,$ $p5 = 0{,}10, B = 0{,}08,$ $p6 = 0{,}06,$ $p7 = 0{,}06,$ $p8 = 0{,}04,$ $p9 = 0{,}04$ e $p10 = 0{,}04$. Os passos de combinar os dois símbolos de menor probabilidade e de reorganização da lista devem ser repetidos até que exista apenas um símbolo de probabilidade igual a um.

Caso sejam feitos todos os passos descritos a cima, e nomeando os novos símbolos na ordem alfabética, o resultado será a árvore mostrada na figura 13.

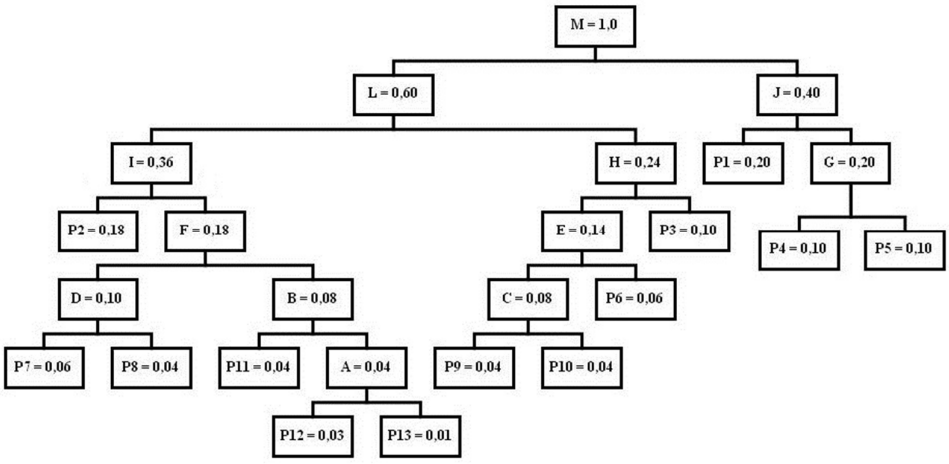

Figura 13 Arvore binária gerada na codificação de Huffman

Para saber como um determinado símbolo será representado pela codificação de Huffman, basta percorrer a árvore gerada atribuindo zeros e uns para cada nó que se passa, sendo que ao descer um nível, o nó a esquerda recebe zero e o nó a direita recebe um.

No exemplo a cima, a representação dada ao $p7$ será 00100. A tabela 1 a seguir mostra todas as representações referentes à árvore da figura 13. Percebe-se que os elementos que tinham uma maior probabilidade de ocorrência possuem uma menor representação ([29]).

Símbolo	Probabilidade	Representação
$p1$	0,20	10
$p2$	0,18	000
$p3$	0,10	011
$p4$	0,10	110
$p5$	0,10	111
$p6$	0,06	0101
$p7$	0,06	00100
$p8$	0,04	00101
$p9$	0,04	01000
$p10$	0,04	01001
$p11$	0,04	00110
$p12$	0,03	001110
$p13$	0,01	001111

Tabela 1 Codificação de gerada para a árvore representada na figura 13.

A tabela 1 é conhecida como tabela de codificação ou dicionário de codificação. Esta tabela não é única, de acordo com a reorganização feita nos símbolos que forem combinados, a tabela pode ser alterada. Por isso, esta deve ser armazenada junto com o dado que se pretende codificar.

Supondo que se queira codificar o trecho da mensagem representado pela sequência de símbolos $p7\ p6\ p1\ p4\ p2$, cujas frequências e codificações foram calculadas acima. Este trecho da mensagem seria codificado como sendo 00100010110110000.

Se fosse utilizada a menor codificação fixa para representar os treze símbolos, seriam necessários quatro bits por símbolo, visto que existem treze elementos e para representar o décimo terceiro seriam necessários quatro bits. Para representar o trecho da mensagem acima, seriam necessários vinte bits, pois existem cinco elementos na mensagem com quatro bits para cada elemento. No entanto, ao utilizar a codificação de Huffman, a mesma mensagem é representada com apenas dezessete bits, o que representa um fator de compressão de 15%.

Para decodificar a mensagem, basta seguir os bits da mensagem um a um, desde a raiz até chegar a uma folha da árvore. Ao chegar à folha, os bits percorridos devem ser substituídos pelo símbolo que ele representa. Repete-se o processo até que todos os bits tenham sido decodificados para os símbolos originais.

Existem variações do algoritmo de Huffman, como por exemplo, o algoritmo adaptativo. Neste caso, é criada uma árvore binária vazia, e conforme as estatísticas dos símbolos vão sendo calculadas a árvore é atualizada. A

vantagem do algoritmo adaptativo é que este não necessita de duas passadas no dado.

Para o algoritmo tradicional, a complexidade do algoritmo é da ordem de $O(n \cdot logn)$. O grande sucesso do algoritmo de Huffman se deve ao fato de ser um método que gerar uma codificação livre de prefixos, ou seja, para quaisquer dois códigos gerados, um não é prefixo para o outro.

2.4.2. Run Length Encode (RLE)

A idéia básica do algoritmo RLE ([28]) é levar em consideração o número de vezes que um determinado elemento aparece consecutivamente. Ao percorrer-se o dado a ser compactado, caso exista uma repetição de um determinado elemento, o conjunto de repetições será trocado pelo número de vezes que o elemento aparece seguido do valor do próprio elemento. Ou seja, se o elemento *a* aparece *n* vezes consecutivas, os *n* elementos *a* serão substituídos pela dupla *n,a.*

Existem várias implementações do RLE. A mais simples é codificar todos os elementos, ou seja, mesmo que não exista repetição, um elemento será substituído por duas informações: o número de repetições seguido do valor do elemento. Neste tipo de implementação, caso não existam muitas repetições, o dado será pouco compactado e poderá até ter seu tamanho aumentado. Para evitar este tipo de problema, existe uma implementação do RLE em que somente os elementos repetidos são codificados. Neste caso, é necessário um indicador de que houve uma codificação. Pode-se utilizar, por exemplo, o caractere especial @ para indicar quando um elemento foi codificado.

O RLE foi desenvolvido para ser utilizado nos antigos aparelhos de fax. Neste tipo de aplicação, desejava-se compactar uma imagem binária, com muitos blocos brancos e alguns pontos pretos.