3
Métodos e Modelos

3.1
Método de Seleção de Variáveis

3.1.1
Introdução

Na maioria das aplicações reais de previsão, as bases de dados contêm um grande número de atributos ou variáveis, muitas delas introduzidas para obter uma melhor representação do problema. Entretanto, na maioria dos casos, grande parte destas variáveis é irrelevante e/ou redundante. Deste modo, um problema comum nestas aplicações reais é a seleção das características ou variáveis mais relevantes do ponto de vista do objetivo final, dentre todos os atributos da base de dados. O modelo de tratamento descrito nesta seção tem essa missão, ou seja, reduzir e otimizar a base de dados que servirá de entrada para os modelos das seções subseqüentes 3.2 e 3.3, na tentativa de alavancar a eficiência desses modelos.

Existem inúmeras técnicas para a tarefa de seleção de variáveis. Dentre os algoritmos de seleção de variáveis, que são independentes do modelo (Model Free) – possuem capacidade de escolha de variáveis em menor tempo e a um menor custo computacional que os algoritmos dependentes do modelo – pode-se citar: Correlação Cruzada, Autocorrelação, Estimador por Mínimos Quadrados (LSE – Least Squares Estimator) e SIE (Single Input Effectiveness). Neste trabalho foi utilizado o LSE, tendo em vista que: tanto a Correlação Cruzada como a Autocorrelação são próprias para medir relações lineares entre as variáveis de entrada e a variável de saída; e o SIE, além de também partir da premissa de relações lineares entre entradas e saída, já foi usado em trabalho anterior semelhante [62], com resultados ruins.

3.1.2
Método do Estimador por Mínimos Quadrados (LSE)

O Estimador por Mínimos Quadrados é um método que busca expressar o comportamento da variação $\Delta \boldsymbol{y}$ da variável de saída $\boldsymbol{y}$ (vetor) em função das variações $\Delta \boldsymbol{x}$ das diversas variáveis de entrada $\boldsymbol{x_i}$ (vetores) do sistema, por meio dos diferenciais de primeira ordem das quantidades, ou seja, busca estimar os coeficientes b na relação $\Delta \boldsymbol{y} = \Delta \boldsymbol{x}.b$. Desta forma, consegue-se linearizar possíveis relações não-lineares e resolver o problema por meio de uma regressão linear multivariada, determinando os coeficientes desta regressão por meio de um método de mínimos quadrados.

3.1.2.1
Descrição

Seja um sistema de n entradas e uma saída; o método LSE calcula a importância da i-ésima variável de entrada $\boldsymbol{x_i}$ estimando o i-ésimo parâmetro b_i da função F (equação 3.1), que descreve a variação da variável de saída $\Delta \boldsymbol{y}$ em relação à variação de cada i-ésima variável de entrada $\Delta \boldsymbol{x_i}$ sobre o conjunto completo de dados.

$$F = \Delta \boldsymbol{y} = b_1 \Delta \boldsymbol{x_1} + b_2 \Delta \boldsymbol{x_2} + \ \ldots \ + b_n \Delta \boldsymbol{x_n} \tag{3.1}$$

Os componentes do vetor $\Delta \boldsymbol{y}$ são obtidos subtraindo os valores da variável de saída nos padrões da base de dados em combinações duas a duas, e os componentes do vetor $\Delta \boldsymbol{x_i}$ são obtidos subtraindo os valores correspondentes do vetor variável de entrada $\boldsymbol{x_i}$ nos padrões da base de dados em combinações duas a duas.

Da equação 3.1, pode-se dizer que cada parâmetro b_i representa a importância da i-ésima variável de entrada com respeito à variável de saída, no sentido estatístico. O cálculo dos parâmetros b_i é feito mediante o algoritmo do Estimador por Mínimos Quadrados (LSE). A seção seguinte descreve com mais detalhes este método.

3.1.2.2
Algoritmo do LSE

O algoritmo do Estimador por Mínimos Quadrados determina a importância de cada variável baseado na teoria a seguir [63].

Seja uma função diferenciável *y* que descreve um sistema de *n* entradas e uma saída:

$$y=f(x_1, x_2, x_3, ..., x_n) \text{ onde } [x_1, x_2, x_3, ..., x_n]^T \in [0,1]^n,$$

havendo disponível um conjunto de *p* pares de dados amostrais (padrões) desta função:

$$[x_1^j, x_2^j, x_3^j, ..., x_n^j, y^j]^T, \text{ onde } j=1, 2, ..., p.$$

Quaisquer dois valores de saída, tais como o valor de saída para o j-ésimo par de dados y_j e valor de saída para o k-ésimo par de dados y_k, podem ser aproximados usando a seguinte Expansão em Série de Taylor em torno de um ponto fixo arbitrário $[X_1, X_2, X_3, ..., X_n]^T$:

$$y_j=f(X_1, X_2, X_3, ...,X_n)+ \sum_{i=1}^{n}\left[\left(\left.\frac{\partial f}{\partial x_i}\right|_{x_i=X_i}\right).(x_i^j - X_i)\right] + r_j \qquad (3.2)$$

$$y_k=f(X_1, X_2, X_3, ...,X_n)+ \sum_{i=1}^{n}\left[\left(\left.\frac{\partial f}{\partial x_i}\right|_{x_i=X_i}\right).(x_i^k - X_i)\right] + r_k \qquad (3.3)$$

Nas equações 3.2 e 3.3, r_j *e* r_k são resíduos de alta ordem e podem ser ignorados sem risco de perder muita informação se $\left|(x_i^j - X_i)\right| \le 1$ e $\left|(x_i^k - X_i)\right| \le 1$, para todas as *i* variáveis.

Subtraindo a equação 3.2 da equação 3.3, de modo a obter a variação da variável de saída Δy em relação à variação de cada i-ésima variável de entrada Δx_i, sobre o sistema, tem-se:

$$y_j - y_k = \sum_{i=1}^{n}[b_i(x_i^j - x_i^k)] \quad , \qquad (3.4)$$

$$\text{onde } b_i = \sum_{i=1}^{n}\left.\frac{\partial f}{\partial x_i}\right|_{x_i=X_i} \quad , \qquad (3.5)$$

pela qual a função original é aproximada por uma função linear. Esse tipo de aproximação é utilizado extensamente em regressão não linear [64]. A equação 3.4

pode ser escrita em função do conjunto de dados da seguinte maneira: se a base de dados contém *p* pares de dados, existem $m = C_2^p$ (número de combinações dos pares de dados dois a dois) "vetores variação", obtidos subtraindo os valores das variáveis de saída e de entrada relativos ao par de dados *j* dos valores das variáveis de saída e de entrada relativos ao par de dados *k*, respectivamente. Portanto, estes vetores variação terão a seguinte forma:

$$[x_1^j - x_1^k, x_2^j - x_2^k, ..., x_n^j - x_n^k, y^j - y^k]^T$$

O número de vetores variação $m = C_2^p$ pode ser um número muito grande, portanto, somente q ($<< m$) vetores variação são aleatoriamente selecionados. Deste modo, a equação 3.4, que descreve a variação da variável de saída $\Delta \boldsymbol{y}$ em relação à variação de cada i-ésima variável de entrada $\Delta \boldsymbol{x_i}$ sobre todo o conjunto de dados, pode ser reescrita na forma matricial como segue:

$$\Delta \boldsymbol{y} = \Delta \boldsymbol{x}.\mathrm{b} \qquad (3.6)$$

Onde $\Delta \boldsymbol{y}$,$\Delta \boldsymbol{x}$ e $\boldsymbol{b}$ têm dimensões $qx1$, qxn e $nx1$, respectivamente. *b* é um vetor desconhecido, cujos elementos são os valores dos coeficientes b_i (equação 3.5).

A equação 3.6 também pode ser escrita como:

$$\Delta \boldsymbol{y} = b_1 \Delta \boldsymbol{x_1} + b_2 \Delta \boldsymbol{x_2} + ... + b_n \Delta \boldsymbol{x_n} \qquad (3.7)$$

Onde $\Delta \boldsymbol{y}$, $\Delta \boldsymbol{x_i}$, b_i têm dimensão $qx1$, $qx1$ e $1 \mathrm{x} 1$, respectivamente.

Das equações 3.6 e 3.7, encontra-se que cada elemento do vetor *b,* isto é, cada b_i, representa a taxa de variação da variável de saída $\boldsymbol{y}$ em relação à variação de cada variável de entrada $\boldsymbol{x_i}$, sobre o conjunto completo de dados. Por conseguinte, o valor de b_i representa a importância da correspondente entrada com respeito à saída, no sentido estatístico [63].

Em geral, na equação 3.6, se $q > n$, não existe solução exata ou única para *b* (sistema sobredeterminado). Para solucionar o problema, utiliza-se a fórmula da pseudo-inversa [64-66] para encontrar *b**, que é o Estimador por Mínimos Quadrados de *b*:

$$b^* = (\Delta \boldsymbol{x}^T \,.\, \Delta \boldsymbol{x})^{-1} \,.\, \Delta \boldsymbol{x}^T \,.\, \Delta \boldsymbol{y} \qquad (3.8)$$

3.1.2.3
Importância das Variáveis de Entrada

Na equação 3.7, cada valor de b_i representa o grau de importância da correspondente variável de entrada x_i com respeito à saída. Os valores de b_i podem ser positivos ou negativos. Deste modo, define-se o termo impo(x_i) para representar o grau de importância de x_i em relação a variável de saída y:

$$impo(x_i) = \frac{|b_i|}{\sum_{j=1}^{n} |b_j|} \tag{3.9}$$

O que implica que : $$\sum_{i=1}^{n} impo(x_i) = 1 \tag{3.10}$$

Sendo assim, as variáveis $\boldsymbol{x_i}$ que apresentarem os maiores graus de importância serão as variáveis selecionadas. A maneira de se definir quais são os maiores graus de importância fica a critério do usuário do método. Por exemplo, pode-se definir como sendo todos os graus maiores que um determinado valor, ou como sendo os *k* maiores graus, ou todos os graus maiores que a média dos graus.

Cabe ressaltar que, se os vetores de entrada $[x_1, x_2, x_3, ..., x_n]^T$ em um conjunto de dados excedem o intervalo $[0,1]^n$, então a expansão da série de Taylor não pode ser usada. Neste caso é aplicada uma função de normalização para normalizar todos os vetores de entrada em [0,1], e então o algoritmo descrito acima pode ser utilizado.

3.2
Redes Neurais Artificiais

3.2.1
Histórico

O cérebro humano possui características desejáveis em qualquer sistema artificial. Como exemplo, pode-se citar sua capacidade para lidar com informações inconsistentes e/ou probabilísticas, sua alta flexibilidade de adaptação a situações aparentemente pouco definidas, sua tolerância a falhas, entre outras. Todas estas características mencionadas despertaram o interesse de

pesquisadores, que na década de 80 intensificaram suas linhas de estudo na área de inteligência computacional com o uso da computação intensiva.

No entanto, o aparecimento da neuro-computação ocorreu bem antes, na década de 40, com o primeiro modelo artificial de um neurônio biológico. Em 1943, Warren McCulloch, psiquiatra e neuroanatomista, e Walter Pitts, matemático, desenvolveram uma máquina inspirada no cérebro humano e um modelo matemático de neurônio biológico artificial denominado *Psychon* [67]. Entretanto, este modelo não era capaz de desempenhar um de seus principais requisitos: o aprendizado.

Em 1951, Marvin Minsky criou o primeiro neurocomputador chamado *Snark* . A partir de um ponto de partida, o *Snark* operava bem, ajustando os seus pesos automaticamente. Apesar de não ter executado uma função de processamento de informação relevante, serviu como "molde" para futuras estruturas.

Em 1958, Frank Rosenblatt e Charles Wightman, juntamente com outros pesquisadores, desenvolveram o primeiro neurocomputador bem sucedido [68]. Estes pesquisadores são considerados como os fundadores da neurocomputação, devido à importância de seus trabalhos para essa linha de pesquisa, muito próxima da forma como existe atualmente. Seus estudos sustentaram os modelos do tipo perceptron (redes de um nível) e MLP (Perceptrons de múltiplas camadas). O objetivo inicial era aplicar a modelagem do tipo *Perceptron* para o reconhecimento de padrões.

Os modelos baseados no *Perceptron* sofreram graves críticas. Na obra: *"Introduction to computacional geometry",* Minsky e Papert mostraram matematicamente que estes modelos, na forma como estavam definidos, não eram capazes de aprender a função lógica do “OU Exclusivo” (XOR) [69]. A função XOR possui padrões de valores de entrada e saída cuja associação não poderia ser aprendida pelos modelos baseados em Perceptrons. Esta constatação impactou negativamente as pesquisas que vinham sendo realizadas sobre este assunto nas décadas de 60 e 70.

A partir dos anos 80, os estudos com redes neurais tomaram um impulso revolucionário. Em 1982, John Hopfield, físico mundialmente conhecido, criou um tipo de rede diferente daquelas fundamentadas no *Perceptron* [70]. Neste modelo a rede apresentava conexões recorrentes (ou seja, o sinal não se propagava

exclusivamente para frente) e baseava-se em um aprendizado não supervisionado com a competição entre os neurônios.

Em 1986, o reaparecimento das redes baseadas em *Perceptrons* foi possível graças à teoria de redes em multinível (MLP) treinadas com o algoritmo de aprendizado por retropropagação *(Backpropagation)* desenvolvido por Rummelhart, Hinton e Williams [71]. Além disso, vale lembrar que a década de 80 foi marcada pelo desenvolvimento de computadores cada vez mais potentes e velozes, que permitiram melhores simulações das redes neurais. Neste período também foram desenvolvidos modelos matemáticos que permitiam solucionar o problema do XOR [72].

A partir de então, um contexto favorável foi criado para o desenvolvimento das pesquisas em neurocomputação:

- (1987): acontece a primeira conferência de redes neurais, a IEEE *Internacional Conference on Neural Networks* em São Francisco - Criou-se ainda a INNS *(International Neural Networks Society).*
- (1989): fundação do INNS *Journal*.
- (1990): criação do *Neural Computation* e do IEEE *Transactions on Neural Networks.*

3.2.2
Estrutura do Neurônio

Para começar a falar de redes neurais, o ponto de partida é definir o que são e como se constituem as suas unidades básicas. Uma rede neural é um sistema computacional constituído por unidades conhecidas como neurônios. Os neurônios são elementos processadores interligados, trabalhando em paralelo para desempenhar uma determinada tarefa.

Como já foi dito anteriormente, o primeiro modelo de neurônio artificial foi o criado por McCulloch & Pitts, em 1943. A partir deste, os neurônios artificiais evoluíram e propiciaram, como estrutura básica, o aparecimento de vários modelos, tais como, *Perceptron*, *Adaline* e *Multilayer perceptron*. Este último é o modelo usado nesta dissertação. Os modelos RNAs constituem uma importante técnica estatística não-linear, capaz de resolver uma gama de

problemas de grande complexidade. Por isso, são modelos úteis em situações nas quais não é possível definir explicitamente uma lista de regras.

A figura 3.1 mostra a estrutura de um neurônio biológico e, em seguida, a figura 3.2 mostra a estrutura funcional de um neurônio artificial, onde é descrito o que se encontra no interior do k-ésimo neurônio de uma rede.

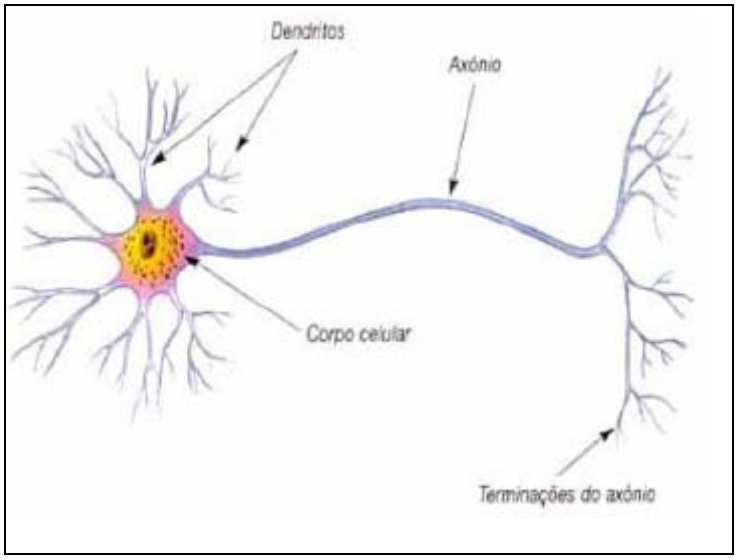

Figura 3.1: Neurônio biológico

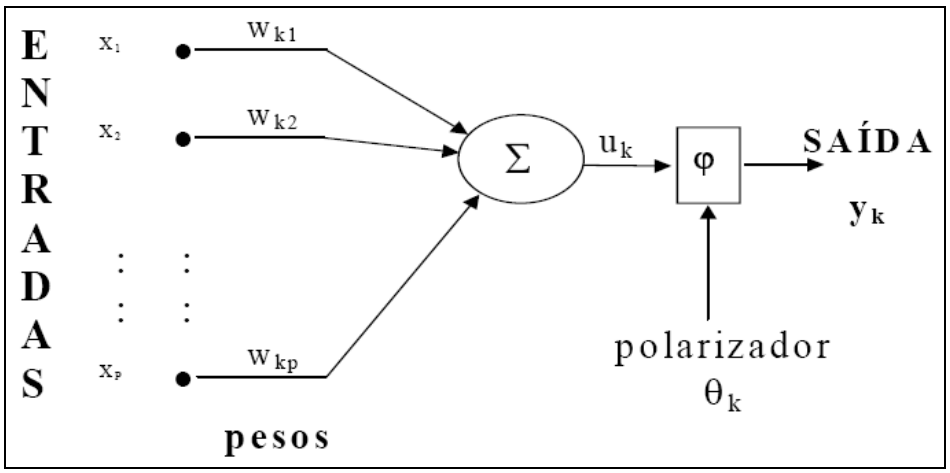

Figura 3.2: Descrição de um k-ésimo neurônio

O neurônio artificial é composto por p terminais de entrada x_1, x_2, x_3,..., x_p, que podem ser os padrões de entrada da rede ou as saídas dos neurônios da camada anterior, representando os dendritos de um neurônio humano. É composto, também, por uma única saída y_k, representando o axônio do neurônio humano. Cada entrada do neurônio artificial x_i possui associado a ela um valor

w_{ki}, chamado de peso sináptico, em uma alusão às sinapses, que são as conexões entre os dendritos de um neurônio e os axônios de outros neurônios. Esses pesos têm a função de regular os valores das entradas no corpo da célula. O corpo da célula, núcleo do neurônio ou neurônio propriamente dito, é onde são processadas as entradas já multiplicadas pelos respectivos pesos, ou seja, é onde é processada a soma $\sum_{i=1}^{p} x_i w_{ki} + \theta_k$, onde $\sum_{i=1}^{p} x_i w_{ki} = u_k$. A soma é, então, fornecida à função de ativação, gerando assim a saída do neurônio. As funções de ativação atualmente utilizadas nos modelos são não-lineares, monotônicas e limitadas (ex. logística e tangente hiperbólica).

3.2.3
Estrutura da Rede

A Rede neural artificial pode ser formada por uma ou mais camadas (fileiras) de neurônios. De uma maneira simples, tem-se camadas de neurônios enfileiradas e ligadas entre si. Na topologia mais tradicional, cada neurônio da primeira camada se liga através de sua saída a uma entrada de cada um dos neurônios da segunda camada. O mesmo acontece com os neurônios da segunda camada em relação aos da terceira e assim por diante.

A figura 3.3 abaixo mostra uma rede neural com quatro entradas, três neurônios na primeira camada (neurônios intermediários) e dois neurônios na camada de saída. As camadas que não fazem parte da camada de saída (neste caso a primeira) são chamadas de camadas escondidas.

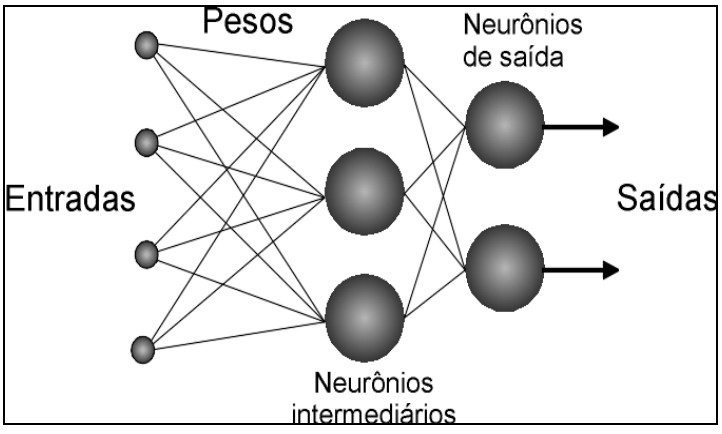

Figura 3.3: Exemplo de estrutura da rede

É importante lembrar que, além das entradas da rede, todas as conexões entre neurônios são multiplicadas por pesos. Visto isso, olhando para a figura 3.3, as saídas dos neurônios intermediários são também multiplicadas por pesos sinápticos antes de serem somadas dentro dos neurônios da camada de saída.

3.2.4
Processamento Neural

As Redes Neurais devem ser configuradas de modo que a apresentação de um conjunto de entradas produza o conjunto de saídas desejado. O processamento de uma Rede Neural pode ser dividido em duas fases:

- Aprendizado (*Learning*): Processo de atualização dos pesos sinápticos para a aquisição do conhecimento - *Aquisição da Informação*
- Recuperação de Dados (*Recall*): Processo de cálculo da saída da rede, dado certo padrão de entrada - *Recuperação da Informação.*

Definir adequadamente o número de camadas escondidas e o número de processadores em cada uma dessas camadas é a garantia do compromisso entre *Convergência e Generalização*.

A *Convergência* é a capacidade da Rede Neural de aprender todos os padrões do conjunto de treinamento. Se a rede neural for pequena, não será capaz de armazenar todos os padrões necessários. Isto é, a rede não deve ser demasiadamente rígida, a ponto de não modelar fielmente os dados. Se a rede for muito grande (muitos parâmetros = pesos), não responderá corretamente a padrões nunca vistos. Isto é, a rede não deve ser excessivamente flexível a ponto de modelar também o ruído.

Chama-se de *Generalização* a capacidade de um modelo de aprendizado responder corretamente aos exemplos que lhe são apresentados, sendo que estes exemplos não devem estar presentes na base de aprendizado. Um modelo que tem uma boa generalização é aquele que responde corretamente aos exemplos contidos na base de aprendizado, mas também a outros exemplos, diferentes daqueles da base de aprendizado, e que estão contidos em uma base de teste. A capacidade de generalizar é a principal capacidade buscada nas tarefas que envolvem aprendizado [73].

3.2.5
Aprendizagem e Treinamento

Na realidade, Redes Neurais Artificiais, imitando o cérebro, possuem a capacidade de aprender por meio de exemplos e fazer interpolações e extrapolações do que aprenderam. No aprendizado conexionista não se procura obter regras como na abordagem simbólica da Inteligência Artificial (IA), mas determinar a intensidade ótima das conexões entre os neurônios. Um conjunto de procedimentos bem definidos para adaptar os parâmetros de uma RNA para que a mesma possa aprender uma determinada função é chamado de *algoritmo de aprendizado*. Como era de se esperar, existem vários algoritmos de aprendizado, cada qual voltado para um conjunto de aplicações específicas e com suas vantagens e desvantagens.

A etapa de aprendizagem consiste em um processo iterativo de ajuste dos pesos sinápticos. Estes, ao final do processo, guardam o conhecimento que a rede adquiriu do ambiente em que está operando.

O aprendizado é o resultado das muitas apresentações de um determinado conjunto de exemplos de treinamento. Neste contexto, é válido destacar o conceito de **época** [74]. Época significa uma apresentação completa de todo o conjunto de treinamento.

Em resumo, basicamente, o aprendizado se dá pela atualização dos pesos sinápticos, atualização esta que é proporcional à diferença (erro) entre a saída calculada pela rede neural (que é diretamente proporcional aos pesos sinápticos utilizados) e a saída real. Existem várias formas de atualização dos pesos, também chamadas de regras de aprendizado, dentre as quais, a minimização do erro médio quadrático pelo algoritmo do gradiente descendente (usado pelo algoritmo de aprendizado *BackPropagation*) ou por meio de um algoritmo que utiliza uma aproximação do método de Newton (usado pelo algoritmo de aprendizado *Levenberg Marquardt*).

O treinamento da rede pode se dar de duas maneiras, *Batch* ou Incremental:

- *Batch, Batelada ou Por ciclos*: a atualização dos pesos acontece somente após a apresentação de todos os padrões. Todos os padrões são avaliados com a mesma configuração de pesos.

- *Incremental ou Por Padrão*: o algoritmo faz a atualização dos pesos após a apresentação de cada novo padrão. Por isso mesmo, a freqüência das atualizações em um mesmo período tende a ser maior do que no caso anterior. Como neste caso o algoritmo tende a levar a rede a aprender melhor o último padrão apresentado, é interessante tornar a apresentação dos exemplos aleatória.

A eficiência dos dois métodos depende do problema em questão. O aprendizado é mantido de época em época até que os pesos se estabilizem e o erro quadrático médio sobre todo o conjunto de treinamento convirja para um valor mínimo, de modo que o objetivo pretendido seja atingido.

Uma rede pode se especializar demasiadamente em relação aos exemplos contidos na base de aprendizado. Este tipo de comportamento leva a um problema de aprendizado conhecido como super-aprendizado ou *over-fitting*. Normalmente o *over-fitting* pode ser detectado e evitado por meio de um método de interrupção ou ajuste do treinamento denominado *early stopping* (ver apêndice 5).

Os procedimentos de treinamento podem ser divididos em dois tipos:

- Supervisionado – a rede é treinada por meio do fornecimento dos valores de entrada e de seus respectivos valores desejados de saída (figura 3.4);
- Não-supervisionado – Não requer o valor desejado de saída da rede. O sistema extrai as características do conjunto de padrões, agrupando-os em classes inerentes aos dados. Este tipo de treinamento é aplicado a problemas de "clusterização" (figura 3.5).

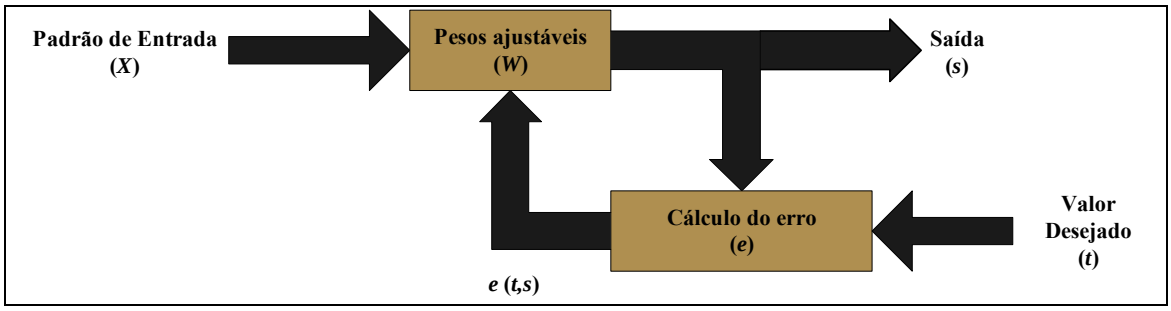

Figura 3.4: Treinamento Supervisionado

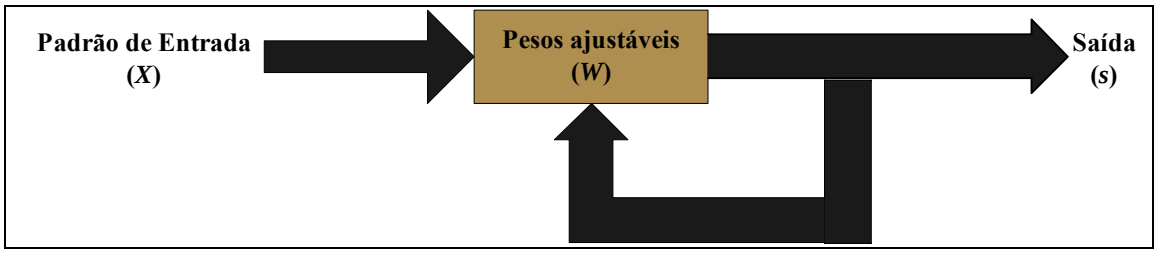

Figura 3.5: Treinamento Não-Supervisionado

O algoritmo de treinamento de redes neurais mais utilizado para previsão, reconhecimento e classificação de padrões é do tipo supervisionado e denominado Backpropagation que, como já dito anteriormente, utiliza como forma de ajuste dos pesos a minimização do erro médio quadrático pelo algoritmo do gradiente descendente.

Nesta dissertação, utiliza-se o algoritmo de treinamento *Levenberg-Marquardt* [25], pois este é um dos métodos mais rápidos para o treinamento de redes neurais *feed-forward* de tamanho moderado (até algumas centenas de pesos). No entanto, pelo fato do *Levenberg-Marquardt* ser uma variação do *Backpropagation*, a seguir são feitas algumas observações relativas a este algoritmo de treinamento. No apêndice 4 encontram-se todas as informações julgadas necessárias ao entendimento do algoritmo *Levenberg-Marquardt*.

O algoritmo *Backpropagation* é usado para treinamento de redes *Multilayer perceptron* e tem como características marcantes:

- os erros dos elementos processadores da camada de saída (que são conhecidos, no treinamento supervisionado) são retropropagados para as camadas intermediárias (daí o nome do algoritmo);

- Regra de propagação – $net_j = \Sigma\, x_i . w_{ji} + \theta_j$;

- Função de ativação – Função não-linear, diferenciável em todos os pontos;

- Topologia – Múltiplas camadas;

- Algoritmo de aprendizado - Tipo supervisionado;

- Valores de entrada/saída – Binários e/ou contínuos.

Outra importante característica do *Backpropagation* é o processo de atualização dos pesos sinápticos, que é feito por meio da minimização do erro quadrático pelo método do *Gradiente descendente*. Por este método, o fator de atualização ótimo para o peso w_{ji} relativo à entrada i do processador j é dado

por:

$$\Delta w_{ji} = -\eta \frac{\delta E}{\delta w_{ji}}, \tag{3.11}$$

lembrando que o gradiente de uma função está na direção e sentido onde a função tem taxa de variação máxima.

Na equação 3.11 tem-se

$$E = \frac{1}{2} \sum_p \sum_{i=1}^{k} (d_i^p - y_i^p)^2, \tag{3.12}$$

onde E é a medida do erro total, p é o número de padrões, k é o número de unidades de saída, d_i é a i-ésima saída desejada e y_i é a i-ésima saída gerada.

Embora o erro total E seja definido pela soma dos erros das saídas para todos os padrões, será assumido, sem perda de generalidade, que a minimização do erro para cada padrão individualmente levará à minimização do erro total. Assim, o erro passa a ser definido por:

$$E = \frac{1}{2} \sum_{j=1}^{k} (d_j - y_j)^2 \tag{3.13}.$$

Usando a regra da cadeia em (3.11), tem-se:

$$\Delta w_{ji} = -\eta \frac{\delta E}{\delta w_{ji}} = -\eta \frac{\delta E}{\delta net_j} \frac{\delta net_j}{\delta w_{ji}} \tag{3.14}.$$

Como $net_j = \Sigma\, s_i . w_{ji} + \theta_j$, então:

$$\frac{\delta net_j}{\delta w_{ji}} = s_i, \tag{3.15}$$

onde s_i é o valor de entrada recebido pela conexão i do neurônio j.

Como $\frac{\delta E}{\delta net_j}$ é o valor calculado do erro do processador j, então passar-se-á a chamá-lo de e_j. Pode-se provar que, para neurônios da camada de saída,

$$e_j = (t_j - s_j) . \frac{\delta s_j}{\delta net_j}, \tag{3.16}$$

e, para neurônios de camada escondida,

$$e_j = \frac{\delta s_j}{\delta net_j} \Sigma_k\, e_k . w_{kj} \quad , \tag{3.17}.$$

Das equações acima, pode-se estabelecer que:

$$-\Delta w_{ji} = \eta . s_i . e_j \qquad (3.18),$$

onde e_j é dado por (3.16) ou (3.17).

Alguns detalhes interessantes:

- No aprendizado supervisionado, em princípio, só se conhece o erro na camada de saída (e_k);
- Este erro na saída (e_k) é função do potencial interno do processador (net_k);
- O net_k depende dos estados de ativação dos processadores da camada anterior (s_j) e dos pesos das conexões (w_{kj});

Portanto, os estados de ativação s_j de uma camada escondida afetam, em maior ou menor grau, o erro de todos os processadores da camada subseqüente.

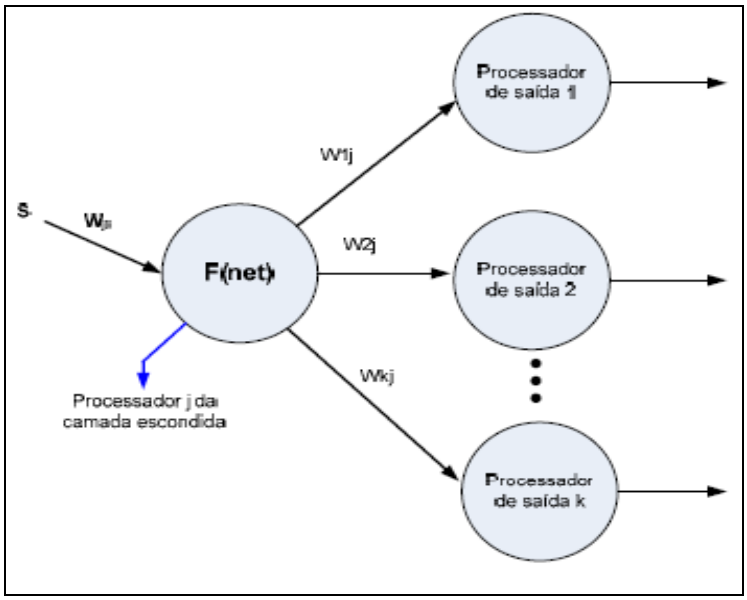

Figura 3.6: Conexões de um neurônio de camada escondida

Ressalta-se, ainda, que o algoritmo *Backpropagation* tem duas fases distintas de propagação de dados, para cada padrão apresentado:

- Feed-forward → as entradas se propagam pela rede, da camada de entrada até a camada de saída;
- Feed-backward → os erros se propagam na direção contrária ao fluxo de dados, indo da camada de saída até a primeira camada escondida.

3.3
Wavelets

3.3.1
Introdução

Segundo Amara Graps [75], Wavelets são funções matemáticas que decompõem dados em diferentes componentes de freqüência, e então estudam cada componente com uma resolução de acordo com a sua escala. Elas levam vantagem sobre o tradicional método de Fourier na análise de situações físicas quando o sinal contiver descontinuidades e picos. Wavelets foram desenvolvidas de forma simultânea e independente nos campos da matemática, física quântica, engenharia elétrica e geologia sísmica. Intercâmbios entre esses campos durante os últimos dez anos têm feito surgirem diversas novas aplicações para Wavelets, tais como na compressão de imagens, visão humana, radar e até em previsão de terremotos.

A idéia fundamental das Wavelets é a análise de dados de acordo com a escala. Algoritmos Wavelet processam dados em diferentes escalas ou resoluções. Similarmente ao zoom de uma câmera, ao se examinar um sinal com uma grande janela, ou seja, sem zoom, notam-se somente as características mais grosseiras dele. No entanto, ao se olhar através de uma pequena janela, ou seja, com um grande zoom, pode-se perceber as características mais finas, os detalhes. Fazendo uma analogia, o resultado de uma análise Wavelet é ver tanto a floresta como suas árvores.

Por muitas décadas, cientistas têm buscado funções mais apropriadas para aproximar sinais pulsados, com picos, do que as funções seno e co-seno, que são a base da análise de Fourier. Senos e co-senos são funções não locais e têm domínio infinito, e portanto, fazem um trabalho muito pobre na aproximação de sinais com muitos picos. Sendo mais específico, a análise de Fourier é muito útil na análise de sinais estacionários, cujo conteúdo de freqüência não muda no tempo. Ou seja, a análise de Fourier é útil quando não é necessário saber a localização temporal das componentes de freqüência. No entanto, para séries não-estacionárias, cujo conteúdo de freqüências varia no tempo, a série de Fourier só será útil para identificar quais componentes de freqüência existem no sinal, não identificando

quando estes ocorrem. Segundo Goldenstein [76], como exemplo, pode-se fazer uma analogia com o processamento de um radar: a existência de determinada freqüência detecta a presença de um objeto, a localização dessa freqüência permite determinar a posição do objeto.

Pelo fato das transformadas Wavelets usarem funções de aproximação que estão contidas habilmente em domínios finitos, além dos sinais estacionários, elas são apropriadas também para análise de sinais não-estacionários, dando a composição de freqüências do sinal em cada janela de tempo (ao decorrer do tempo). Esta sua habilidade pode ser definida na afirmação de que a transformada Wavelet possibilita a análise do sinal em dois domínios: freqüencial e temporal.

O procedimento principal da análise Wavelet é adotar uma função protótipo chamada Wavelet-mãe, a qual, quanto mais se assemelhar ao sinal original, melhor será o resultado da decomposição wavelet. Em termos gerais, por meio da dilatação ou compressão dessa Wavelet-mãe, pode-se verificar respectivamente características de baixa freqüência – maior intervalo de tempo – maior janela (olhar para a floresta – menor zoom), sendo assim mais grosseiras, e características de alta freqüência – menor intervalo de tempo – menor janela (olhar para as árvores da floresta – maior zoom), sendo assim mais finas, com mais detalhes.

3.3.2
Perspectiva Histórica

Segundo Hubbard em [77], traçar a história de wavelets é quase um trabalho para um arqueólogo. “Eu já encontrei no mínimo 15 raízes distintas da teoria, algumas anteriores a 1930”, afirma Yves Meyer [78].

Muito do trabalho foi desenvolvido na década de 1930 e, naquela época, esforços separados não pareciam ser partes de uma teoria coerente. Antes de 1930, um dos principais matemáticos a lidar com Wavelets foi Jean Baptiste Joseph Fourier (1807), com sua teoria de análise de freqüências. Ele afirmou que qualquer função periódica *f(x)* pode ser expressa como uma soma de senos e/ou co-senos – chamada série de Fourier:

$$f(x) = a_0 + \sum_{-\infty}^{\infty} (a_k \cos kx + b_k \, senkx) \tag{3.19}$$

A afirmação de Fourier desempenhou um papel essencial no que diz respeito à evolução das idéias que os matemáticos tinham sobre as funções. Ele abriu a porta para um novo universo funcional.

A partir de 1807, depois de explorarem o significado das funções, a convergência das séries de Fourier e sistemas ortogonais, os matemáticos foram gradualmente migrando de seus conceitos prévios de análise de freqüência para a noção de análise de escala. Isto é, analisando *f(x)* por meio da criação de estruturas matemáticas que variam em escala. A idéia consiste em se construir uma função, deslocá-la no eixo *x* (das abscissas) de alguma quantidade, trocar a sua escala (expandi-la ou contraí-la) e aplicar esta estrutura na aproximação de um sinal. A seguir, desloca-se a última estrutura usada na aproximação do sinal, e troca-se a sua escala novamente. Aplica-se, então, esta nova estrutura ao mesmo sinal para se conseguir uma nova aproximação, e assim por diante. Esse tipo de análise por meio de mudanças de escala é menos sensível a ruídos, pois mede as flutuações médias do sinal em diferentes escalas.

A primeira menção a Wavelets apareceu em um apêndice da tese de A. Haar (1909). Uma propriedade da Wavelet de Haar é que ela tem suporte compacto, o que significa que desaparece fora de um intervalo finito. Infelizmente, as Wavelets de Haar não são continuamente diferenciáveis, o que limita suas aplicações.

Em meados de 1930, vários grupos trabalhando independentemente pesquisaram a representação de funções usando funções base de escala variável. Usando a função base de escala variável de Haar, Paul Levy, um físico de meados de 1930, investigou o movimento browniano, um tipo de sinal aleatório [79]. Ele verificou que a função de Haar era superior às funções base de Fourier no estudo de detalhes do movimento browniano. Outra pesquisa feita por Littlewood, Paley e mais tarde Stein, envolvia computação da energia de uma função *f(x)*:

$$\text{energia} = \frac{1}{2}\int_0^{2\pi} |f(x)|^2 \, dx \tag{3.20}$$

Os pesquisadores descobriram uma função que podia variar em escala e conservar sua energia. Hoje, sabe-se que esta é uma importante característica ou condição crucial para se ter uma wavelet, pois, como a energia se mantém, então é

garantido que, depois de sofrer uma transformação wavelet, uma função pode ser retornada pela transformada inversa.

Entre 1960 e 1980, os matemáticos Guido Weiss e Ronald Coifman estudaram os elementos mais simples de espaço de função, chamados átomos, com o objetivo de encontrar as regras de montagem que permitem a reconstrução de todos os elementos de um espaço de função usando esses átomos.

Por volta de 1981, Jean Morlet, um geofísico da companhia francesa Elf-Aquitaine procurou Alex Grossman, um físico, e juntos desenvolveram wavelets no contexto de física quântica.

Então, em 1989, o francês Stephane Mallat desenvolveu a análise multiresolução via algoritmo piramidal [80]. Alguns anos mais tarde, Ingrid Daubechies usou o trabalho de Mallat para construir um conjunto de funções base ortonormais wavelet, que são bem mais elegantes, e hoje constituem a base das aplicações em wavelet.

3.3.3
A Transformada de Fourier

Segundo Hubbard [77], embora Wavelets representem um departamento da análise de Fourier, elas são também uma extensão natural dela: as duas linguagens claramente pertencem à mesma família. A história de Wavelets, assim sendo, começa com a história da análise de Fourier. Por sua vez, as raízes da Análise de Fourier são anteriores ao próprio Fourier, embora ele seja um ponto lógico de partida.

Nascido em Auxerre (cidade francesa entre Paris e Dijon) no ano de 1768, Jean Baptiste Joseph Fourier mostrou, em 1807, que qualquer função periódica pode ser expressa como uma soma de senos e/ou co-senos – a chamada série de Fourier. Grosseiramente, o que isso significa é que qualquer curva que periodicamente repita ela mesma, não importando quão recortada ou irregular seja, pode ser expressa como a soma de oscilações perfeitamente suaves (senos e co-senos), como mostrado na figura 3.7.

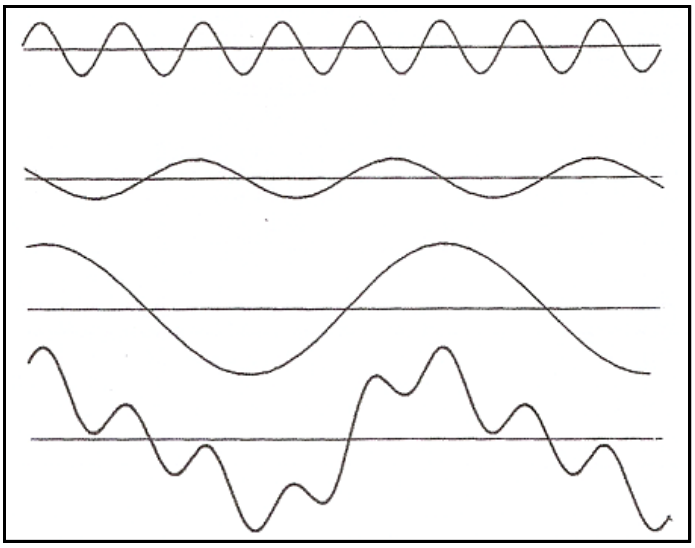

Figura 3.7: A função na parte de baixo é composta pelas três funções acima dela

Funções não-periódicas também podem ser representadas por somas de senos e co-senos, bastando para isso que decresçam rápido o suficiente para que a área sob seu gráfico seja finita.

A transformada de Fourier é o procedimento matemático que quebra uma função nas freqüências que a compõem, de forma similar à chuva que, funcionando como um prisma, decompõe a luz do sol em todas as cores, formando assim o arco-íris. Ela transforma uma função f de t (tempo) em uma função F de k (freqüência). Esta nova função é chamada de transformada de Fourier da função original (ou, quando a função original é periódica, sua série de Fourier). Para funções ou sinais que variam com o tempo – música, por exemplo, ou as flutuações do mercado de ações – a freqüência é mais comumente medida em Hertz, ou ciclos por segundo.

Funções podem também variar com o espaço. Como numa função que depende do tempo, a freqüência é o inverso do tempo, em uma função que varie com a distância, chama-se de *número de onda*, ao inverso da distância.

Uma função e sua transformada de Fourier são duas faces da mesma informação. A função mostra a informação temporal e esconde a informação sobre as freqüências. A função correspondente a uma gravação musical mostra como a pressão do ar (produzida pelas ondas sonoras) muda com o tempo, mas não indica quais freqüências – quais notas – compõem a música. A transformada de Fourier mostra informação sobre freqüências e esconde a informação temporal:

a transformada de Fourier da música indica quais notas são tocadas, mas não diz quando são tocadas. De qualquer maneira, a função e sua transformada, em conjunto, contêm toda a informação do sinal.

A série de Fourier de uma função periódica f de período T é:

$$f(t) = a_0 + (a_1 \cos\omega_0 t + b_1 sen\omega_0 t) + (a_2 \cos 2\omega_0 t + b_2 sen2\omega_0 t) + ... \qquad (3.21)$$

onde $\omega_0 = \frac{2\pi}{T} = 2\pi k$, sendo *k* a freqüência temporal, que é igual ao inverso do período: $k = \frac{1}{T}$.

Os coeficientes de Fourier a_1, a_2, a_3 ... indicam quanto das funções $\cos\omega_0 t$, $\cos 2\omega_0 t$, $\cos 3\omega_0 t$ ... (isto é, co-senos de freqüências 1 Hz, 2 Hz, 3 Hz ... quando k=1) a função f(t) contém; os coeficientes b_1, b_2,b_3, ... indicam quanto das funções $sen\omega_0 t$, $sen 2\omega_0 t$, $sen 3\omega_0 t$ ... (isto é, senos de freqüências 1 Hz, 2 Hz, 3 Hz ...quando k=1) a função f(t) contém. A série de Fourier consiste somente daqueles senos e co-senos que são múltiplos inteiros da freqüência base ou fundamental.

A equação (3.21) é mais comumente escrita assim:

$$f(t) = a_0 + \sum_{n=1}^{\infty} (a_n \cos n\omega_0 t + b_n sen\, n\omega_0 t) \qquad (3.22)$$

Para computar os coeficientes de Fourier de uma função periódica *f* de período T, multiplica-se *f* pelas funções *cos* $n\omega_0 t$ e *sen* $n\omega_0 t$. Como estas funções oscilam entre +1 e -1, esta multiplicação produz uma função cujo gráfico oscila entre os gráficos de $+f$ e $-f$.

A integral dessa multiplicação (essa integral equivale à área da nova função formada por essa multiplicação) é o coeficiente de Fourier na freqüência *nk*. O intervalo de integração equivale ao período T da função e os coeficientes são dados pelas seguintes equações:

$$a_0 = \frac{1}{T}\int_0^T f(t)\,dt,\ a_n = \frac{2}{T}\int_0^T f(t)\cos n\omega_0 t\,dt \ \ e \ \ b_n = \frac{2}{T}\int_0^T f(t)\, sen\, n\omega_0 t\,dt \qquad (3.23)$$

Os coeficientes de freqüências muito altas de uma função suave tendem a zero, entretanto, não é verdade que eles fiquem menores quanto maiores forem suas freqüências.

Já foi dito que as únicas freqüências que contribuem para a série de Fourier de uma função periódica são os múltiplos inteiros da freqüência fundamental base da função. Se uma função não é periódica, mas decresce suficientemente rápido

no infinito de forma que a área sob seu gráfico seja finita, é possível descrevê-la como uma superposição de senos e co-senos – para analisá-la em termos de suas freqüências. Entretanto, agora, para funções não periódicas, devem ser computados coeficientes para todas as freqüências possíveis, ou seja, não mais somente valores inteiros de freqüências, mas valores em toda a reta real. Neste caso (funções não-periódicas) os coeficientes podem ser calculados por:

$$a(\tau) = \int_{-\infty}^{\infty} f(t)\cos 2\pi\tau t\,dt \qquad e \qquad b(\tau) = \int_{-\infty}^{\infty} f(t)sen2\pi\tau t\,dt \qquad (3.24)$$

sendo τ a freqüência, que pode assumir valores em toda a reta real. O intervalo de integração -∞ a +∞ se deve ao fato da função ser não periódica.

Pode-se, ainda, expressar as equações (3.22) a (3.24) usando números complexos, ficando a equação (3.22):

$$f(t) = \sum_{k=-\infty}^{\infty} c_k e^{-2\pi ikt} \qquad (3.25)$$

enquanto as equações em (3.23) para os coeficientes de uma série de Fourier ficam:

$$c_k = \int_{0}^{T} f(t)e^{2\pi ikt}\,dt \qquad (3.26)$$

As fórmulas para a transformada de Fourier de uma função que decresce no infinito e para a reconstrução da função através da transformada são, respectivamente

$$\hat{f}(\xi) = \int_{-\infty}^{\infty} f(x)e^{2\pi i\xi x}dx \qquad e \qquad f(x) = \int_{-\infty}^{\infty} \hat{f}(\xi)\,e^{-2\pi i\xi x}d\xi \qquad (3.27)$$

Empregou-se nas equações acima ξ ao invés de τ, e *x* ao invés de *t*, pois as funções podem variar também com o espaço, e não só com o tempo. É isso que informam as letras ξ e *x*.

3.3.3.1
Análise de Fourier Short-Time (Windowed)

Enquanto a análise de Fourier força a escolha entre tempo e freqüência, "nossas experiências diárias insistem na descrição em termos de ambos", como escreveu Dennis Gabor [81]. Em 1946, ele adaptou a transformada de Fourier para

analisar uma pequena seção de um sinal no tempo – uma técnica chamada "janelamento" do sinal. Em inglês, a STFT (short-time Fourier transform) mapeia um sinal em uma função bi-dimensional (tempo e freqüência), provendo portanto, alguma informação sobre quais e quando as freqüências de um sinal ocorrem. Entretanto, pode-se obter esta informação com exatidão limitada pelo tamanho da janela.

Apesar de a STFT propiciar informação útil sobre tempo e freqüência simultaneamente, o ponto fraco desta técnica é que, uma vez escolhido o tamanho da janela de tempo, ele permanece fixo para todos os níveis (valores) de freqüência, durante todo o tempo (figura 3.6).

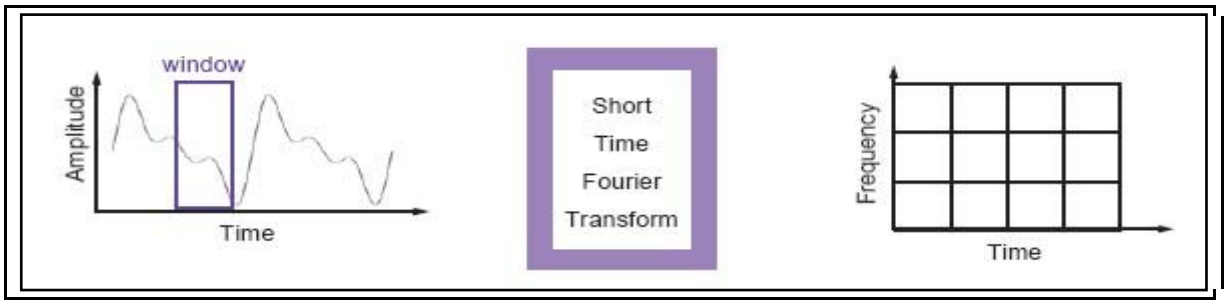

Figura 3.8: Dois gráficos ilustrativos da STFT, também conhecida como Windowed Fourier Analysis

A idéia é estudar as freqüências de um sinal, segmento por segmento. A janela que define o tamanho do segmento a ser analisado é um pequeno pedaço de curva; esta curva é sucessivamente preenchida com pedaços de oscilações de funções de diferentes freqüências.

Enquanto a transformada clássica de Fourier compara todo o sinal com infinitos senos e co-senos de diferentes freqüências, no intuito de levantar quanto o sinal possui de cada freqüência, a STFT compara um segmento do sinal a pedaços de curvas oscilantes (funções oscilantes), primeiro de uma freqüência, depois de outra, e assim por diante. Depois de um segmento (um pedaço janelado) do sinal ter sido analisado, translada-se a janela ao longo do sinal para que outro segmento seja analisado.

Na figura 3.6, o gráfico da esquerda mostra a escolha do tamanho (largura) da janela. O pedaço da curva que está "janelado" é então analisado. Esta análise, conforme dito acima, é tão-somente a comparação deste pedaço janelado com pedaços de curvas, primeiro de uma freqüência menor, e depois se vai aumentando essa freqüência. Depois que o pedaço janelado já foi analisado,

desliza-se a janela para a direita e passa-se a analisar outro pedaço da curva. O gráfico da esquerda, na verdade, lembra através das linhas verticais que a largura da janela de análise é fixa. Suas linhas horizontais lembram que cada pedaço janelado da curva será analisado (comparado) a curvas de diversas freqüências.

Infelizmente, este método impõe sérias restrições. Quanto menor a janela, melhor pode-se perceber mudanças repentinas, tais como picos e descontinuidades – em contrapartida não se consegue perceber as componentes de baixa freqüência do sinal. Essas baixas freqüências não podem ser enquadradas dentro de pequenas janelas. Caso seja escolhida uma janela grande, pode-se ver mais das baixas freqüências e não se consegue perceber mudanças repentinas como picos e descontinuidades.

O fato é que muitos sinais requerem uma metodologia mais flexível – onde se possa variar o tamanho da janela e assim estudar-se mais precisamente ambos, tempo e freqüência.

3.3.4
A Transformada Wavelet

A análise Wavelet representa o próximo passo mais lógico: uma técnica de janelamento na qual o tamanho da janela varie. A análise Wavelet permite o uso de grandes intervalos de tempo, onde se deseja informações de baixa freqüência mais precisas, e regiões menores, onde se deseja informações de alta freqüência.

Wavelets é uma extensão da análise de Fourier. Como acontece com a transformada de Fourier, o ponto principal em relação às wavelets não são elas próprias – elas são um meio para um fim. O objetivo é transformar a informação contida em um sinal em números – coeficientes – que possam ser manipulados, armazenados, transmitidos, analisados ou usados para reconstruir o sinal original.

Wavelets são formas de onda de valor médio igual a zero – metade da sua área é positiva e metade é negativa e, ao contrário dos senos e co-senos de Fourier, que são formas de onda de duração não limitada, se estendendo de $-\infty$ até $+\infty$, as wavelets são formas de onda de duração limitada ou, como muitos dizem, de domínio finito ou compacto (figura 3.7). Além disso, enquanto senóides são suaves e previsíveis, wavelets são irregulares e assimétricas. Isso faz com que sinais com rápidas mudanças, como picos, por exemplo, possam ser melhor

analisados com o auxílio de wavelets (intrinsecamente irregulares) do que com senóides (intrinsecamente suaves). Sendo assim, faz sentido que características locais sejam melhor descritas com wavelets.

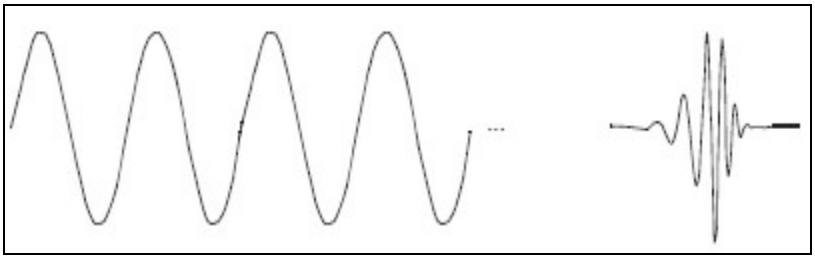

Figura 3.9: Uma senóide à esquerda e uma wavelet à direita

Comparando mais uma vez com a transformada de Fourier, a abordagem e condução básica são as mesmas. Os coeficientes indicam de que maneira a função analisadora (os senos, co-senos ou a wavelet-mãe) precisa ser modificada para que o sinal seja reconstruído. Pode-se literalmente reconstruir o sinal somando-se wavelets-mãe de diferentes tamanhos (dilatadas ou contraídas), em diferentes posições (transladadas), exatamente como se constrói um sinal somando-se senos e co-senos. A técnica básica para computar os coeficientes é a mesma: multiplica-se o sinal e a função analisadora e computa-se a integral desse produto.

Como mostrado na figura 3.8, um coeficiente wavelet mede a correlação, ou ajuste, entre a wavelet (com seus picos e vales) e o correspondente segmento analisado do sinal. Uma forte correlação sugere que o segmento analisado se assemelha muito à wavelet.

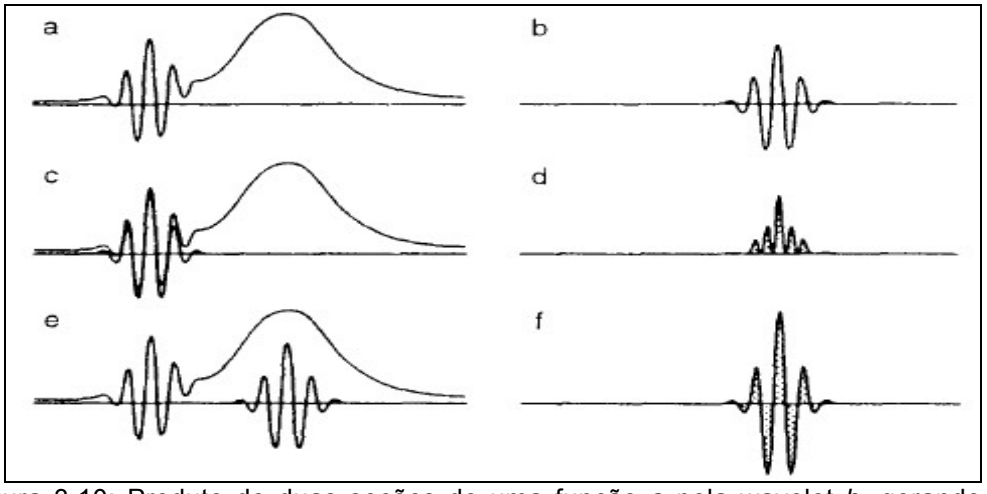

Figura 3.10: Produto de duas seções de uma função *a* pela wavelet *b*, gerando os sinais *d* e *f*, cujas áreas são os Coeficientes wavelets

Na figura 3.8, a wavelet (b) é comparada sucessivamente a diferentes seções de uma função (a). O produto de uma seção da função pela wavelet (b) gera uma nova função e a **área** delimitada por essa nova função é o coeficiente wavelet. Em (c) a wavelet é comparada a uma seção da função bastante semelhante à wavelet. Sempre que há essa semelhança, onde há uma superposição quase perfeita de uma wavelet em relação a uma seção da função tratada, o produto das duas é sempre positivo, pois as duas assumem valores positivos e negativos ao mesmo tempo.

Na Fig. 3.8, é gerado o grande coeficiente mostrado em (d). Em (e) a wavelet é comparada a uma seção da função que, por ter característica de baixa freqüência (mudança lenta de valores), não se assemelha à wavelet. Sendo assim, o produto da wavelet e da função origina a curva (f), que apresenta tanto áreas negativas (abaixo do eixo das abscissas) quanto positivas (acima do eixo das abscissas), fazendo então com que o coeficiente wavelet (igual à soma dessas áreas) seja pequeno. O sinal é analisado em diferentes escalas, usando wavelets de diferentes larguras.

Pode-se concluir então que é possível construir wavelets que dão coeficientes pequenos ou até nulos, quando são comparados a funções lineares, quadráticas e até mesmo polinômios de grau mais elevado. Ou seja, a análise wavelet não consegue interpretar, analisar bem ou enxergar sinais muito comportados, gerando como resultado muitos coeficientes de valor desprezível. Yves Meyer disse, "É como nossa resposta à velocidade. O corpo humano é somente sensível a acelerações, não à velocidade" [77].

3.3.4.1
A Transformada Wavelet Contínua (TWC)

A transformada wavelet contínua de um sinal *f(t)* é definida como a soma sobre todo o tempo do sinal multiplicado por versões "escalonadas" (comprimidas ou esticadas) e transladadas da wavelet-mãe ψ :

$$C(a,b) = \frac{1}{\sqrt{a}} \int_{-\infty}^{+\infty} f(t)\, \psi\left(\frac{t-b}{a}\right) dt \quad \text{, onde } a>0 \text{ e b} \in R \qquad (3.28).$$

Os resultados da TWC são muitos coeficientes wavelets *C*. Em resumo, na equação 3.28, uma função base ψ (wavelet-mãe) é usada para criar uma família

de wavelets $\psi(escala, translação, t) = \frac{1}{\sqrt{a}}\psi\left(\frac{t-b}{a}\right)$, onde *a* (fator de escala) e *b* (fator de translação) são números reais, *a* escalonando (comprimindo ou esticando) a função ψ e *b* a transladando.

$$\psi_b(t) = \psi(t-b) \quad \rightarrow \quad translação\,, \qquad \psi_a(t) = \frac{1}{\sqrt{a}}\psi(\frac{t}{a}) \quad \rightarrow \quad escalonamento\,.$$

A palavra *contínua* se refere à transformada e não às wavelets, embora alguns digam "wavelets contínuas".

É importante ressaltar que, conforme o exemplo da figura 3.8, o valor de *C*(*a*,*b*) representa a similaridade entre a função wavelet-mãe escalonada e transladada $\frac{1}{\sqrt{a}}\psi\left(\frac{t-b}{a}\right)$ e a função analisada *f(t)*. Quanto maior o valor de *C*(*a*,*b*), maior a similaridade da função analisada com a wavelet escalonada e transladada por aqueles valores específicos de *a* e *b*.

A multiplicação de cada coeficiente wavelet pela respectiva wavelet dilatada (escalonada) e transladada produz as constituintes wavelets do sinal original. Mas o que graficamente significam os termos *a* (escala) e *b* (translação)? Considerando senóides, por exemplo, o efeito do fator de escala *a* é muito fácil de notar.

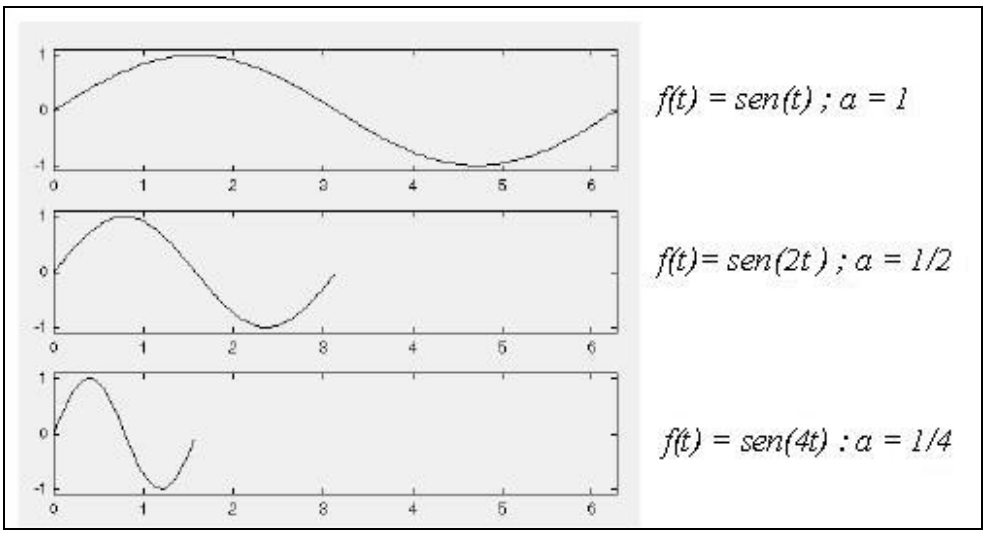

Figura 3.11: O efeito do fator de escala numa senóide

O fator de escala trabalha exatamente da mesma maneira com wavelets. Quanto menor o fator de escala, mais comprimida a wavelet.

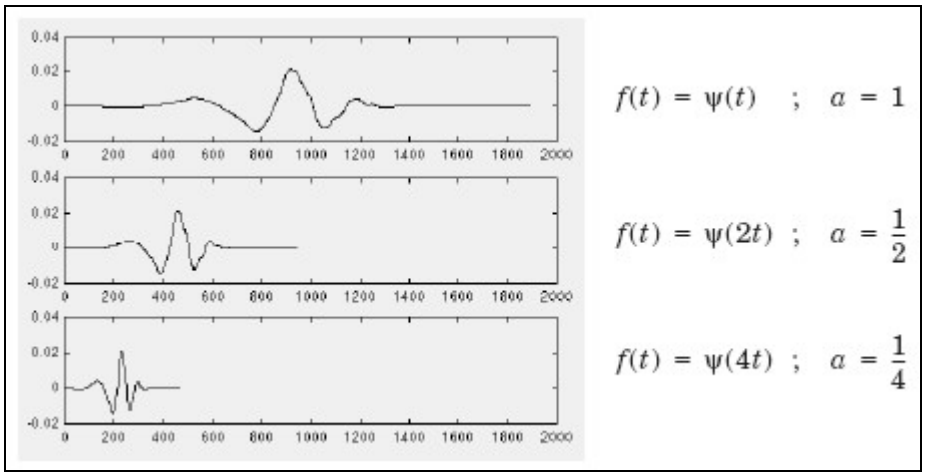

Figura 3.12: O efeito do fator de escala numa wavelet ψ(*t*)

Logicamente, pelos diagramas anteriores, pode-se notar que, para uma senóide *sen(wt)*, o fator de escala *a* é o inverso da freqüência em radianos ω e é inversamente proporcional à freqüência *f* em hertz, tendo em vista que $\omega = 2\pi f$. Obviamente, o mesmo acontece na análise wavelet. De modo mais direto, pode-se relacionar o fator de escala *a* à freqüência ω em radianos e em hertz f , da seguinte maneira:

Pequeno fator de escala $a \Rightarrow$ wavelet comprimida $\Rightarrow$ detalhes de curta duração $\Rightarrow$ alta freq. $\omega \Rightarrow$ gráfico de baixo da figura 3.12 $\Rightarrow f(t) = \psi(4t)$;

Grande fator de escala $a \Rightarrow$ wavelet esticada $\Rightarrow$ mudanças lentas, características grosseiras $\Rightarrow$ baixa freqüência $\omega \Rightarrow$ gráfico de cima da figura 3.12 $\Rightarrow f(t) = \psi(t)$.

Quando se fala, agora, em transladar uma wavelet, isso significa deslocá-la no eixo do tempo (abscissa) de algum valor. Matematicamente, transladar uma função *f(t)* por *k* implica *f(t-k)*.

Figura 3.13: Deslocando uma wavelet

O gráfico à esquerda representa a wavelet $\psi(t)$, enquanto o gráfico à direita representa $\psi(t-k)$.

Nas figuras (3.12) e (3.13), a seguir, mostra-se uma função wavelet $\psi(x) = sen(x)\, e^{-\frac{x^2}{2}}$ com alguns exemplos de translação e escalonamento [82-83].

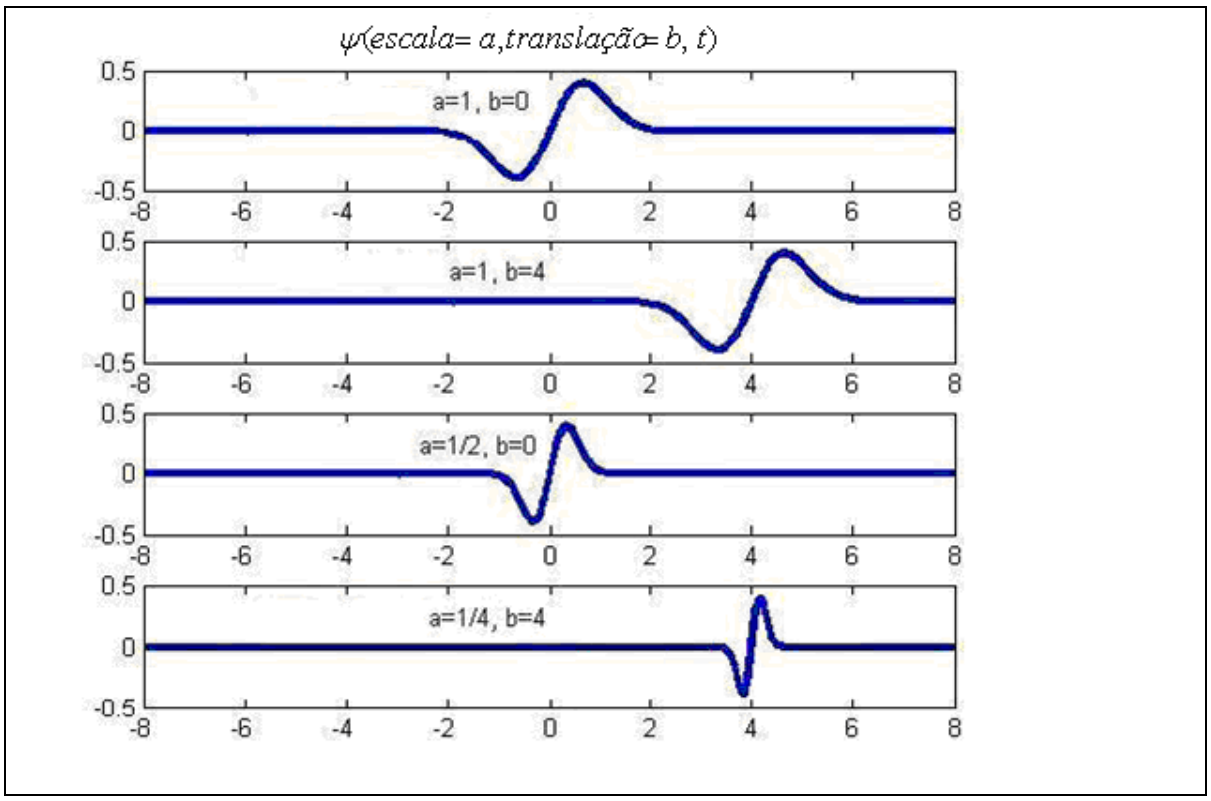

Figura 3.14: Exemplos de translação e escalonamento

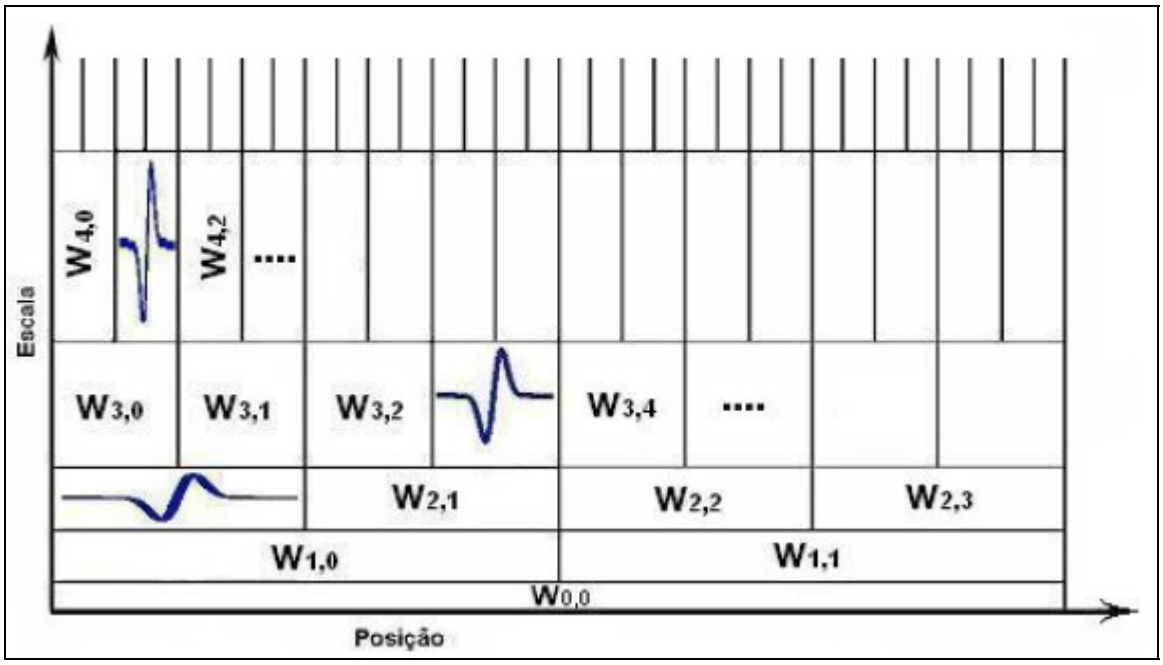

Figura 3.15: Exemplos de translação e escalonamento

3.3.4.2

A Transformada Wavelet Discreta (TWD)

Calcular os coeficientes wavelets para toda escala e posição possíveis é um trabalho árduo. Uma vez que a TWC é conseguida dilatando-se – quer dizer, comprimindo-se ou esticando-se – e transladando-se a wavelet-mãe continuamente, uma grande quantidade de informação redundante é gerada [84]. Portanto, ao invés de se proceder dessa maneira, a wavelet-mãe pode ser dilatada e transladada por meio de valores de escalas e translações especiais. Nesse tipo de análise, conhecida como a Transformada Wavelet Discreta (TWD), a dilatação é mais comumente representada por potências de 2 (algumas vezes chamada dilatação diádica) [77]. A TWD é muito mais eficiente e tão precisa quanto a TWC, e se diferencia desta pela fórmula da wavelet-mãe:

$$\psi(2^k t + l), \tag{3.29}$$

com k e l números inteiros.

De acordo com [85], uma forma eficiente para implementar esse esquema, usando filtros, foi desenvolvida, em 1988, pelo francês Stephane Mallat. Esse caminho prático e eficiente, que na verdade é um algoritmo de filtragem digital,

produz uma "transformada wavelet rápida" – uma caixa preta na qual entram sinais e da qual emergem coeficientes wavelets.

3.3.4.3
O Algoritmo Piramidal de Mallat: Aproximações e Detalhes

Para muitos sinais, o conteúdo de baixa freqüência é a parte mais importante, sendo o que dá ao sinal sua identidade. O conteúdo de alta freqüência, por outro lado, "dá o tempero, a nuância" [85]. Tomando como exemplo a voz humana, se as componentes de alta freqüência são removidas, a voz soa diferente, mas o que está sendo dito ainda pode ser entendido. Entretanto, caso se remova porção suficiente das componentes de baixa freqüência, o que sobra é uma linguagem inarticulada, ininteligível.

Em análise wavelet, são usuais os termos *aproximações* e *detalhes*. Uma aproximação provém da alta escala (*a* grande), sendo a componente de baixa freqüência do sinal. Os detalhes provêm das baixas escalas (*a* pequeno), representando as componentes de alta freqüência do sinal, sendo ainda igual à diferença entre duas aproximações sucessivas do sinal original. Uma aproximação mantém a tendência geral do sinal, enquanto um detalhe mostra suas componentes de alta freqüência.

O processo de filtragem, basicamente, pode ser visualizado na figura 3.14, abaixo. O sinal original S passa através de dois filtros complementares e emerge como dois sinais. Infelizmente, na saída deste processo tem-se o dobro do número de dados da entrada. Então, por exemplo, se o sinal original S consiste de 1.000 amostras de dados, os sinais A e D da saída têm também cada um 1.000 amostras, totalizando 2.000 amostras na saída.

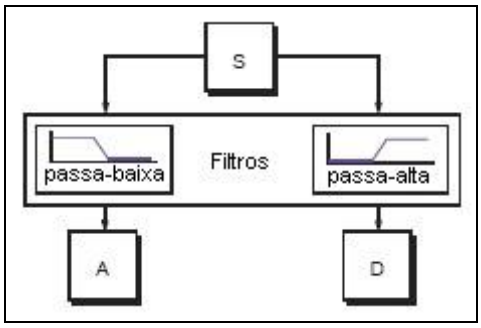

Figura 3.16: S=A+D

Há um método chamado *downsampling*, na qual se pode extrair somente 500 amostras em cada filtro, mantendo assim o tamanho original de 1.000 amostras. No entanto, agora, ao invés de 1.000 valores de *aproximações* e 1.000 valores de *detalhes*, haverá 500 coeficientes de detalhe e 500 coeficientes de aproximação, totalizando 1.000 coeficientes TWD (figuras 3.15 e 3.16).

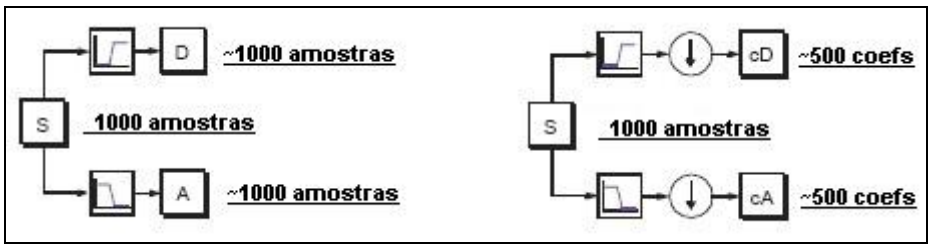

Figura 3.17: Filtragem básica e Downsampling

Esses coeficientes *cA e cD* são obtidos, respectivamente, por meio da convolução de S com a resposta impulsional do filtro passa-baixa e da convolução de S com a resposta impulsional do filtro passa-alta, seguido de uma decimação diádica (representada na figura 3.15 por um círculo com uma seta para baixo), isto é, a cada duas saídas do filtro, descarta-se uma delas.

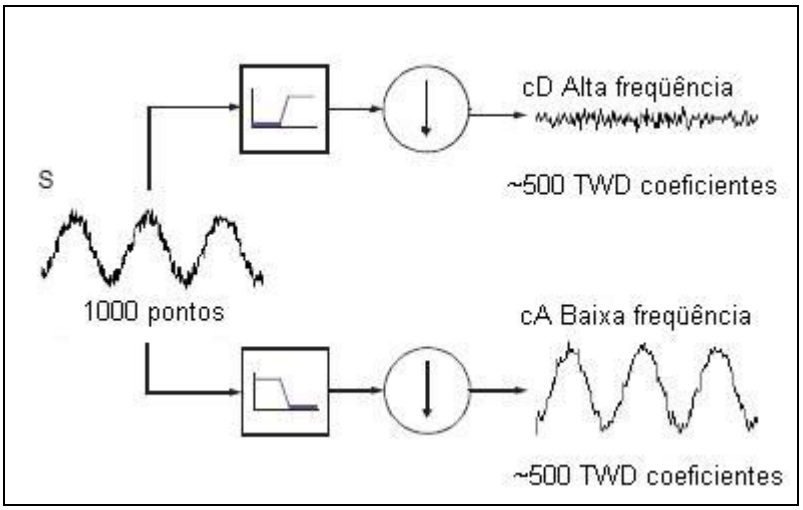

Figura 3.18: Downsampling

Pode-se notar que os coeficientes de detalhes *cD* são pequenos e consistem principalmente de ruídos de alta freqüência. Por sua vez, os coeficientes de aproximação *cA* contêm muito menos ruído do que o sinal original.

3.3.4.4
Decomposição em Múltiplos Níveis

O processo de decomposição em múltiplos níveis é efetivado por meio das decomposições dos coeficientes de aproximação dos vários níveis do sinal. É a chamada Árvore de Decomposição Wavelet (figuras 3.17 e 3.18).

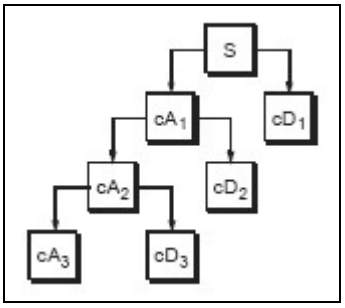

Figura 3.19: Árvore de decomposição wavelet

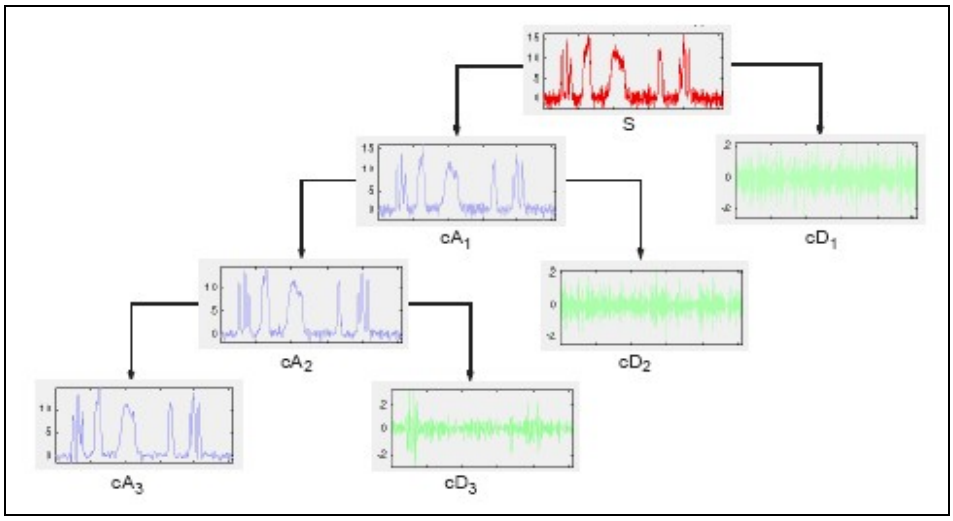

Figura 3.20: Detalhes dos coeficientes na árvore de decomposição wavelet

Teoricamente, pode se decompor o sinal infinitas vezes, só que na prática isso não acontece, devendo ser selecionado um número finito de níveis, com base na natureza do sinal e na experiência do pesquisador.

3.3.4.5
Reconstrução Wavelet

Foi visto que, por meio de *transformadas wavelet discretas*, um sinal é decomposto em coeficientes de aproximação e detalhes. No sentido contrário, esses coeficientes de aproximação e detalhes podem ser usados para montar de volta o sinal original, por meio das *transformadas wavelet discretas inversas* (TWDI). A figura seguinte mostra o processo de reconstrução de um sinal decomposto em dois níveis.

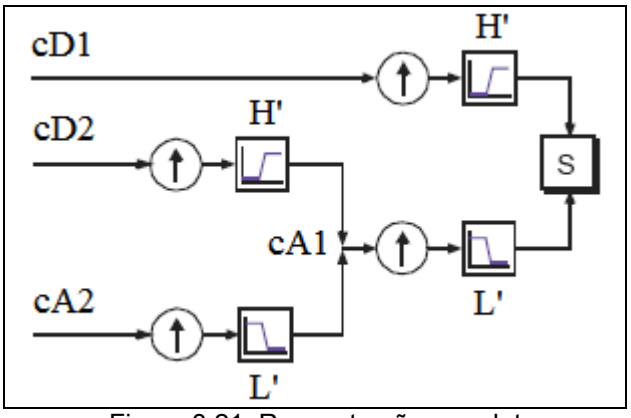

Figura 3.21: Reconstrução wavelet

Na figura, pode-se notar a reconstrução do coeficiente de aproximação *cA1* por meio de *upsampling* (processo inverso ao *downsampling* e representado pela seta para cima) de *cA2 e cD2* e posterior passagem pelos filtros de reconstrução **H´** e **L´**.

Upsampling é o processo de prolongamento de uma componente do sinal por meio da inserção de zeros entre as amostras do sinal.

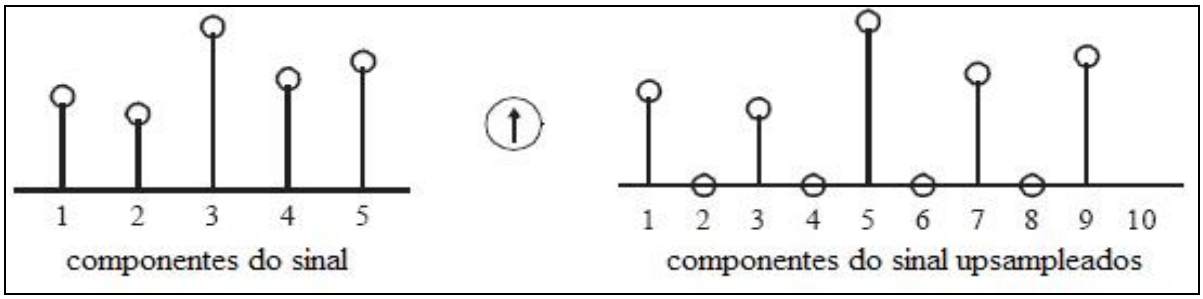

Figura 3.22: Upsampling do sinal

3.3.4.6
Reconstrução de Aproximações e Detalhes

Também é possível reconstruir as aproximações e detalhes por meio de seus coeficientes. Por exemplo, pode-se reconstruir o componente de aproximação de 1º nível *A1* por meio do coeficiente *cA1*. O vetor de coeficientes *cA1* sofre *upsampling* e passa pelo filtro de reconstrução **L´**. Agora, ao invés de combiná-lo com o vetor de detalhe nível 1 *cD1*, a combinação é feita com um vetor de zeros que antes passará por *upsampling* e por reconstrução no filtro **H´**. Processo similar ocorre na reconstrução do detalhe de 1º nível *D1* (figura 3.19).

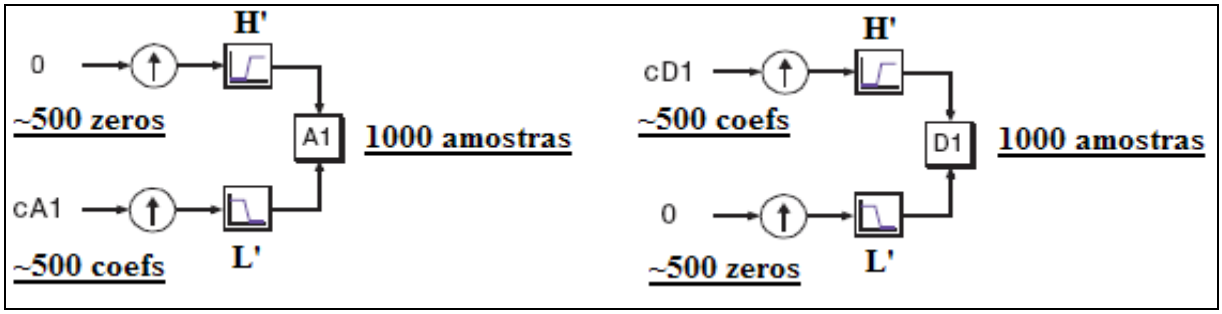

Figura 3.23: Construção dos detalhes e aproximações

Os detalhes e aproximações são partes verdadeiras do sinal original. Então, no caso da decomposição em um único nível da figura 3.19, tem-se:

A1 + D1 = S.

Estendendo para uma análise de múltiplos níveis, tem-se a estrutura da figura 3.22.

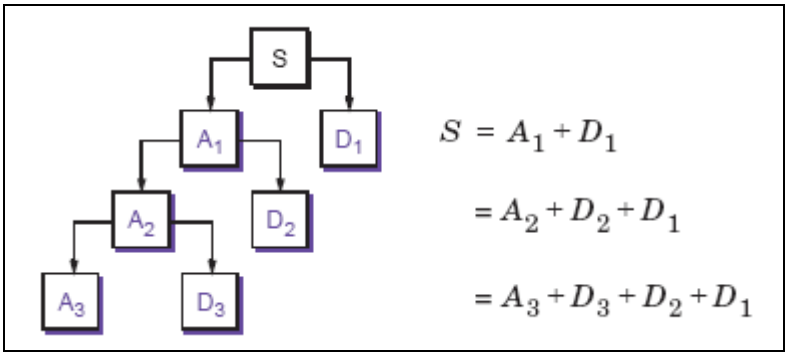

Figura 3.24: Decomposição em múltiplos níveis

3.3.4.7
Famílias wavelets

Em princípio, cada wavelet-mãe é mais apropriada para uma determinada aplicação, ou até para mais de uma. No entanto, há algumas famílias mais comumente usadas do que outras. Nesta seção, apresenta-se a wavelet de Haar e mais três famílias que, além de serem bastante usadas em diversas aplicações, foram utilizadas nesta dissertação.

Haar

A wavelet de Haar é a primeira e a mais simples. Ela é composta por uma função pulso unitário, sendo também chamada de wavelet-mãe Daubechies 1 – db_1.

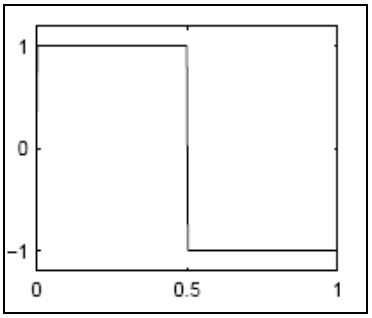

Figura 3.25: Wavelet Haar

Daubechies

Esta família foi criada por Ingrid Daubechies e tem como principal característica sua ortonormalidade e seu suporte compacto. O índice *n* em db_n indica a ordem, que teoricamente pode variar de 1 a infinito. Abaixo, seguem as daubechies de ordem 2 a 9, tendo em vista que a db_1 é igual à wavelet mãe de Haar.

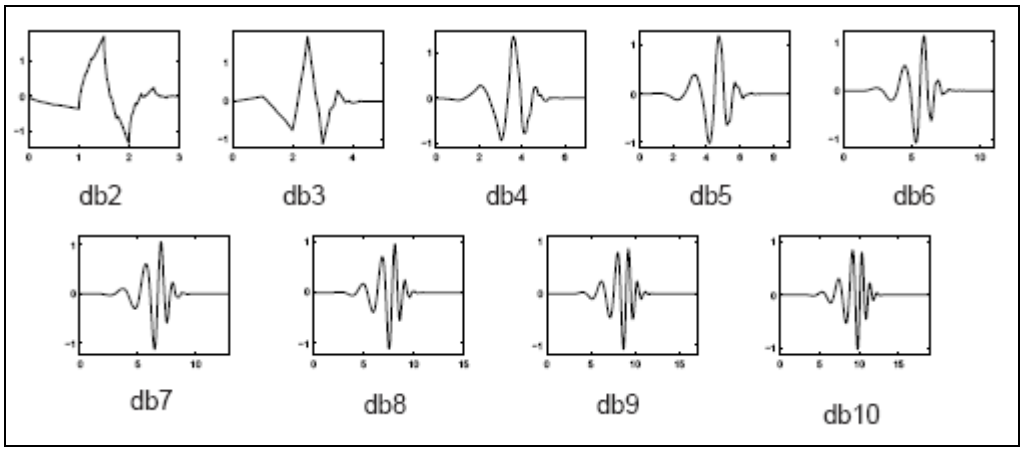

Figura 3.26: Família Daubechies

Biortogonal

Nas figuras 16 e 17 abaixo, as wavelets são apresentadas em pares. Em cada par, a wavelet da esquerda é a wavelet-mãe propriamente dita, a wavelet de decomposição, enquanto a da direita, é a wavelet de reconstrução (função de escalonamento).

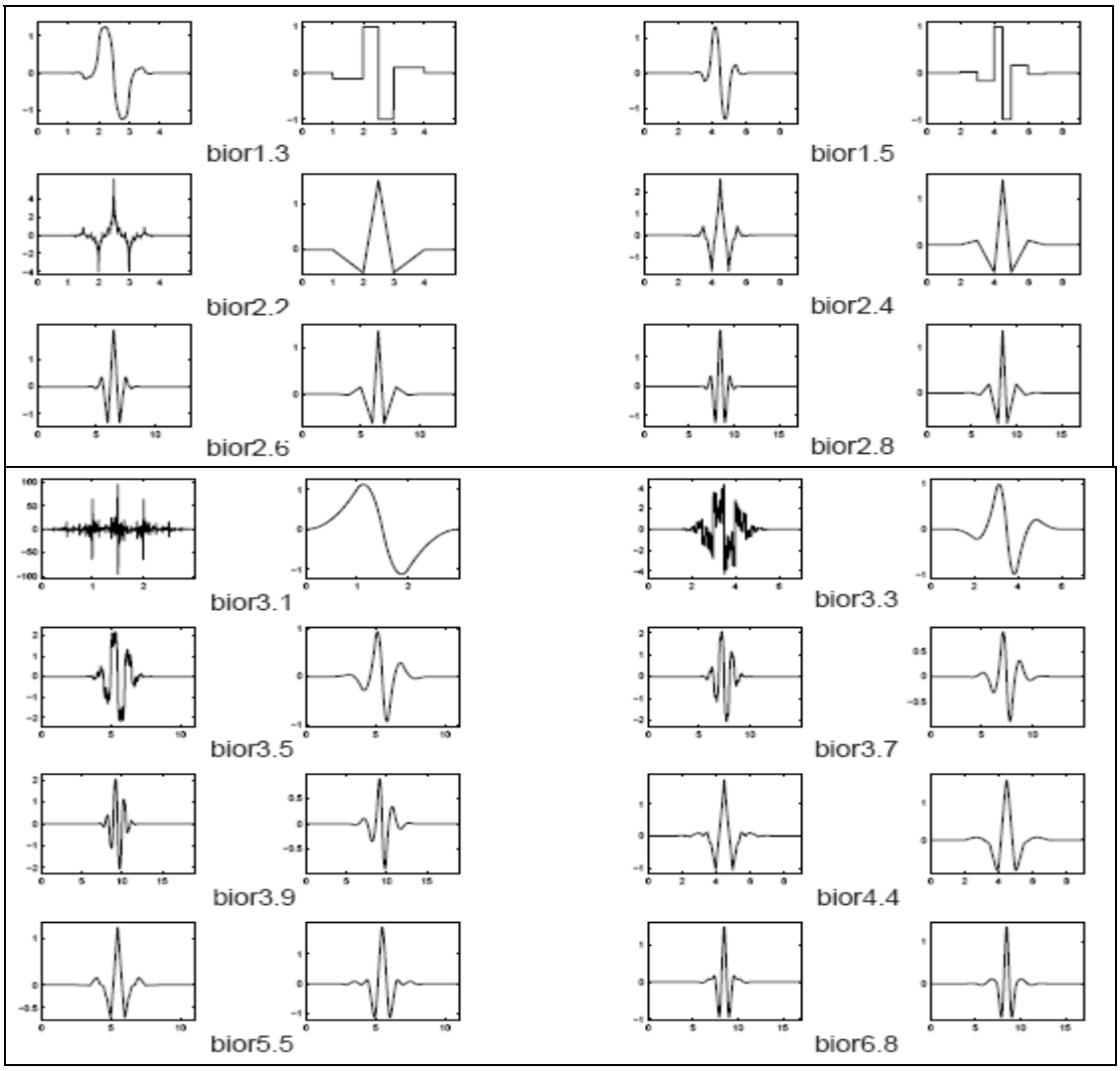

Figura 3.27: Família Biortogonal

Coiflets

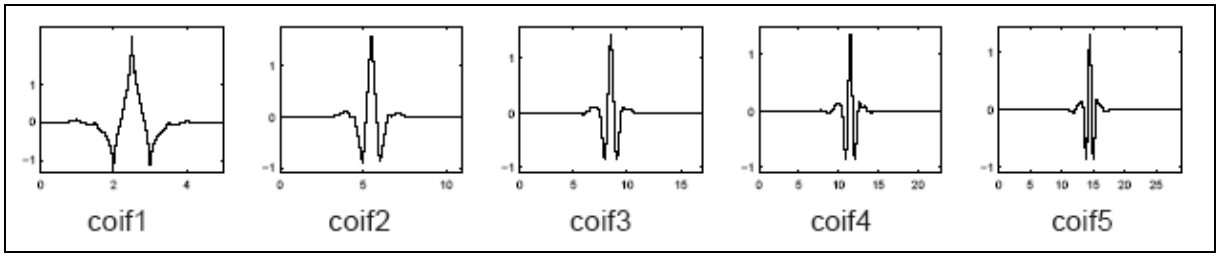

Figura 3.28: Família Coiflets