UNIVERSIDADE FEDERAL DO CEARÁ
DEPARTAMENTO DE ENGENHARIA DE TELEINFORMÁTICA
CURSO DE PÓS-GRADUAÇÃO EM ENGENHARIA DE TELEINFORMÁTICA

LUÍS GUSTAVO MOTA SOUZA

PROPOSIÇÃO E AVALIAÇÃO DE ALGORITMOS DE FILTRAGEM ADAPTATIVA BASEADOS NA REDE DE KOHONEN

FORTALEZA
2005

LUÍS GUSTAVO MOTA SOUZA

PROPOSIÇÃO E AVALIAÇÃO DE ALGORITMOS DE FILTRAGEM ADAPTATIVA BASEADOS NA REDE DE KOHONEN

Dissertação submetida à Coordenação do Curso de Pós-Graduação em Engenharia de Teleinformática, da Universidade Federal do Ceará, como parte dos requisitos exigidos para obtenção do grau de Mestre em Engenharia de Teleinformática.

Orientador: Prof. Dr. Guilherme de Alencar Barreto

Co-Orientador: Prof. Dr. João César Moura Mota

FORTALEZA

2005

FICHA CATALOGRÁFICA ELABORADA PELA
BIBLIOTECA DE PÓS-GRADUAÇÃO EM ENGENHARIA - BPGEE - UFC

S716p Souza, Luís Gustavo Mota

Proposição e avaliação de algoritmos de filtragem adaptativa baseados na rede de Kohonen / Luís Gustavo Mota Souza. 115f.:il.-

Orientador: Prof. Dr. Guilherme Alencar Barreto
Co-Orientador: Prof. Dr. João César Moura Mota
Dissertação (Mestrado) em Engenharia de Teleinformática - Universidade Federal do Ceará, Fortaleza, 2005.

1.Processamento de sinais 2. Mapas auto-organizáveis I. Título II. Universidade Federal do Ceará.

C.D.D. 621.382
C.D.U. 621.391

Esta Dissertação sob o título *Proposição e Avaliação de Algoritmos de Filtragem Adaptativa Baseados na Rede de Kohonen*, foi submetida como parte dos requisitos necessários à obtenção do Grau de Mestre em Engenharia de Teleinformática, outorgado pela Universidade Federal do Ceará, e encontra-se à disposição dos interessados na biblioteca da referida universidade.

A citação de qualquer trecho desta Dissertação é permitida, desde que feita de conformidade com as normas da ética científica.

Luís Gustavo Mota Souza

DISSERTAÇÃO APROVADA EM 02/06/2005.

Prof. Dr. Guilherme Alencar Barreto
Universidade Federal do Ceará - Orientador

Prof. Dr. João César Moura Mota
Universidade Federal do Ceará - Co-Orientador

Prof. Dr. Charles Casimiro Cavalcante
Universidade Federal do Ceará

Prof. Dr. Fernando José Von Zuben
Universidade de Campinas

Dedico este trabalho ao meu pai
Expedito(in memorium),
pelo seu amor
zeloso por mim, à minha amada mãe
Terezinha, aos meus irmãos Paulo
Sérgio, José Raimundo,
Expedito Filho e Cláudio,
pelos seus companheirismos,
e à minha amada noiva Carla,
pela sua compreensão e ajuda,
tão indispensáveis
ao longo deste período.

Agradecimentos

A Deus, o Senhor da minha vida.

Ao meu orientador e co-orientador, Profs. Guilherme de Alencar Barreto e João César Moura Mota, a estes sou grato pela orientação.

Aos professores e funcionários do Departamento de Engenharia de Teleinformática que de forma direta ou indireta participaram do desenvolvimento deste trabalho.

Aos demais colegas de graduação e pós-graduação, pelas críticas e sugestões.

À minha família pelo apoio durante esta jornada.

À FUNCAP (Fundação Cearense de Amparo à Pesquisa) por ter custeado meus estudos de mestrado, processo No.: 1068/04.

"Mestre não é quem ensina, mas
quem de repente aprende."

Guimarães Rosa

Resumo

A **Rede Auto-Organizável de Kohonen** (*Self-Organizing Map* - SOM), por empregar um algoritmo de aprendizado não-supervisionado, vem sendo tradicionalmente aplicada na área de processamento de sinais em tarefas de quantização vetorial, enquanto que redes MLP (*Multi-layer Perceptron*) e RBF (*Radial Basis Function*) dominam as aplicações que exigem a aproximação de mapeamentos entrada-saída. Este tipo de aplicação é comumente encontrada em tarefas de filtragem adaptativa que podem ser formatadas segundo a ótica da modelagem direta e inversa de sistemas, tais como identificação e equalização de canais de comunicação. Nesta dissertação, a gama de aplicações da rede SOM é estendida através da proposição de filtros adaptativos neurais baseados nesta rede, mostrando que os mesmos são alternativas viáveis aos filtros não-lineares baseados nas redes MLP e RBF. Isto torna-se possível graças ao uso de uma técnica recentemente proposta, chamada **Memória Associativa Temporal por Quantização Vetorial** (*Vector Quantized Temporal Associative Memory* - VQTAM), que basicamente usa a filosofia de treinamento da rede SOM para realizar a quantização vetorial simultânea dos espaços de entrada e de saída relativos ao problema de filtragem analisado. A partir da técnica VQTAM, são propostos três arquiteturas de filtros adaptativos baseadas na rede SOM, cujos desempenhos foram avaliados em tarefas de identificação e equalização de canais não-lineares. O canal usado nas simulações foi modelado como um processo auto-regressivo de Gauss-Markov de primeira ordem, contaminado com ruído branco gaussiano e dotado de não-linearidade do tipo saturação (sigmoidal). Os resultados obtidos mostram que filtros adaptativos baseados na rede SOM têm desempenho equivalente ou superior aos tradicionais filtros transversais lineares e aos filtros não-lineares baseados na rede MLP.

Palavras–Chave: Redes Neurais Artificiais, Aproximação de Funções, Filtragem Adaptativa, Rede Auto-Organizável de Kohonen, Quantização Vetorial.

Abstract

Kohonen's Self-Organizing Map (SOM), as an unsupervised learning approach, has been usually applied to vector quantization tasks within the field of signal processing, while supervised approaches, such as the MLP (*Multi-layer Perceptron*) and RBF (*Radial Basis Function*) networks, dominate those applications requiring learning of input-output mappings. This is the scenario typically found for adaptive filtering tasks that can be described under the framework of direct and inverse system modeling, such as identification and equalization of communication channels. In this dissertation, the range of applications of the SOM is enlarged through the proposal of SOM-based neural adaptive filters, showing that they are feasible alternatives to MLP- and RBF-based nonlinear filters. This is made possible by means of a recently proposed technique called *Vector Quantized Temporal Associative Memory* (VQTAM), which basically uses the SOM to perform vector quantization of the input and output data spaces simultaneously. Based on the VQTAM, three SOM-based adaptive filter topologies are proposed and their performance are evaluated in identification and equalization of nonlinear channels. The simulated channel was modeled as a first order Gauss-Markov autoregressive process, contaminated with white gaussian noise and containing a sigmoidal nonlinearity. The obtained results demonstrate that the proposed SOM-based adaptive filters have equivalent or superior performance in comparison to standard transversal linear filters and the MLP-based nonlinear filter.

Keywords: Artificial Neural Networks, Function Approximation, Adaptive Filtering, Self-Organizing Map, Vector Quantization.

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas e Siglas

ANN	*Adaptive Neural Network*
BER	*Bit Error Rate*
EEG	Eletro-encefalograma
EG	Erro de Generalização
XOR	*eXclusive-OR*
FIR	*Finite Impulse Response*
GRNN	*Generalized Regression Neural Network*
GRBF	*Global Radial Basis Function*
IIR	*Infinite Impulse Response*
KRBF	*Local RBF over K-winners*
KSOM	*Local Linear Mapping over K-winners*
LMS	*Least-Mean-Squares*
LLM	*Linear Local Mapping*
LESSOM	*Local Least-Squares SOM*
MSE	*Mean-Squared Error*
MPNN	*Modified Probabilistic Neural Network*
MLP	*Multi-layer Perceptron*
MIMO	*Multiple-Input Multiple-Output*
NMSE	*Normalized Mean-Squared Error*
RBF	*Radial Basis Function*
RNN	*Recurrent Neural Network*
RNA	Redes Neurais Artificiais
SOM	*Self-Organizing Map*
SISO	*Single-Input Single-Output*
VQTAM	*Vector-Quantized Temporal Associative Memory*
WTA	*Winner Take All*

Lista de Símbolos

t	Tempo discreto
α	Taxa de aprendizagem usada pelos filtros lineares e não-lineares, e na etapa de quantização vetorial da rede LLM
α'	Taxa de aprendizagem do algoritmo LMS na rede LLM
$h(i^*, i; t)$	Função de vizinhança da rede SOM
γ_i	Largura da função de ativação gaussiana do neurônio i na rede RBF
$\sigma(t)$	Abertura da vizinhança topológica da função gaussiana
σ_0, σ_T	Valores inicial e final de $\sigma(t)$, respectivamente
$\mathbf{x}(t)$	Vetor de entrada da rede no instante t
N	Tamanho total da seqüência de símbolos gerada
$\mathcal{X}$	Espaço contínuo dos dados de entrada
$p(\mathbf{x})$	Densidade de probabilidade do vetor de entrada $\mathbf{x}(t)$
$\mathcal{A}$	Topologia do espaço de saída
Φ	Transformação não-linear
$\mathbf{w}_i(t)$	Vetor de pesos associado ao neurônio i
$\mathbf{a}(t)$	Vetor de coeficientes do filtro transversal linear
$\mathbf{R}$	Matriz de auto-correlação do vetor de entrada
$\mathbf{p}$	Vetor de correlação cruzada entre os vetores de entrada e a saída desejada
$\{x(t)\}$	Seqüência de símbolos de dados de entrada
$d(t)$	Saída desejada
$\hat{y}(t)$	Saída estimada
$e(t)$	Erro de aproximação
p	Comprimento do filtro transversal e do vetor de entrada
$F[\cdot]$	Função desconhecida a ser aproximada
$\hat{F}[\cdot]$	Aproximação da função desconhecida $F[\cdot]$
q	Número de neurônios existentes na camada escondida da rede MLP
M	Número de neurônios usados na camada de saída da rede MLP
$\phi(\cdot)$	Função de ativação não-linear dos neurônios nas redes neurais
$\delta(t)$	Gradiente local da rede MLP
σ_e^2	Variância do erro
σ_y^2	Variância do sinal de saída

Sumário

1 CONCEITOS GERAIS

1.1 Introdução

Os modernos sistemas de telecomunicações e a conseqüente necessidade de transmitir sinais de diferentes modalidades (voz, texto e imagem), de maneira rápida e com qualidade, têm demandado técnicas de processamento de sinais cada vez mais sofisticadas. Por exemplo, o rápido avanço das comunicações sem-fio, tais como sistemas celulares e via satélite, tem exigido a transmissão de uma quantidade de voz e dados cada vez maior por unidade de tempo em meios de transmissão com banda limitada e, muitas vezes, sob uma restrição de tempo de transmissão severa.

Sabe-se que o canal de comunicação, seja com ou sem fio, distorce de diferentes modos os sinais (símbolos) transmitidos, tanto na amplitude quanto na fase. As causas destas distorções são as mais variadas, sendo que algumas têm origem endógena, ou seja, na própria dinâmica do canal (resposta ao impulso), no ruído térmico dos semi-condutores, enquanto outras têm origem exógena, tais como interferência eletromagnética e do tipo co-canal.

Em particular, pode-se também listar distorções de caráter fortemente não-linear que são inseridas no sistema de comunicação através (IBNKAHLA; CASTANIE, 1995; ERDOGMUS et al., 2001):

- da saturação dos amplificadores de potência no lado do transmissor;
- dos conversores Analógico-Digital (A/D) e Digital-Analógico (D/A);
- dos processos de modulação e demodulação;
- e da própria natureza dinâmica do meio de propagação (e.g. canal via satélite, fibra ótica, etc.).

Do exposto acima, pode-se afirmar que, dependendo do tipo específico de canal ou de tecnologia de transmissão utilizada, uma ou outra distorção pode estar mais em evidência que outras. De qualquer modo, a fim de oferecer todos os serviços disponíveis

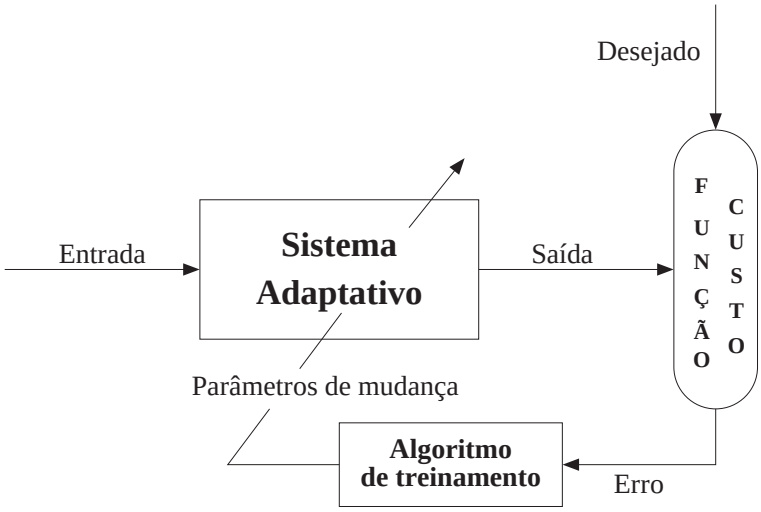


Figura 1.1 Projeto do sistema adaptativo.

com qualidade, deve-se lançar mão de ferramentas práticas que mantenham, de forma confiável, a transmissão/recepção de sinais[1] através do sistema de comunicação, permitindo ao usuário desfrutar de todos os recursos existentes nas tecnologias atuais, tais como acesso à Internet via aparelhos celulares, canais de televisão por assinatura e sistemas de posicionamento global.

O efeito nocivo das distorções sobre o sinal transmitido pode ser aliviado através do uso de técnicas de processamento de sinais, que têm como objetivo último restaurar no receptor a informação presente no sinal transmitido. Diversas técnicas de processamento de sinais têm sido propostas ao longo dos anos, com sua evolução atrelada ao surgimento de novas tecnologias de telecomunicações. De particular interesse para esta dissertação são as técnicas ou modelos ditos *adaptativos*, sejam lineares ou não-lineares.

A característica principal destas técnicas está na sua capacidade de rastrear (modelar), de forma automática, o comportamento dinâmico do canal de transmissão, mesmo que não se tenha informação prévia sobre este, permitindo a recuperação total ou parcial da informação transmitida.

A introdução de conceitos de sistemas adaptativos na Engenharia de Sistemas introduziu um estilo totalmente novo de projeto. Em vez de ser construído através da especificação prévia do comportamento dinâmico e estatístico de todo os subsistemas que compõem um canal de comunicações (ver, por exemplo, o livro de Wiener (1949)), sistemas adaptativos "aprendem a imitar" o comportamento do canal de comunicação em tempo-real, simplesmente com base no sinal transmitido e no sinal recebido, através do ajuste de alguns parâmetros específicos.

Tomando como base a Figura 1.1, pode-se perceber que sistemas adaptativos ajustam

[1] O termo *sinais* é aqui usado de forma genérica para denotar tanto voz quanto dados.

seus parâmetros com base na qualidade da *saída* gerada através de um *laço de realimentação de desempenho* que inclui um mecanismo de avaliação, geralmente implementado na forma de uma *função custo*. A realimentação de desempenho é utilizado diretamente para mudar os parâmetros através de procedimentos sistemáticos chamados de *regras de aprendizagem* ou *treinamento*, para que a saída do sistema melhore com respeito ao *objetivo desejado*, ou seja, que o erro gerado diminua com o passar do tempo.

Em sistemas adaptativos, os parâmetros do sistema são continuamente adaptados durante a operação com o sinal de entrada atual. No começo do processo de ajuste dos parâmetros, estes elementos do sistema adaptativo podem apresentar resultados não satisfatórios que estejam distantes dos valores reais que melhor descrevem a dinâmica do canal, gerando um erro elevado. Entretanto, com a realimentação do erro que é fornecido ao sistema, esta estrutura passa a ter um guia indicando em que direção mudar seus parâmetros para diminuir o erro tanto quanto possível. É importante enfatizar também que, mesmo que o sinal mude suas estatísticas com o passar do tempo, a metodologia subjacente ao projeto de sistemas adaptativos permitirá a alteração dos parâmetros do sistema de modo que o melhor desempenho *possível* seja obtido.

Em suma, os principais elementos envolvidos no projeto dos sistemas adaptativos de interesse para esta dissertação são:

- um sistema (modelo matemático) com parâmetros adaptativos;
- a existência de uma resposta desejada ou alvo d;
- um critério de avaliação do erro, a ser otimizado (na verdade, minimizado!); e
- um mecanismo para modificar os parâmetros do sistema adaptativo continuamente, na direção dos parâmetros ótimos que correspondem ao ponto de menor erro possível.

Os elementos mencionados acima auxiliarão na compreensão do modo como modelos de sistemas adaptativos são aplicados às tarefas de filtragem adaptativa de interesse para esta dissertação, ou seja, identificação e equalização de canais. Além disso, para lidar com os problemas pertinentes aos canais de comunicação listados anteriormente, em particular os de natureza não-linear, se faz necessária o emprego dos modelos adaptativos não-lineares, tais como aqueles implementados por redes neurais artificiais (PRINCIPE et al., 2000; HAYKIN, 1996).

A próxima seção faz uma breve exposição sobre a implementação de sistemas adaptativos voltados para processamento de sinais, dando margem ao aparecimento de técnicas de filtragem adaptativa.

1.2 Uma Breve Introdução a Filtragem Adaptativa

Sistemas adaptativos demandam um mínimo de conhecimento *a priori* acerca dos sinais ou sistemas sob análise. As origens dos modernos sistemas adaptativos remontam às tentativas de projetar máquinas de aprendizagem linear e não-linear, tais como o Adaline (*Adaptive Linear Element*) e o Perceptron, por volta da década de 1960 (ROSENBLATT, 1958; WIDROW; HOFF, 1960; WIDROW, 1962; GABOR et al., 1960).

Estes também foram efetivamente os estágios iniciais do desenvolvimento das Redes Neurais Artificiais (ANNs). É importante destacar que este esforço inicial foi quase abandonado em razão da publicação do trabalho de Minsky & Papert (1969) que demonstrou que máquinas adaptativas simples não eram capazes de aprender certos problemas elementares, tal como o *OU-exclusivo* (XOR), e assim não seriam capazes de simular adequadamente mecanismos de reconhecimento de padrões e memória tal como observado no cérebro humano.

Apesar disso, o uso de máquinas adaptativas simples (e.g. Adaline) foi sendo gradativamente absorvido pela emergente indústria de Telecomunicações (em franca expansão devido ao impulso provocado pelas duas grandes guerras), enquanto alguns poucos pesquisadores da área de redes neurais artificiais continuaram suas pesquisas com máquinas adaptativas não-lineares. Somente a partir de meados da década de 1980, com a proposição do Algoritmo de **Retropropagação do Erro** (*Error Backpropagation*) é que os dois campos, redes neurais artificiais e filtragem adaptativa, voltaram a se envolver com mais intensidade (WIDROW; LEHR, 1990).

A seguir, são apresentados algumas das principais idéias, conceitos e algoritmos relacionados à filtragem adaptativa linear, discutindo sua estreita relação com a teoria de estimação linear ótima. Algoritmos de filtragem não-linear baseados em redes neurais serão alvo de discussão em seções posteriores.

1.2.1 Estimação Linear Ótima

O início de um estudo sistemático sobre estimação de parâmetros surgiu com estudos sobre o movimento dos planetas e cometas iniciados por Galileo Galilei em 1632, para minimizar as várias funções de erro obtidas então. Contudo, a origem da teoria de estimação linear é atribuída a Gauss que, na idade de 18 anos em 1795, introduziu o método dos *Quadrados Mínimos* (QM) para estudar o movimento de corpos celestes (AGUIRRE, 2000).

Segundo mencionado em Haykin (1996), os primeiros estudos sobre a estimação média quadrática mínima em processo estocástico foram feitos por Kolmogorov, Krein e Wiener

durante o final da década de 1930 e o começo da década de 1940. Com o trabalho nesta área sendo desenvolvido e aperfeiçoado em paralelo entre os dois primeiros e Wiener, então este, independentemente, formulou o problema de predição linear em tempo contínuo e derivou uma fórmula explícita para o estimador ótimo. Wiener também considerou a filtragem do problema de estimação um processo corrompido por um ruído aditivo. A fórmula explícita para a estimação ótima requisitou a solução de uma equação integral conhecida como a equação de Wiener & Hopf (1931).

Em 1947, Levinson formulou o problema de filtragem de Wiener em tempo discreto. No caso de sinais discretos no tempo, a equação de Wiener-Hopf é escrita em forma matricial como segue

$$\mathbf{R}\mathbf{a}_0 = \mathbf{p}, \tag{1.1}$$

em que $\mathbf{a}_0$ é o vetor de coeficientes do filtro de Wiener ótimo estruturado na forma de um filtro transversal, $\mathbf{R} = E[\mathbf{x}\mathbf{x}^T]$ é a matriz de auto-correlação do vetor de entrada $\mathbf{x}$, e $\mathbf{p} = E[d\mathbf{x}]$ é o vetor de correlação cruzada entre o vetor de entrada e a resposta desejada d correspondente. Os coeficientes ótimos são então calculados pela inversão da matriz $\mathbf{R}$, ou seja, $\mathbf{a}_0^* = \mathbf{R}^{-1}\mathbf{p}$. A Equação de Wiener-Hopf exige o conhecimento dos momentos estatísticos dos sinais de entrada e saída e, portanto, não se apresenta como um modelo adaptativo.

1.2.2 Sistemas Adaptativos

Um sistema adaptativo genérico pode ser definido como a estrutura representada na Figura 1.2. Perceba que esta figura é uma versão particular da Figura 1.1, em que o sistema adaptativo passa a ser chamado de *filtro ajustável* (que pode ser linear ou não), a função de avaliação do desempenho reduz-se a uma medida de erro, e o algoritmo de treinamento torna-se equivalente ao algoritmo de adaptação (ajuste) dos parâmetros do filtro adaptativo. Nesta figura, para cada instante de tempo discreto t, apresenta-se uma janela de valores $\{x(t)\}$ com comprimento p à entrada do filtro e calcula-se a resposta do mesmo, $\hat{y}(t) \in \mathbb{R}$, que é então comparada com a saída desejada, $d(t) \in \mathbb{R}$, para gerar o sinal de erro, $e(t)$, que irá guiar o processo de modificação dos parâmetros.

O sinal de erro é fornecido junto com a seqüência de entrada para o algoritmo de adaptação dos parâmetros. A tarefa deste algoritmo é ajustar iterativamente os parâmetros do filtro $\mathbf{a}(t)$ com o intuito de reduzir progressivamente o erro de saída de acordo com alguma função custo (por exemplo, o erro quadrático médio para um dado horizonte de tempo).

Do exposto na Figura 1.2, dois problemas de estimação são aqui colocados: (i) estimação (predição) da saída e (ii) estimação dos parâmetros ótimos do filtro. O primeiro caso

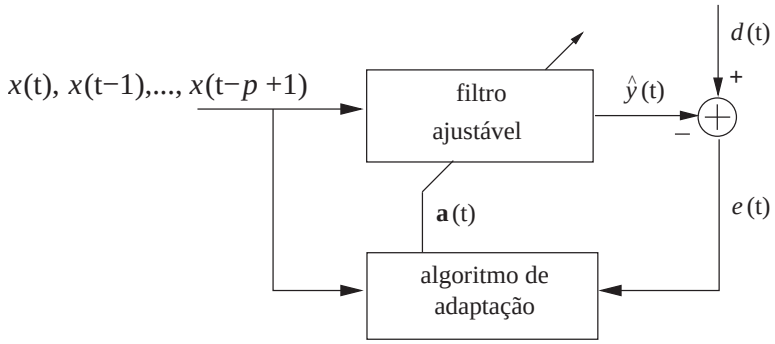


Figura 1.2 Estrutura genérica de um filtro adaptativo.

está associado à capacidade de aprendizado do filtro, ou seja, ao seu poder computacional como aproximador da resposta desejada $d(t)$, que depende do tipo de modelo matemático escolhido para construí-lo, podendo ser linear ou não-linear. Também relacionado ao primeiro tópico está o tipo de modelo dinâmico a ser adotado para o filtro: resposta ao impulso finita (FIR) ou infinita (IIR).

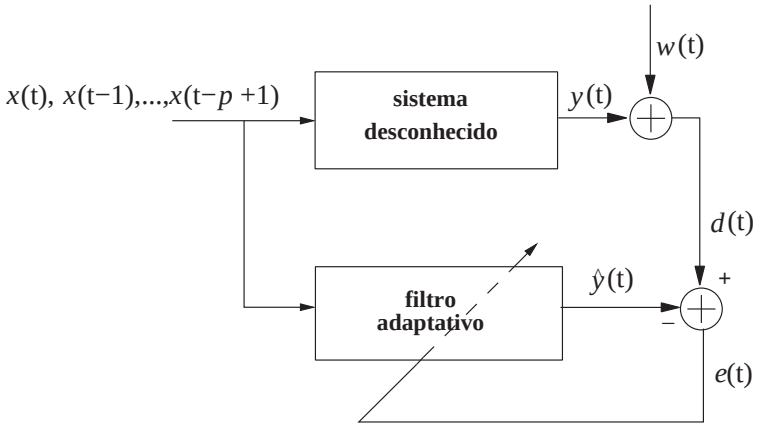
Esta dissertação se limita a estudar filtros FIR. Este capítulo em particular discute basicamente filtros lineares. Filtros não-lineares são apenas mencionados de passagem, sendo os detalhes deixados para os próximos capítulos. Com relação ao segundo problema de estimação dos parâmetros ótimos de um filtro FIR, esta dissertação adota procedimentos recursivos de minimização do *Erro Quadrático Médio* (MSE) entre a resposta do filtro e a saída desejada para o mesmo.

Na próxima seção, são apresentados argumentos gerais para contextualizar a tarefa de filtragem adaptativa como equivalente à tarefa mais abrangente de *identificação* (*ou modelagem caixa-preta*) *de sistemas*. Muito do material a ser exposto a seguir foi extraído do artigo de Johnson (1995).

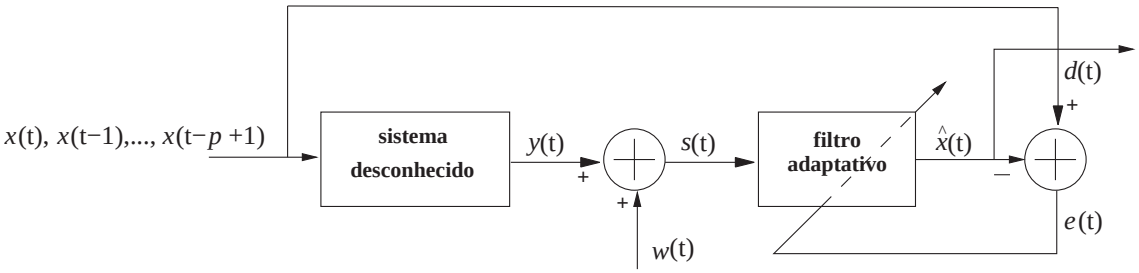
1.2.3 Filtragem Adaptativa como Identificação de Sistemas

Identificação de sistemas, também chamada de *Modelagem Caixa-Preta*, é a área de estudo que provê ferramentas matemático-computacionais para aprender algumas (senão todas!) características dinâmicas de sistemas desconhecidos, com base apenas nos sinais de entrada e de saída (AGUIRRE, 2000). Assim, de modo abrangente, os diagramas mostrados na Figura 1.3 correspondem às tarefas gerais de interesse para esta dissertação, ou seja, aplicação de estruturas (filtros) adaptativas na modelagem direta e inversa de sistemas de comunicações.

As tarefas de modelagem direta e inversa às quais o parágrafo anterior se refere são comumente chamadas de identificação e equalização de canais. No primeiro caso, o filtro adaptativo da Figura 1.2 tenta "imitar" a função de transferência do canal (ou seja, sua



(a) Configuração de modelagem direta de sistemas.



(b) Configuração de modelagem inversa de sistemas.

Figura 1.3 Exemplos de estruturas de aplicação de filtragem adaptativa.

relação entrada-saída), enquanto que no segundo caso, a relação entrada-saída inversa deve ser modelada.

Na Figura 1.3(a), um sinal discreto qualquer $\{x(t), x(t-1), \ldots, x(t-p+1)\}$ é aplicado simultaneamente ao *canal* e ao *filtro*, tal que com o passar do tempo a saída do filtro $\hat{y}(t)$, vai se aproximando do valor da saída do *canal de comunicação*, $d(t)$, passando o filtro a se comportar como o canal. É importante enfatizar aqui a presença de ruído na formação da saída desejada $d(t)$, indicando incertezas de modelagem e/ou medição. Modelos de canais são úteis, por exemplo, no projeto de receptores, simulações computacionais para avaliação de desempenho de canais de comunicação, além de detecção e diagnóstico de falhas.

Na estrutura correspondente à equalização adaptativa (Figura 1.3(b)), tem-se que o filtro está agora em série com o sistema desconhecido. Se este for um canal de comunicação, então o processador tenta recuperar o sinal apresentado, cujos componentes (símbolos) foram distorcidos pelas características dinâmicas do canal e/ou por ruído aditivo. Após a convergência, a função de transferência do filtro tende a aproximar a função de transferência inversa do canal.

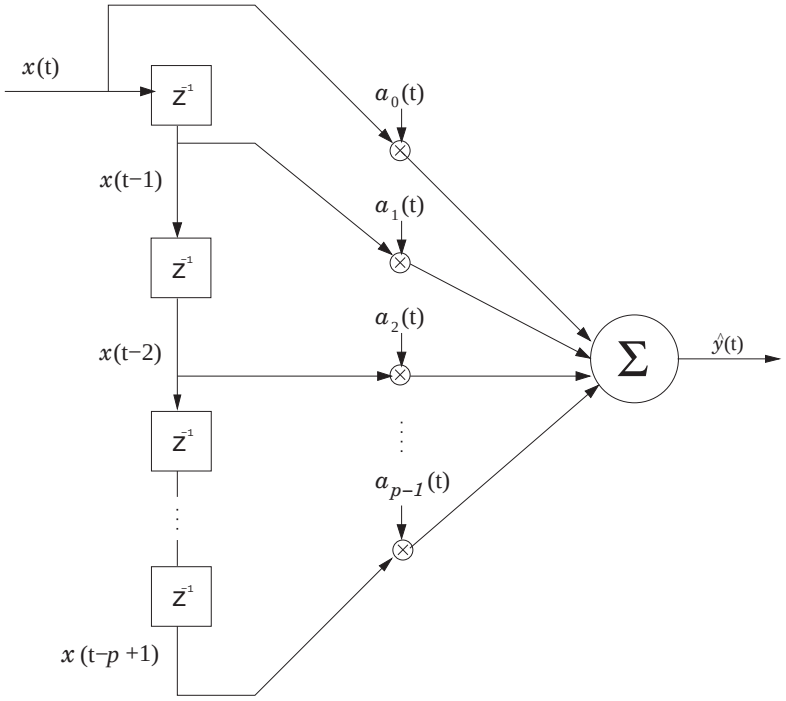


Figura 1.4 Filtro transversal com os coeficientes dependentes do tempo.

A próxima seção faz uma breve exposição sobre o modelo matemático básico de um filtro adaptativo linear e sobre duas das principais estratégias de ajuste dos parâmetros de tal filtro.

1.3 Filtros Transversais Lineares Adaptativos

O filtro ajustável adaptativo mostrado na Figura 1.2 assume a forma de um filtro transversal de comprimento p, como representado na Figura 1.4. Para este filtro, em cada instante de tempo t, a saída $\hat{y}(t)$ é calculada pela combinação linear das entradas atual e atrasada $x(t), x(t-1), x(t-2), \ldots, x(t-p+1)$, na forma:

$$\hat{y}(t) = \sum_{j=0}^{p-1} a_j(t)x(t-j), \tag{1.2}$$

em que as variáveis $a_j(t)$ definem os coeficientes do filtro. A Equação (1.2) pode ser escrita na forma vetorial, usando $\mathbf{x}(t) = [x(t), x(t-1), \ldots, x(t-p+1)]^T$ para indicar o vetor de entrada no instante de tempo t, e $\mathbf{a}(t) = [a_0(t), a_1(t), \ldots, a_{p-1}(t)]^T$, o vetor de coeficientes no mesmo instante de tempo

$$\hat{y}(t) = \mathbf{a}^T(t)\mathbf{x}(t), \tag{1.3}$$

em que $(\cdot)^T$ indica o vetor transposto. No caso especial dos coeficientes $\mathbf{a}(t)$ não dependerem do tempo t, i.e. $\mathbf{a}(t) = \mathbf{a}$, a estrutura do filtro transversal corresponde a um filtro

FIR de comprimento p. Nesta dissertação, está-se interessado apenas no caso em que os coeficientes do filtro são variáveis e são corrigidos por um algoritmo de aprendizado.

1.3.1 O Algoritmo de Adaptação LMS

O algoritmo de adaptação mais popular até os dias de hoje, é chamado de LMS (*Least-Mean-Square*), conhecido também por *Regra Delta* ou a *Regra de Widrow-Hoff* (WIDROW; HOFF, 1960). O algoritmo LMS foi originalmente formulado para aplicação em circuitos adaptativos, usando-se vários filtros lineares do tipo FIR, formando as chamadas redes MADALINE (*Multi-ADALINE*) (WIDROW; LEHR, 1990).

Seja $d(t)$ a *resposta desejada* ou a *saída alvo* para o filtro, no tempo t. Pode-se então definir o *sinal erro*, como segue

$$e(t) = d(t) - \hat{y}(t). \tag{1.4}$$

Como medida de desempenho ou função custo utiliza-se o *erro quadrático* definido como

$$J(t) = \frac{1}{2}\sum_{t=1}^{N} e^2(t), \tag{1.5}$$

em que N é o comprimento total da seqüência de símbolos $\{x(t)\}$. Assim, pode-se definir o *problema de filtragem linear ótima* dado a seguir.

> *Deve-se determinar o conjunto de coeficientes ótimos* $a_0^*, a_1^*, \ldots, a_j^*, \ldots, a_{p-1}^*$ *para que o erro quadrático J(t) seja mínimo.*

Para isto, utiliza-se o *Método do Gradiente Estocástico* (HAYKIN, 1996), que consiste na aplicação de uma equação de atualização recursiva para cada coeficiente $a_j(t)$

$$a_j(t+1) = a_j(t) - \alpha\frac{\partial J(t)}{\partial a_j(t)}. \tag{1.6}$$

Assim, utilizando a regra da cadeia para obter $\frac{\partial J(t)}{\partial a_j(t)}$, chega-se à seguinte expressão

$$\frac{\partial J(t)}{\partial a_j(t)} = \frac{dJ(t)}{de(t)}\frac{de(t)}{d\hat{y}(t)}\frac{d\hat{y}(t)}{da_j(t)} = -e(t)x(t-j). \tag{1.7}$$

Substituindo o resultado da Equação (1.7) na Equação (1.6), tem-se o seguinte resultado

$$a_j(t+1) = a_j(t) + \alpha e(t)x(t-j), \qquad j = 0, \ldots, p-1, \tag{1.8}$$

na qual o termo $0 < \alpha < 2/\lambda_{\max}$ (HAYKIN, 1996) define o passo de adaptação (ou aprendi-

zagem), em que $\lambda_{\max}$ é o maior autovalor da matriz de correlação dos vetores de entrada, $\mathbf{R} = E[\mathbf{x}\mathbf{x}^T]$. A forma vetorial da Equação (1.8) é a seguinte

$$\mathbf{a}(t+1) = \mathbf{a}(t) + \alpha e(t)\mathbf{x}(t). \tag{1.9}$$

Note que o algoritmo LMS utiliza apenas informação disponível localmente sobre a entrada $x(t-j)$ para ajustar o coeficiente $a_j(t)$ correspondente. A seguir, é apresentado um algoritmo de adaptação que faz uso de informação global sobre a entrada para atualizar cada coeficiente.

1.3.2 O Algoritmo de Adaptação LMS/Newton

Um outro algoritmo de adaptação bastante usado é conhecido como LMS/Newton (HAYKIN, 1996; WIDROW; STEARNS, 1985; DINIZ et al., 1995; FARHANG-BOROUJENY, 1997; WIDROW; KAMENETSKY, 2003). A equação recursiva de adaptação dos coeficientes $a_j(t)$ para este método é dada por

$$\mathbf{a}(t+1) = \mathbf{a}(t) - \alpha\mathbf{R}^{-1}(t)\frac{\partial J(t)}{\partial \mathbf{a}(t)}. \tag{1.10}$$

Deste modo, a equação recursiva para atualização dos coeficientes do filtro FIR transversal passa a ser dada por

$$\mathbf{a}(t+1) = \mathbf{a}(t) + \alpha\mathbf{R}^{-1}(t)e(t)\mathbf{x}(t). \tag{1.11}$$

Comparando esta equação com a Equação (1.9) pode-se notar que a informação do gradiente é multiplicada pela inversa da matriz de auto-correlação da entrada. Se a matriz $\mathbf{R}$ não for conhecida *a priori* pode ser também estimada recursivamente pela seguinte expressão (WIDROW; STEARNS, 1985; ABRANTES, 2000):

$$\mathbf{R}(t+1) = (1-\beta)\mathbf{R}(t) + \beta\mathbf{x}(t)\mathbf{x}^T(t), \tag{1.12}$$

em que a constante $0 < \beta < 1$ é chamada de *fator de esquecimento*. No início do processo de estimação, em geral, adota-se $\mathbf{R}(0) = \mathbf{I}_p$, sendo $\mathbf{I}_p$ a matriz de identidade de dimensão $p \times p$.

Nota-se que, para implementar o algoritmo LMS/Newton, é preciso calcular a inversa da matriz de auto-correlação, que demanda significativamente mais custo computacional do que a única multiplicação requerida pelo método LMS. Além disso, o método LMS/Newton também requer informação global, ou seja, a atualização de um coeficiente $a_j(t)$ depende de todas as entradas $x(t-j)$, $j = 0, \ldots, p-1$, diferentemente do LMS que utiliza apenas informação de uma única entrada por coeficiente.

Do ponto de vista geométrico, o método LMS/Newton corrige os coeficientes ótimos

sempre na direção do mínimo de $J(t)$, enquanto o gradiente descendente aponta para a direção de máxima variação de $J(t)$. Estas duas direções podem não coincidir. Elas coincidem quando as curvas de nível ou de contorno[2] da superfície de desempenho gerada por $J(t)$ são circulares, ou seja, quando o maior e o menor autovalor da matriz de autocorrelação $\mathbf{R}$ são os mesmos.

Quando a razão do maior pelo menor autovalor, chamado de **Espalhamento dos Autovalores** (*eigenvalue spread*) aumenta, tem-se que, em vez de um círculo, a curva será uma elipse com a sua forma alongada na direção de maior variância dos dados. Então, para maiores espalhamentos do autovalor o caminho de otimização tomado pelo gradiente descendente é normalmente muito mais longo do que o caminho tomado pelo método de Newton. Isto implica que o método LMS/Newton, em geral, atinge a vizinhança do valor ótimo dos coeficientes $\mathbf{a}_0^*$ (ver Eq. (1.1)) mais rapidamente que o algoritmo LMS quando a matriz de auto-correlação do vetor de entrada tem o espalhamento dos autovalores alto.

1.3.2.1 Considerações sobre a Aplicação dos Algoritmos LMS e LMS/Newton

Os algoritmos de adaptação LMS e LMS/Newton se aplicam com eficiência a canais de comunicação lineares, estacionários e com ruído aditivo gaussiano (HAYKIN, 1996). Ao tratar sinais que não obedecem estritamente a tais condições, estes algoritmos geralmente apresentam resultados não muito satisfatórios (REUTER; ZEIDLER, 1999), ou seja, os coeficientes podem convergir para uma solução não-ótima ao final do processo de treinamento. Isto ocorre, devido a solução poder cair em um mínimo local ou devido a hipótese de modelo ser falha e a adaptação de parâmetros, embora bem-sucedida (ou seja, buscando atingir o ótimo global, ver Eq. (1.1)), não consiga atender os objetivos, sendo portanto, a segunda hipótese mais possível de acontecer pois os modelos são lineares. Em outras palavras, os coeficientes convergem sim para uma solução ótima global, mas ela representa uma proposta ruim.

Conforme mencionado na introdução deste capítulo, canais de comunicações, tais como os sistemas móveis e de satélite, podem apresentar um certo grau de não-linearidade e não-estacionariedade. Tais canais são encontrados facilmente na prática (HAYKIN, 1996; IBNKAHLA et al., 1998; IBNKAHLA, 2000), graças a fenômenos já mencionados, tais como saturação dos amplificadores no transmissor, introduzindo distorção não-linear na amplitude e fase do sinal transmitido, causando falta de sincronização dos dados enviados e recebidos, implicando na degradação do desempenho do filtro, conforme avaliado por várias figuras de mérito, tais como **Taxa de Erro de Bit** (BER), **Erro Quadrático Médio** (MSE) e velocidade de convergência para a solução ótima.

[2] São curvas que ligam todos os pontos com o mesmo valor de J (J = constante). A curva de contorno para J é formado por elipsóides concêntricas (elipses para o caso 2-D).

Esta dissertação aborda o problema da não-linearidade através do emprego de filtros adaptativos implementados por arquiteturas de redes neurais artificiais, comparando o desempenho deste tipo de filtro não-linear com o desempenho dos filtros lineares convencionais. Por outro lado sabe-se que estes filtros não-lineares apresentam também algumas desvantagens, tais como: (*i*) existência de mínimos locais (não existente no caso linear), e (*ii*) maior custo para convergência até o conjunto de parâmetros ótimo (comparando-se ao caso linear). A seguir, é feita uma breve introdução ao uso de filtros não-lineares baseados em redes neurais artificiais.

1.4 Filtros Não-Lineares e Motivações desta Dissertação

Dentro do conjunto dos filtros adaptativos não-lineares destacam-se as redes neurais artificiais (RNAs) por serem capazes de aprender mapeamentos entrada-saída complexos (e.g. não-lineares). Existem outros modelos de filtros não-lineares que se destacam bem, como exemplo tem-se o filtro de Volterra. Por isto, redes de diferentes arquiteturas têm sido amplamente utilizadas em problemas de identificação e equalização adaptativa de canais, em particular de canais não-lineares (HAYKIN, 1996; IBNKAHLA, 2000).

Grosso modo, uma rede neural é formada pela interconexão de um grande número de unidades de processamento não-linear denominadas *neurônios* que estão organizados em camadas. Pelo menos duas camadas de neurônios são necessárias, sendo estas chamadas de camada de entrada e de camada de saída. Algumas arquiteturas de redes neurais possuem uma ou mais camadas de neurônios entre as camadas de entrada e de saída.

Uma apresentação mais detalhada sobre as arquiteturas de redes neurais de interesse para este trabalho é deixada para o próximo capítulo. Contudo, pode-se adiantar que o interesse por redes neurais como ferramentas matemático-computacionais para o processamento digital de sinais e conseqüente aplicação em telecomunicações, no caso de sistemas do tipo SISO, é justificada por uma série de características tais como: (*i*) não-linearidade inerente, (*ii*) capacidade de aprendizado, (*iii*) capacidade de generalizar o conhecimento adquirido, e (*iv*) baixa exigência quanto ao conhecimento das estatísticas dos sinais de entrada e de saída.

Em particular, a natureza não-linear do mapeamento de entrada-saída realizado pelas redes neurais é de grande utilidade para esta dissertação, pois assume-se que os mecanismos físicos responsáveis por causar algumas das distorções nos sinais transmitidos são de natureza não-linear.

Dentre as diversas arquiteturas existentes, duas redes de aprendizado supervisionado

se destacam pela larga utilização em filtragem adaptativa não-linear: a rede **Perceptron Multicamadas** (MLP) e a rede com **Função de Base Radial** (RBF), por serem *aproximadores universais de funções* (HORNIK et al., 1989; HARTMAN et al., 1990; PARK; SANDBERG, 1991, 1993; MULGREW, 1996). Isto significa que as redes MLP e RBF são capazes de implementar as tarefas de modelagem direta e inversa (Figura 1.3), com precisão arbitrária, para sistemas dos mais variados graus de complexidade (IBNKAHLA, 2000).

Graças à propriedade de aproximação universal de função, as redes MLP e RBF são as arquiteturas dominantes no cenário de aplicação de algoritmos neurais em processamento digital de sinais, a ponto de obscurecer a aplicação potencial de outras arquiteturas neurais no mesmo cenário, tal como a rede **Auto-Organizável de Kohonen** (*Kohonen's Self-Organizing Map* - SOM) (KOHONEN, 1990, 1998).

No final da década de 1980 surgiram as primeiras aplicações da rede SOM em modelagem de sistemas estáticos, principalmente em robótica (BARRETO et al., 2003). Nestas aplicações, a rede SOM e arquiteturas equivalentes tiveram desempenho equivalente ao apresentado pelas redes MLP e RBF, porém poucos trabalhos se referem à rede SOM como uma rede capaz de ser utilizada como aproximador de funções. Em geral, a rede SOM é lembrada como um algoritmo de quantização vetorial ou de *clusterização* (HOFMANN; BUHMANN, 1998).

Mais recentemente, foi mostrado que, com pequenas modificações na sua arquitetura original, a rede SOM também pode ser usada para modelar sistemas dinâmicos, tais como aqueles encontrados comumente em problemas de filtragem adaptativa e controle (BARRETO; ARAúJO, 2004; BARRETO et al., 2003; FANCOURT; PRINCIPE, 1998; PRINCIPE et al., 1998). Utilizando a rede SOM nestes problemas, busca-se uma alternativa viável às redes MLP e RBF, sendo esta uma das principais, senão a principal, motivação deste trabalho.

Um outro elemento motivador para esta dissertação está na percepção de que o universo de arquiteturas baseadas ou inspiradas na rede SOM é bastante amplo, o que abre um leque de possibilidades para novas investidas na direção de mostrar como tais arquiteturas podem ser aplicadas em problemas de filtragem adaptativa.

Tendo em mente as motivações supracitadas, a seguir são apresentados os objetivos gerais e específicos desta dissertação.

1.5 Objetivos da Dissertação

De modo resumido, o objetivo maior deste trabalho é oferecer filtros não-lineares baseados na rede SOM como alternativas plausíveis aos filtros não-lineares baseados nas redes MLP e RBF. Para isto, mostra-se como a rede SOM pode ser usada como aproximador

de funções e, assim, ser aplicada em tarefas de filtragem adaptativa.

Os objetivos específicos desta dissertação correspondem aos seguintes itens.

- Aplicar duas arquiteturas conhecidas de aproximadores de funções baseados na rede SOM em tarefas de filtragem.
- Propor novas arquiteturas de filtros adaptativos baseados na rede SOM, e aplicá-las em problemas de filtragem adaptativa.
- Comparar o desempenho das arquiteturas acima mencionadas com filtros adaptativos clássicos lineares (FIR/LMS e FIR/LMS-Newton) e não-lineares (redes MLP e RBF) nas tarefas de identificação e equalização de canais não-lineares.

A organização dos próximos capítulos deste documento e as publicações obtidas até o presente momento são apresentadas nas duas seções seguintes.

1.6 Resumo dos Capítulos Restantes

O restante desta dissertação está organizado segundo a lista de capítulos apresentada abaixo:

Capítulo 2 - Neste capítulo, é feita uma introdução às arquiteturas de redes neurais supervisionadas (MLP e RBF) e não-supervisionadas (SOM) de interesse para esta dissertação. Em particular, será dada ênfase à descrição do funcionamento da rede SOM, em virtude de sua importância para os algoritmos a serem propostos neste trabalho.

Capítulo 3 - Este capítulo apresenta cinco arquiteturas de filtros adaptativos não-lineares baseados na rede SOM. Duas destas arquiteturas já estavam disponíveis na literatura e as outras três representam contribuições desta dissertação.

Capítulo 4 - Neste capítulo, os vários filtros estudados nesta dissertação são avaliados quanto ao desempenho obtido na tarefa de identificação de canais não-lineares. A metodologia de simulação e as características do canal não-linear simulado são também apresentadas neste capítulo.

Capítulo 5 - Este capítulo é similar ao anterior, porém os filtros são avaliados na tarefa de equalização de canais não-lineares.

Capítulo 6 - Este capítulo conclui a dissertação ao fazer um resumo geral dos resultados obtidos, além de apontar direções para trabalhos futuros na mesma linha de pesquisa.

1.7 Produção Científica

Esta dissertação, até o presente momento, teve como principais contribuições os artigos abaixo listados:

1. **L.G.M. SOUZA**, G.A. BARRETO, J.C.M. MOTA. "Using the Self-Organizing Map to Design Efficient RBF Models for Nonlinear Channel Equalization", Aceito para publicação em: V Workshop on Self-Organizing Maps, WSOM'2005.

2. **L.G.M. SOUZA**, G.A. BARRETO, J.C.M. MOTA. "'Novel Algorithms for Nonlinear Channel Equalization Using Neural Vector Quantization", Aceito para publicação em: XXII Simpósio Brasileiro de Telecomunicações, SBrT'05.

3. G.A. BARRETO, J.C.M. MOTA, **L.G.M. SOUZA**, R.A. FROTA. "Nonstationary Time Series Prediction Using Local Models Based on Competitive Neural Networks". *Lecture Notes in Artificial Intelligence*, Alemanha, v. 3029, p. 1146-1155, 2004.

4. G.A. BARRETO, J.C.M. MOTA, **L.G.M. SOUZA**, R.A. FROTA.; "Previsão de séries temporais não-estacionárias usando modelos locais baseados em redes neurais competitivas". *In: VI SIMPÓSIO BRASILEIRO DE AUTOMAÇÃO INTELIGENTE*, 2003, Bauru, SP. *Anais do VI Simpósio Brasileiro de Automação Inteligente (SBAI)*. 2003. p. 941-946.

5. G.A. BARRETO, J.C.M. MOTA, **L.G.M. SOUZA**, R.A. FROTA.; L. AGUAYO (2005), "Condition monitoring of 3G cellular networks using competitive neural models", Aceito para publicação em: IEEE Transactions on Neural Networks.

6. G.A. BARRETO, J.C.M. MOTA, **L.G.M. SOUZA**, R.A. FROTA, L. AGUAYO, J.S. YAMAMOTO, P.E.O. MACEDO. "Competitive neural networks for fault detection and diagnosis of 3G cellular system". *Lecture Notes in Computer Science*, Alemanha, v. 3124, p. 207-213, 2004.

7. G.A. BARRETO, J.C.M. MOTA, **L.G.M. SOUZA**, R.A. FROTA, L. AGUAYO, J.S. YAMAMOTO, P.E.O. MACEDO. "A New Approach to Fault Detection and Diagnosis in Cellular Systems Using Competitive Learning". *In: 8TH BRAZILIAN SYMPOSIUM ON NEURAL NETWORKS*, 2004, São Luís, Maranhão. *Proceedings of SBRN 2004*. 2004. p. 3525-3525.

2 REDES NEURAIS ARTIFICIAIS

Existem muitas definições de redes neurais artificiais (RNAs), porém todas são unânimes em enfatizar algumas características chaves desta tecnologia, tais como paralelismo, não-linearidade, distributividade, conectividade, aprendizado e adaptação, tendo estas duas últimas uma relação muito forte, pois a adaptação da rede neural ao problema proposto é uma conseqüência direta de uma etapa de aprendizado que se realiza de uma forma satisfatória, na qual os elementos constituintes da rede conseguem obter informações necessárias no aprendizado permitindo assim uma boa adaptação do modelo ao problema de interesse. Assim, RNAs são máquinas de aprendizagem não-linear, construídas de muitos elementos processadores simples, chamados usualmente de neurônios artificiais, com capacidade de se adaptar ao meio em função de estímulos (informação) oriundos ao meio em que estão inseridas, e que lançam mão de processamento paralelo e distribuído para processar e codificar informação em conexões entre neurônios. Variações em torno desta definição podem ser encontradas em diversos livros sobre redes neurais artificiais, tais como os escritos por Principe et al. (2000) e Haykin (1994).

Cada neurônio recebe as conexões de outros neurônios e/ou deles próprios. Os sinais que fluem nas conexões são ponderados por parâmetros ajustáveis chamados *pesos sinápticos*, w_{ij}. Grosso modo, um certo neurônio soma todas estas contribuições e produz uma saída que é uma função (estática) não-linear da soma. Saídas dos neurônios se transformam em saídas do sistema, ou são enviadas para si mesmos, ou ainda para os outros neurônios, sendo que neste trabalho, utiliza-se redes do tipo *feedforward*. Outra característica é que essas redes mostram um alto grau de **conectividade**, determinado pelas conexões sinápticas entre neurônios da rede.

RNAs estão entre as mais recentes e bem-sucedidas ferramentas computacionais para tratamento de problemas não-lineares em processamento de sinal (ZAKNICH, 2003; HWANG et al., 1997). Em Engenharia e Computação, redes neurais se aplicam a duas classes genéricas de problemas, que são aparentemente distintas, porém correlatas:

- classificação (ou reconhecimento) de padrões;

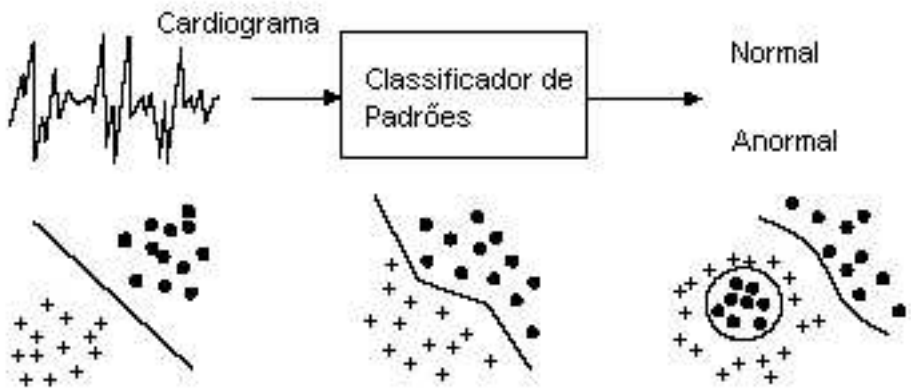


Figura 2.1 Ilustração de um processo de classificação de padrões (adaptada da referência Jain et al. (1996)).

- aproximação de funções.

Na tarefa de classificação de padrões deve-se associar um padrão de entrada (e.g., forma de onda de voz ou símbolo escrito à mão) representado por um vetor de característica para uma das classes predefinidas. Funções discriminantes ou bordas de decisão são construídas de um conjunto de padrões de treinamento com rótulos de classes conhecidas para separar padrões de classes diferente. As bordas de decisão podem ser, linear, linear por partes, ou de alguma forma arbitrária (ver Figura 2.1). Dois aspectos importantes em uma tarefa de classificação de padrão são: (*i*) extração/representação de característica; e (*ii*) construção da borda de decisão. Aplicações bem conhecidas de classificação de padrão são, reconhecimento de caractere, reconhecimento de voz, classificação de forma de onda de EEG e classificação de células de sangue.

Para um dado conjunto rotulado de N padrões de treinamento (pares de vetores entrada-saída) $\{(\mathbf{x}_t, \mathbf{y}_t) : (\mathbf{x}_1, \mathbf{y}_1), (\mathbf{x}_2, \mathbf{y}_2), \ldots, (\mathbf{x}_N, \mathbf{y}_N)\}$, gerado a partir de uma função desconhecida $\mathbf{F}(\mathbf{x}_t)$ (sujeita a ruído), a tarefa de aproximação de função é encontrar uma estimação, chamada $\hat{\mathbf{F}}(\cdot)$, da função desconhecida $\mathbf{F}(\cdot)$. Na literatura estatística, este problema é freqüentemente conhecido como **regressão**. A função estimada $\hat{\mathbf{F}}(\cdot)$ pode ser obtida para ajustar os dados de treinamento com uma precisão arbitrária contornando a complexidade dos mesmos. Um aspecto importante é evitar **sobreajuste** (*overfitting*) do modelo aos dados de treinamento, que são em geral ruidosos. A Figura 2.2 ilustra dois casos típicos em que se verifica o dilema **bias-variância** (GEMAN et al., 1992). Na Figura 2.2(a), tem-se um bom ajuste aos dados de treinamento (simbolizado por $\times$) e a interpolação feita com o dado de teste (simbolizado por o). Para que isto ocorra, a ordem do polinômio (graus de liberdade do modelo) da curva deve ser um valor bem menor do que o número de pontos de treinamento. Já na Figura 2.2(b), tem-se o sobreajuste do

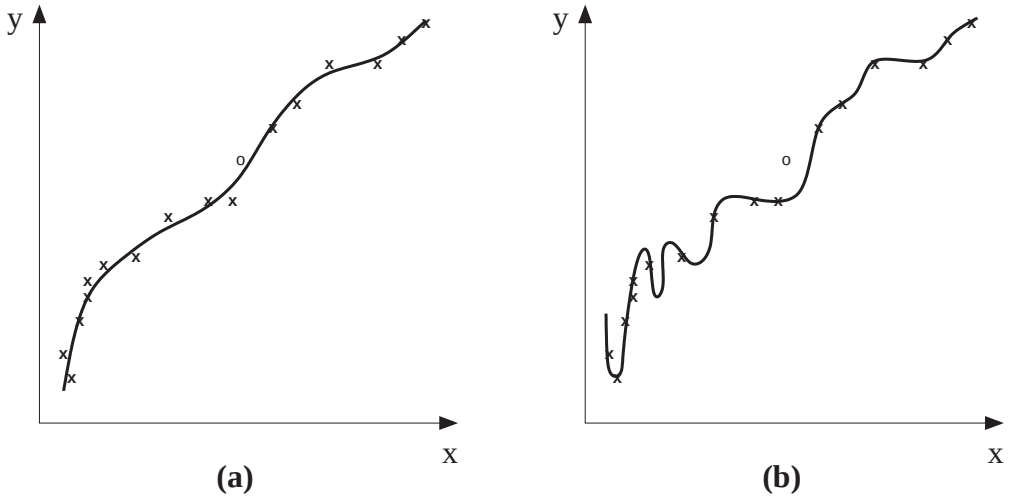


Figura 2.2 (a) Um bom ajuste aos dados ruidosos. (b) Sobreajuste dos mesmos dados: o ajuste é perfeito sobre o "conjunto de treinamento" (×'s), porém é impreciso sobre o "conjunto de teste" representado pelo círculo. Ilustração do dilema bias-variância (adaptada da referência Hertz et al. (1991)).

modelo aos dados de treinamento, efeito este representado por meio de uma curva em que a ordem do seu polinômio é igual ao número de pontos, apresentando portanto, uma insatisfatória interpolação para o dado de teste.

Por serem máquinas de aprendizagem bastante poderosas (i.e. possuírem muitos parâmetros livres) redes neurais artificiais são susceptíveis a sobreajuste quando há um número excessivo de neurônios. Apesar dos dados de treinamento e teste virem de uma mesma distribuição de probabilidade originada pela função que se quer aproximar, ocorre que na etapa de treinamento, a rede neural aprende a representar informação útil assim como desvios estatísticos indesejados (ruído) existente nos dados de entrada. Assim, na etapa de teste a rede não consegue extrapolar o conhecimento adquirido de modo satisfatório para tratar dados não-vistos durante o treinamento.

Classificação de padrão pode também ser visto como um problema de aproximação de função. Vários problemas de modelagem científica e de engenharia requerem aproximação de função. Os problemas de filtragem adaptativa a serem tratados nesta dissertação pertencem à esta classe de problemas.

Quanto ao paradigma de aprendizado, RNAs podem ser divididas em duas categorias: redes com aprendizado supervisionado e redes com aprendizado não-supervisionado. No caso supervisionado, cada entrada apresentada à rede vem acompanhada de uma resposta (saída) desejada e os pesos sinápticos da rede são ajustados para tornar a saída a mais próxima possível daquela desejada. Aprendizado por reforço é um caso especial do aprendizado supervisionado onde a rede é munida somente com análises críticas sobre a exatidão das saídas da rede, não sobre os valores corretos das próprias saídas. No caso

não-supervisionado, a rede neural por si mesma detecta padrões (regularidades dinâmicas e/ou estatísticas) e características no espaço de entrada de forma a construir uma representação no espaço dos pesos sinápticos da rede neural.

As redes neurais recorrentes (RNN) são bastante utilizadas em Engenharia e Computação, sendo aplicadas em problemas de filtragem adaptativa, tais como predição de séries temporais (GENï¿½Y; LIU, 1997; KOSKELA et al., 1998) e equalização de canais (KECHRIOTIS et al., 1994; ONG et al., 1997; PARISI et al., 1997). É importante enfatizar que o uso destas redes neste trabalho não faz parte do objetivo maior do mesmo, deixando a cargo do leitor procurar em outras fontes de pesquisa, textos que se referem ao uso destes modelos nos problemas propostos por esta dissertação. No caso das redes neurais não-recorrentes, elas serão aplicadas aos problemas de filtragem adaptativa vistos no Capítulo 1, e nas seções posteriores, serão abordadas as arquiteturas neurais a serem utilizadas neste trabalho.

Este capítulo tem por objetivo mostrar sucintamente as arquiteturas de redes neurais avaliadas neste trabalho. Apesar dos algoritmos a serem descritos já terem se tornados clássicos, optou-se por descrevê-los brevemente, a fim de facilitar a compreensão dos métodos a serem propostos no próximo capítulo. Neste texto, será dada a denominação de filtros não-lineares às redes supervisionadas, MLP e RBF, como uma forma de facilitar a compreensão do texto, mantendo o raciocínio lógico começado no Capítulo 1 sobre o uso de filtros não-lineares nas tarefas de filtragem adaptativa.

2.1 Redes Neurais Supervisionadas

Nas redes supervisionadas, existe uma exigência quanto à saída que ela deve apresentar (saída desejada). Por isso, o treinamento de tais redes não cessa até que se alcance um nível aceitável de semelhança entre a saída atual da rede e a saída desejada. Elas são as redes mais populares e de uso mais comum graças à celebrada capacidade de aproximar funções com precisão arbitrária (HORNIK et al., 1989; HARTMAN et al., 1990; PARK; SANDBERG, 1991, 1993; MULGREW, 1996).

Muitas das aplicações de redes neurais, particularmente nas áreas de identificação (modelagem) de sistemas não-lineares e filtragem adaptativa, resumem-se ao problema de aproximação de funções desconhecidas, de uma ou mais variáveis (WIDROW; WINTER, 1988; NARENDA; PARTHASARATHY, 1990; CHEN; CHEN, 1993; SADEGH, 1993). Vários autores têm demonstrado empírica e teoricamente que redes neurais multicamadas de alimentação direta (*feedforward*), com uma variedade de funções de ativação não-linear, são aproximadores universais de funções, por exemplo Blum & Li (1991), Geva & Sitte (1992), Hornik et al. (1989), Yang & Tseng (1996), Pinkus (1999), Schilling et al. (2001).

Para utilizar uma rede neural supervisionada é preciso ter em mãos um número representativo finito de N exemplos de treinamento, dados na forma de pares de vetores entrada-saída $(\mathbf{x}(t), \mathbf{d}(t))$

$$\begin{aligned} \mathbf{x}(1), &\quad \mathbf{d}(1) \\ \mathbf{x}(2), &\quad \mathbf{d}(2) \\ \vdots &\quad \vdots \\ \mathbf{x}(N), &\quad \mathbf{d}(N) \quad , \end{aligned} \tag{2.1}$$

para os quais se assume uma relação de causa e efeito dada por uma lei ou função matemática $\mathbf{F}(\cdot)$ desconhecida, ou seja:

$$\mathbf{d}(t) = \mathbf{F}[\mathbf{x}(t)], \tag{2.2}$$

com $t = 1, \ldots, N$. Sem conhecer o mapeamento $\mathbf{F}(\cdot)$, uma maneira de se adquirir conhecimento sobre o mesmo é através dos pares entrada-saída disponíveis. Para isto, pode-se utilizar uma rede neural para gerar um modelo ou representação aproximada de $\mathbf{F}(\cdot)$, denotada por $\hat{\mathbf{F}}(\cdot)$, tal que:

$$\mathbf{y}(t) = \hat{\mathbf{F}}[\mathbf{x}(t)], \tag{2.3}$$

sendo que $\mathbf{y}(t)$ é a saída gerada pela rede. Espera-se que está saída $\mathbf{y}(t)$ seja muito próxima da saída real $\mathbf{d}(t)$.

Cada vetor de entrada é representado como

$$\mathbf{x}(t) = \begin{pmatrix} x_0(t) \\ x_1(t) \\ \vdots \\ x_p(t) \end{pmatrix} = \begin{pmatrix} +1 \\ x(t) \\ \vdots \\ x(t-p+1) \end{pmatrix}, \tag{2.4}$$

com o tempo discreto $t = 1, 2, \cdots, N$ servindo para indicar o instante de apresentação deste vetor à rede, enquanto o vetor de saída é escrito como

$$\mathbf{d}(t) = \begin{pmatrix} d_1(t) \\ \vdots \\ d_M(t) \end{pmatrix}, \tag{2.5}$$

o qual representa o vetor de saídas desejadas associado ao vetor de entrada atual. Ainda, x_j denota um componente qualquer do vetor de entrada $\mathbf{x}$ e d_i denota um componente qualquer do vetor de saídas desejadas $\mathbf{d}(t)$.

A seguir, mostra-se como construir redes neurais multicamadas a partir de um modelo matemático simplificado do neurônio biológico.

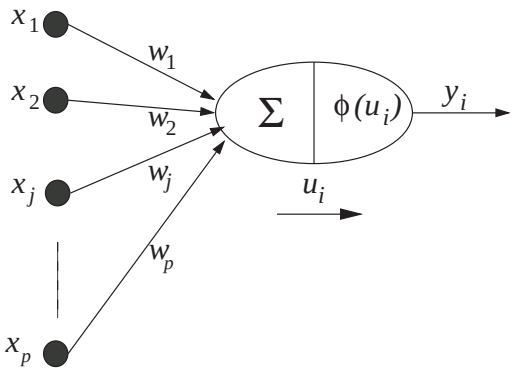


Figura 2.3 Modelo do neurônio de McCulloch-Pitts.

2.2 Filtros Não-Lineares Multicamadas

Conforme brevemente discutido na Seção 1.4, filtros não-lineares, tais como redes neurais, têm sido bastante aplicados em tarefas de filtragem adaptativa. Para uma melhor compreensão destas aplicações das redes, faz-se necessário introduzir conceitos sobre o funcionamento básico de suas arquiteturas e, com isso, analisar as características que as habilitam a ser empregadas como filtros adaptativos não-lineares nos problemas de identificação e equalização de canais de comunicação. Para isto, define-se a seguir o neurônio como bloco construtivo básico de modelos de redes neurais artificiais.

2.2.1 O neurônio não-linear

Como estrutura básica dos filtros não-lineares multicamadas, o modelo do neurônio não-linear foi inicialmente proposto por McCulloch-Pitts[1] (MCCULLOCH; PITTS, 1943), definindo um modelo que tivesse a habilidade de ser usado inicialmente em classificação de padrões.

O elemento processador de McCulloch-Pitts (M-P) é simplesmente uma soma de produtos seguido por uma não-linearidade limiar (ver Figura 2.3). Sua equação de entrada-saída é representada por (HAYKIN, 1994)

$$y_i = \phi(u_i) = \phi\left(\sum_{j=0}^{p} w_{ij}x_j\right) = \phi\left(w_{i0}x_0 + \sum_{j=1}^{p} w_{ij}x_j\right) = \phi\left(\sum_{j=1}^{p} w_{ij}x_j + \theta_i\right), \qquad (2.6)$$

na qual p é o número de entradas, que corresponde ao tamanho do filtro FIR visto no Capítulo 1, x_j são as entradas ao elemento processador, w_{ij} são os pesos do neurônio i, e θ é chamado limiar (*threshold*). A função de ativação $\phi(\cdot)$ é uma função limiar definida

[1]McCulloch e Pitts foram os pesquisadores que contribuíram com o processo de compreensão do funcionamento do neurônio humano (Warren S. McCulloch era um fisiologista e Walter H. Pitts um matemático) e que na década de 1940 propuseram um modelo matemático deste, como um sistema para cálculo lógico.

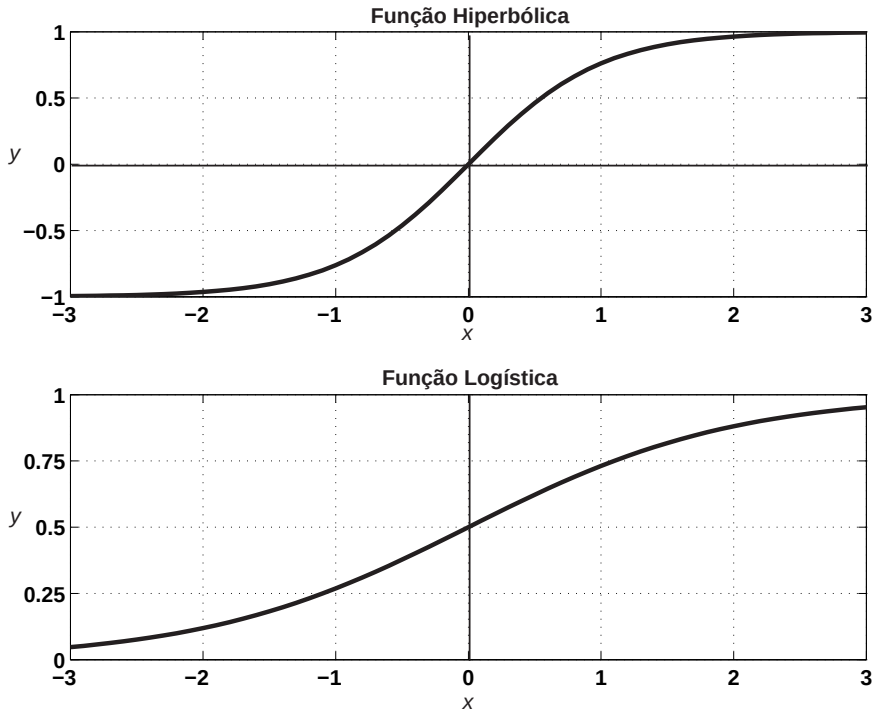


Figura 2.4 Funções sigmoidais do tipo hiperbólica e logística.

por

$$\phi(u_i) = \begin{cases} 1 & ,\text{se} \quad u_i \geq 0 \\ -1 & ,\text{se} \quad u_i < 0 \end{cases}, \tag{2.7}$$

que é comumente chamada de *função sinal*. Pode-se notar que o elemento processador de M-P é composto do elemento linear adaptativo (*Adaline*), que é o elemento processador proposto por Widrow & Hoff (1960) e que implementa uma soma ponderada de entradas, estudado no capítulo anterior, e seguido por uma não-linearidade do tipo sinal.

O neurônio de M-P ainda é o modelo-base da maioria das arquiteturas neurais atualmente em uso. Porém, outras funções de ativação tem sido bastante utilizadas, tais como as chamadas *funções sigmoidais*. Estas funções são chamadas assim porque elas crescem monotonicamente e têm forma da letra S esticada. Existem diversos modelos de funções deste tipo, mas dois dos mais comuns são listados a seguir.

- Função Tangente Hiperbólica:

$$\phi(u_i) = \frac{1 - \exp[-\eta u_i(t)]}{1 + \exp[-\eta u_i(t)]}, \tag{2.8}$$

- Função Logística:

$$\phi(u_i) = \frac{1}{1 + \exp(-\eta u_i)}, \tag{2.9}$$

em que a constante $\eta > 0$ define a inclinação da parte central da curva. Nesta dissertação, adota-se sempre $\eta = 1$ (ver Figura 2.4).

A equação de ajuste dos pesos do neurônio da Figura 2.3 assume também a forma recursiva do gradiente estocástico usado pela regra LMS, ou seja,

$$w_{ij}(t+1) = w_{ij}(t) - \alpha \frac{\partial J}{\partial w_{ij}}, \quad \text{tal que} \quad J = \frac{1}{2N} \sum_{t=1}^{N} \left[d_i(t) - y_i(t) \right]^2 . \tag{2.10}$$

Aplicando a regra da cadeia duas vezes, uma vez para a saída y_i e outra vez para a variável u_i, obtém-se

$$\begin{aligned} \frac{\partial J}{\partial w_{ij}(t)} &= \frac{\partial J}{\partial y_i(t)} \frac{\partial y_i(t)}{\partial u_i(t)} \frac{\partial u_i(t)}{\partial w_{ij}(t)} && (2.11) \\ &= -\left[d_i(t) - y_i(t) \right] \phi^{'} \left[u_i(t) \right] x_j(t) && (2.12) \\ &= -e_i(t) \phi^{'} \left[u_i(t) \right] x_j(t), && (2.13) \end{aligned}$$

sendo $\phi' \left[u_i(t) \right]$ a derivada ordinária da função de ativação $\phi[u_i(t)]$ em relação à ativação $u_i(t)$, e $e_i(t)$ o erro entre a saída desejada para o i-ésimo neurônio e a saída produzida por este neurônio no instante t.

A aplicação da regra do gradiente descendente estocástico resulta na seguinte expressão (compare com a Eq. (1.8)) (PRINCIPE et al., 2000)

$$\begin{aligned} w_{ij}(t+1) &= w_{ij}(t) + \alpha x_j(t) e_i(t) \phi' \left[u_i(t) \right] \\ &= w_{ij}(t) + \alpha x_j(t) \delta_i(t), \end{aligned} \tag{2.14}$$

em que $\delta_i(t)$ é chamado de *gradiente local*, sendo definido como o produto do erro no instante atual e a derivada da função de ativação não-linear. Esta regra é chamada a *Regra LMS Generalizada* e é uma extensão direta da regra LMS aos sistemas com não-linearidade suave.

A regra LMS generalizada é mais estável do que a versão instantânea, que é usada pelo algoritmo LMS, porque a derivada $\phi' \left[u_i(t) \right]$ atenua os efeitos causados por elevados valores do erro $e_i(t)$, em geral causados por elevados valores da ativação $u_i(t)$. Por exemplo, sendo a função de ativação do tipo sigmoidal, a curva gerada para a derivada $\phi' \left[u_i(t) \right]$ tem a forma aproximada de um sino (ver Figura 2.5). Assim, valores de $u_i(t)$ que levem à saturação de $y_i(t) = \phi[u_i(t)]$ não afetarão o ajuste dos pesos correspondentes, visto que $\phi' \left[u_i(t) \right]$ terá valores quase nulos nestes casos.

O modelo de neurônio não-linear descrito acima será usado a seguir para projetar arquiteturas neurais multicamadas em que vários destes elementos processadores estão dispostos em camadas de processamento.

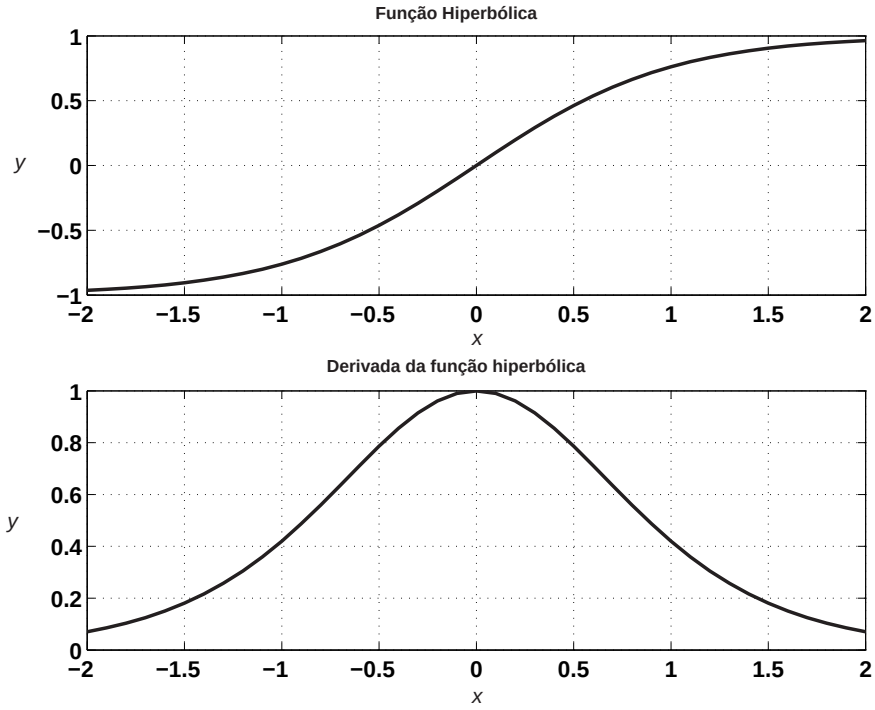


Figura 2.5 Função hiperbólica e sua derivada.

2.2.2 Filtro Não-linear Multicamadas - MLP

Depois de ter apresentado o modelo do neurônio não-linear, explicado a sua estrutura e definido a forma de atualização dos seus pesos, pode-se empregá-lo como elemento básico de uma estrutura em que se tem camadas destes elementos formadores, sendo destinados a solucionar problemas mais complexos de filtragem adaptativa. A primeira destas estruturas multicamadas com neurônios não-lineares é chamada de **Perceptron Multicamadas** (MLP). Tipicamente, uma rede MLP é constituída de uma *camada de entrada* que recebe os sinais, uma ou mais *camadas escondidas* ou *ocultas*, compostas por neurônios com função de ativação não-linear e uma *camada de saída*, também composta por um ou mais neurônios somadores (que podem ser lineares ou não). Os neurônios da camada intermediária são chamados de neurônios escondidos por não terem acesso direto à saída da rede MLP. É na saída que são calculados os sinais de erro necessários para guiar o ajuste dos pesos.

São as camadas escondidas que conferem à rede MLP todo o seu poder computacional, permitindo que ela seja capaz de aproximar, com precisão arbitrária, funções contínuas (rede com 1 camada escondida) e funções descontínuas (rede com pelo menos 2 camadas escondidas) (HAYKIN, 1994). A Figura 2.6 mostra um esboço de uma rede MLP com uma camada de neurônios ocultos e apenas um neurônio na camada escondida.

Para os problemas de filtragem de interesse para esta dissertação utiliza-se a rede

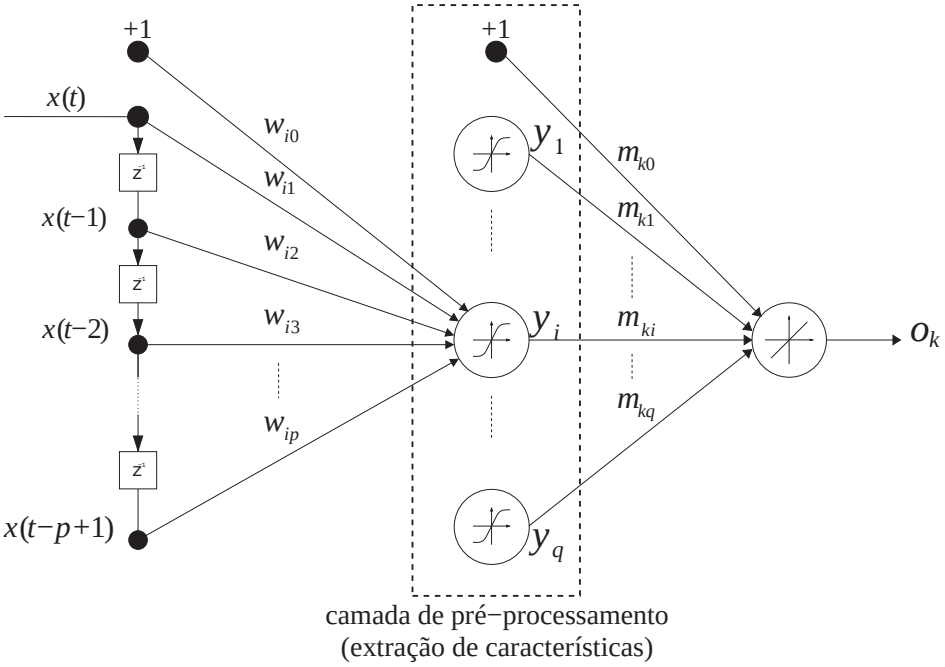


Figura 2.6 Filtro não-linear implementado por uma rede MLP de uma camada escondida e um neurônio linear de saída.

MLP com apenas uma camada escondida e um neurônio de saída com função de ativação linear. Todos os neurônios são treinados segundo o algoritmo de **retropropagação do erro** (*Error Back-propagation*) (HAYKIN, 1994; PRINCIPE et al., 2000), a ser descrito na seqüência do texto.

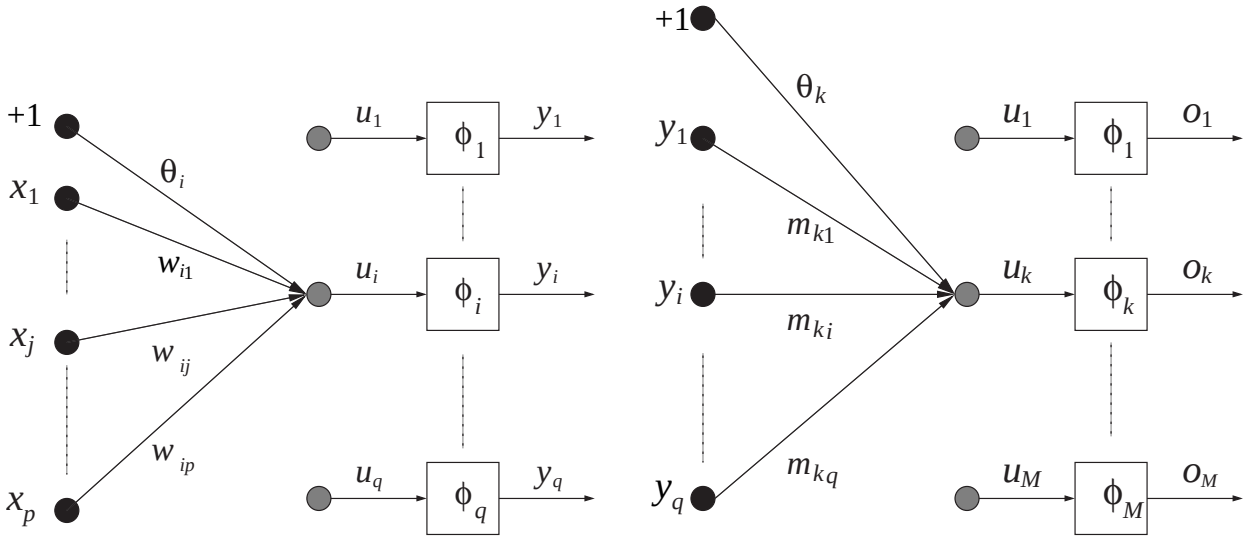
A Figura 2.7 mostra de maneira geral a estrutura de uma arquitetura MLP, com diversos neurônios na camada de saída. Os neurônios da camada escondida estão representados conforme mostrado na Figura 2.7(a), enquanto os neurônios da camada de saída estão representados conforme mostrado na Figura 2.7(b).

O vetor de pesos associado a cada neurônio i da camada escondida é representado como

$$\mathbf{w}_i = \begin{pmatrix} w_{i0} \\ w_{i1} \\ \vdots \\ w_{ip} \end{pmatrix} = \begin{pmatrix} \theta_i \\ w_{i1} \\ \vdots \\ w_{ip} \end{pmatrix}, \tag{2.15}$$

em que θ_i é o limiar associado ao i-ésimo neurônio de entrada; $i = 1, 2, \ldots, q$, sendo q o número de neurônios na camada escondida.

De modo semelhante, o vetor de pesos associado a cada neurônio k da camada de



(a) Neurônios da camada escondida. (b) Neurônios da camada de saída.

Figura 2.7 Estrutura de camadas da rede MLP.

saída é representado como

$$\mathbf{m}_k = \begin{pmatrix} m_{k0} \\ m_{k1} \\ \vdots \\ m_{kq} \end{pmatrix} = \begin{pmatrix} \theta_k \\ m_{k1} \\ \vdots \\ m_{kq} \end{pmatrix}, \tag{2.16}$$

tal que θ_k é o limiar associado ao k-ésimo neurônio de saída; $k = 1, \ldots, M$, sendo M o número de neurônios na camada de saída.

O treinamento da rede MLP se dá em duas fases, detalhadas a seguir.

Sentido Direto: Esta etapa de funcionamento da rede MLP envolve o cálculo das ativações e saídas de todos os neurônios da camada escondida e de todos os neurônios da camada de saída. Assim, o fluxo de informação se dá dos neurônios de entrada para os neurônios de saída, passando obviamente pelos neurônios da camada escondida. Por isso, diz-se que a informação está fluindo no sentido **direto** (*forward*), ou seja:

Entrada → Camada Intermediária → Camada de Saída.

Assim, após a apresentação de um vetor de entrada $\mathbf{x}$, na iteração t, o primeiro passo é calcular as ativações dos neurônios da camada escondida

$$u_i(t) = \sum_{j=0}^{p} w_{ij}(t) x_j(t) = \mathbf{w}_i^T(t)\mathbf{x}(t), \quad i = 1, \ldots, q, \tag{2.17}$$

sendo que $(\cdot)^T$ denota a operação de transposição dos vetores e q indica o número de neurônios da camada escondida. Em seguida, as saídas correspondentes são calculadas como

$$y_i(t) = \phi[u_i(t)] = \phi\left[\sum_{j=0}^{p} w_{ij}(t)x_j(t)\right] = \phi\left[\mathbf{w}_i^T(t)\mathbf{x}(t)\right], \tag{2.18}$$

tal que a função de ativação ϕ assume geralmente uma das seguintes formas:

$$\phi[u_i(t)] = \frac{1}{1+\exp[-u_i(t)]}, \qquad \text{(Logística)}, \tag{2.19}$$

$$\phi[u_i(t)] = \frac{1-\exp[-u_i(t)]}{1+\exp[-u_i(t)]}, \qquad \text{(Tangente Hiperbólica)}. \tag{2.20}$$

O segundo passo consiste em repetir as operações das Equações (2.17) e (2.18) para os neurônios da camada de saída:

$$u_k(t) = \sum_{i=0}^{q} m_{ki}(t)y_i(t), \quad k = 1, \ldots, M, \tag{2.21}$$

em que M é o número de neurônios de saída.

Em seguida, as saídas dos neurônios da última camada são calculadas como

$$o_k(t) = \phi[u_k(t)] = \phi\left[\sum_{i=0}^{q} m_{ki}(t)y_i(t)\right] = \phi\left[\mathbf{m}_k^T(t)\mathbf{y}(t)\right], \tag{2.22}$$

tal que a função de ativação $\phi(\cdot)$ assume geralmente uma das formas definidas nas Equações (2.19) e (2.20), podendo ser também uma função linear.

Sentido Reverso: Esta etapa de funcionamento da rede MLP envolve o cálculo dos gradientes locais (ver Equação (2.11)) e o ajuste dos pesos de todos os neurônios da camada escondida e da camada de saída. O cálculo deste gradiente local serve para definir a parcela de participação do erro causado pelos neurônios da camada escondida em relação à saída da rede, e ajustar os parâmetros para corrigir os erros propagados pelas camadas. Assim, o fluxo de sinais (informação) se dá dos neurônios de saída para os neurônios da camada escondida. Por isso, diz-se que a informação está fluindo no sentido **reverso** (*backward*), ou seja:

Camada de Saída → Camada Escondida.

Assim, após os cálculos das ativações e saídas levados a cabo na fase direta, o primeiro passo da fase reversa consiste em calcular os gradientes locais dos neurônios

da camada de saída. Logo:

$$\delta_k(t) = e_k(t)\phi'[u_k(t)], \quad k = 1, \ldots, M, \tag{2.23}$$

em que $e_k(t)$ é o erro entre a saída desejada $d_k(t)$ para o neurônio k e a saída gerada por ele, $o_k(t)$, ou seja,

$$e_k(t) = d_k(t) - o_k(t), \qquad k = 1, \ldots, M. \tag{2.24}$$

A derivada $\phi'[u_k(t)]$ assume diferentes formas, dependendo da escolha da função de ativação. Assim, temos as seguintes possibilidades:

$$\begin{aligned} \phi'_k[u_k(t)] &= \frac{d\phi_k[u_k(t)]}{du_k(t)} \qquad (2.25)\\ &= o_k(t)[1 - o_k(t)], \quad \text{se } \phi_k[u_k(t)] \text{ é a função logística,} \\ \phi'_k[u_k(t)] &= \frac{d\phi_k[u_k(t)]}{du_k(t)} \qquad (2.26)\\ &= 1 - o_k^2(t), \quad \text{se } \phi_k[u_k(t)] \text{ é a tangente hiperbólica.} \end{aligned}$$

O segundo passo da fase reversa consiste em calcular os gradientes locais dos neurônios da camada escondida

$$\delta_i(t) = \phi'_i[u_i(t)] \sum_{k=1}^{M} m_{ki}\delta_k(t), \quad i = 1, \ldots, q \tag{2.27}$$

tal que a derivada $\phi'[u_i(t)]$ pode ser calculada por uma das seguintes formas

$$\begin{aligned} \phi'_i[u_i(t)] &= y_i(t)[1 - y_i(t)], \quad \text{se } \phi_i[u_i(t)] \text{ é a função logística,} \qquad (2.28)\\ \phi'_i[u_i(t)] &= 1 - y_i^2(t), \quad \text{se } \phi_i[u_i(t)] \text{ é a tangente hiperbólica.} \qquad (2.29) \end{aligned}$$

O terceiro passo da fase reversa corresponde ao processo de atualização ou ajuste dos parâmetros (pesos sinápticos e limiares) da rede MLP com uma camada escondida. Assim, para a camada escondida, temos que a regra de atualização dos pesos, w_{ij}, é dada por:

$$\begin{aligned} w_{ij}(t+1) &= w_{ij}(t) + \Delta w_{ij}(t) \\ &= w_{ij}(t) + \alpha\delta_i(t)x_j(t), \end{aligned} \tag{2.30}$$

em que α é a taxa de aprendizagem que é também constante. E para a camada de saída temos que a regra de atualização dos pesos, m_{ki}, é dada por:

$$\begin{aligned} m_{ki}(t+1) &= m_{ki}(t) + \Delta m_{ki}(t) \\ &= m_{ki}(t) + \alpha\delta_k(t)y_i(t) \end{aligned} \tag{2.31}$$

Assumindo um neurônio de saída com função de ativação linear, pode-se entender a rede MLP como um filtro transversal clássico em que a camada oculta faz o papel de camada de pré-processamento (extração de características) do sinal de entrada (ver Figura 2.6).

2.2.3 Rede MLP em Filtragem Adaptativa

A rede MLP vem sendo aplicada há bastante tempo como identificador e/ou equalizador de canais de comunicação (COWAN, 1991; PENG et al., 1992; AL-MASHOUQ; REED, 1994; YOU; HONG, 1996; ADALI et al., 1997).

Uma das primeiras aplicações das RNAs em equalização de canais em comunicação digital é apresentada por Siu et al. (1990). Eles propuseram uma estrutura de perceptrons multi-camadas (MLP) para equalização de canais e mostraram que o desempenho desta rede é superior ao de um equalizador linear treinado com algoritmo LMS.

O algoritmo de retropropagação foi também usado com o objetivo de equalizar canais de comunicação, através de classificação não-linear, por Gibson et al. (1989). O MLP foi aplicado ao problema de classificar os níveis de informação ± 1 na modulação binária. Este algoritmo foi comparado com modelos de equalizadores lineares aplicados em canais do tipo não-linear e o resultado demonstrou que o classificador linear não resolvia a falta de linearidade nos dados, sendo que o MLP resolvia. Têm-se vários artigos publicados que descrevem o MLP sendo utilizado como um classificador não-linear (ver Gibson et al. (1990), Siu et al. (1990), Gibson & Cowan (1990), Gibson et al. (1991)).

Em Peng et al. (1992), é mostrado a rede MLP sendo usado na equalização de sinais PAM e QAM de alguns tamanhos de constelações (e.g. 4-PAM, 8-PAM, 16-QAM e 64-QAM), dando resultados que superam o equalizador linear baseado no algoritmo LMS. E também, a aplicação da rede MLP em canais de satélite por Chang & Wang (1995).

Uma outra arquitetura neural multicamadas muito usada em filtragem adaptativa é descrita a seguir.

2.2.4 Redes de Funções de Base Radial

Outro modelo que usa neurônios não-lineares na constituição básica de sua arquitetura é a rede RBF (*Radial Basis Function*). A rede RBF é formada por uma camada de neurônios não-lineares cujas funções de ativação são de base radial, do tipo *gaussiana*, em vez de sigmoidais.

A rede RBF é normalmente concebida como sendo composta de uma camada de

entrada, uma só camada intermediária e uma camada de saída (cujos neurônios são geralmente lineares). O funcionamento da rede RBF é descrito a seguir.

Após a apresentação de um vetor de entrada $\mathbf{x}$ na iteração t, calcula-se a ativação do i-ésimo neurônio da camada escondida por meio da seguinte expressão:

$$u_i(t) = \|\mathbf{x}(t) - \mathbf{c}_i\|, \quad i = 1, \ldots, q \tag{2.32}$$

sendo q o número de neurônios desta camada, e o vetor $\mathbf{c}_i$, mantido constante para o neurônio i, define o que se chama de *centro do campo receptivo* associado ao neurônio i. A saída dos neurônios é calculada da seguinte forma:

$$y_i(t) = \phi_i[u_i(t)] = \exp\left[-\frac{u_i^2(t)}{2\gamma_i^2}\right], \quad i = 1, \ldots, q, \tag{2.33}$$

em que a constante $\gamma_i > 0$ define a largura (*spread*) da função de base do neurônio i.

De acordo com a Equação (2.33), o neurônio i fornece resposta máxima, em que $y_i(t) \approx 1$, para vetores de entrada próximos do seu centro $\mathbf{c}_i$. Desta forma, diz-se que cada neurônio da camada escondida tem seu próprio **campo receptivo** (*receptive field*) no espaço de entrada, que é uma região centrada em $\mathbf{c}_i$ com tamanho proporcional a γ_i.

Em seguida, as saídas dos neurônios da última camada são calculadas como

$$o_k(t) = \sum_{i=0}^{q} m_{ki}(t) y_i(t) = \sum_{i=0}^{q} m_{ki}(t) \phi(\|\mathbf{x}(t) - \mathbf{c}_i\|), \tag{2.34}$$

na qual m_{ki}, $k = 1, \ldots, M$, são os pesos associados com a camada de saída e M é o número de neurônios. Na Equação (2.34), assumiu-se que $y_0(t) = +1$ e $m_{k0} = \theta_k$, em que θ_k é o limiar do k-ésimo neurônio de saída.

Foi demonstrado que a rede neural RBF também é um aproximador universal de funções, o que permite que esta arquitetura aproxime funções contínuas com grau de precisão arbitrária, desde que um grande número de neurônios esteja disponível (POGGIO; GIROSI, 1990).

Os parâmetros da rede RBF a serem determinados são a largura (γ_i) e o centro ($\mathbf{c}_i$) de cada função de base da camada escondida e os pesos (m_{ki}) da camada de saída. A seguir, são apresentados alguns procedimentos clássicos para se determinar tais parâmetros.

2.2.4.1 Treinamento da Rede RBF

Diversos algoritmos têm sido propostos para determinar os parâmetros da rede RBF. O procedimento mais usado para este fim é composto de duas etapas: uma etapa de aprendizagem não-supervisionada para determinar os centros e uma etapa de aprendiza-

gem supervisionada para calcular os pesos de saída.

***Determinar os centros através de algoritmos de clusterização*:** Os centros são determinados de acordo com o seguinte algoritmo:

1. Gerar q vetores aleatórios e defini-los como centros iniciais da rede, ou seja, $\mathbf{c}_i(0)$.
2. Apresentar um sinal $\mathbf{x}(t) = [x(t), x(t-1), \ldots, x(t-p+1)]^T$ para a rede.
3. Calcular as distâncias euclidianas entre o vetor de entrada atual e os centros da rede:
$$D_i(t) = \|\mathbf{x}(t) - \mathbf{c}_i(t)\|;\ \forall\ i = 1, \ldots, q.$$
4. Determinar o neurônio (*vencedor*) cujo centro está mais próximo ao sinal de entrada atual:
$$i^*(\mathbf{x}(t)) = \arg\min_{\forall i} D_i(t).$$
5. Atualizar o centro do neurônio vencedor da seguinte forma:
$$\mathbf{c}_{i^*}(t+1) = \mathbf{c}_{i^*}(t) + \alpha[\mathbf{x}(t) - \mathbf{c}_{i^*}(t)],$$
em que $0 < \alpha < 1$.
6. Retorne ao Passo 2.

***Aprendizagem supervisionada para os pesos da camada de saída*:** Uma vez determinados os centros $\mathbf{c}_i$, como a camada de saída possui uma função de ativação linear, pode-se utilizar o algoritmo LMS para atualizar os seus respectivos pesos, conforme descrito a seguir.

1. Apresentar um par de sinais de entrada-saída desejada $[\mathbf{x}(t), \mathbf{d}(t)]$.
2. Calcular a função de ativação $\phi(\cdot)$ para a entrada $\mathbf{x}(t)$ corrente:
$$z_i(t) = y_i(t), \text{ver Equação (2.33).}$$
3. Calcular o erro entre a saída da rede e a saída desejada do k-ésimo neurônio:
$$e_k(t) = d_k(t) - o_k(t).$$
4. Atualizar os pesos da camada de saída de acordo com:
$$m_{ki}(t+1) = m_{ki}(t) + \alpha e_k(t) z_i(t),$$
em que $0 < \alpha < 1$.

5. Retorne ao passo 1.

Em Principe et al. (2000) e Haykin (1994) diversos métodos para determinação de γ_i são apresentados. Nesta dissertação adota-se o seguinte procedimento. Empiricamente, define-se a largura da gaussiana como sendo uma fração da distância média entre os P centros mais próximos (vizinhos), ou seja

$$\gamma_i^2 = \frac{1}{P} \sum_{v=1}^{P} \|\mathbf{c}_i - \mathbf{c}_v\|^2, \tag{2.35}$$

na qual os P centros mais próximos do i-ésimo centro são escolhidos. O valor adotado neste trabalho para o parâmetro P é 2.

2.2.5 Rede RBF em Filtragem Adaptativa

Quando aplicada em filtragem adaptativa, a rede RBF em geral possui apenas um neurônio de saída. Deste modo, assim como a rede MLP, a rede RBF pode ser entendida como um filtro FIR linear com uma camada de pré-processamento não-linear. A rede RBF é também utilizada como classificador em problemas de identificação e equalização de canais de comunicações, sendo que alguns artigos retratam este emprego da rede RBF (CHEN et al., 1993; KUMAR et al., 2000). Tem-se também aplicação conjunta das redes RBF e MLP (LU; EVANS, 1999).

Em DaSilva (2001), o autor utiliza as redes MLP e RBF para obter equalizadores não-lineares e compará-los com o equalizador linear transversal e com os equalizadores ótimos segundo os critérios de Bayes e da máxima verossimilhança. Nesta comparação, foram utilizados um alfabeto binário e um quaternário transmitidos em modelos de canais cuja resposta ao pulso unitário é finita. O autor fez um estudo comparativo dos algoritmos de treinamento das redes e obteve um algoritmo do tipo acelerador para o treinamento de redes MLP. Com o intuito de se obter uma estrutura não-linear menos complexa e mais flexível, o autor propôs ainda um equalizador híbrido constituído de uma combinação do equalizador linear e da rede RNN que faz uso de realimentação de decisões. Resultados de simulações indicam que o seu uso pode ser vantajoso tanto para canais não-lineares como lineares.

2.3 Redes Neurais Competitivas

Esta seção faz uma introdução à teoria da **Rede Auto-Organizável de Kohonen** (*Kohonen's Self-Organizing Map*-SOM), proposta por Kohonen (1982). Esta rede, que pertence à classe das redes neurais (não-supervisionadas) competitivas, é uma das principais arquiteturas neurais, encontrando aplicações em um número variado de áreas da Engenharia e da Ciência, conforme pode ser averiguado no levantamento bibliográfico feito em Oja et al. (2003). Contudo, a rede SOM tem sido de aplicação restrita em filtragem adaptativa não-linear, devido em parte ao domínio das arquiteturas neurais de aproximadores universais de funções, MLP e RBF. Conforme mencionado no Capítulo 1, esta dissertação tem por objetivo maior alargar o campo de aplicação da rede SOM, em filtragem adaptativa, para além da sua aplicação tradicional como quantizador vetorial.

Antes porém, faz-se necessário apresentar os principais conceitos envolvidos no projeto e treinamento de uma rede SOM. Devido à importância da rede SOM para esta dissertação, será dedicada a esta rede mais espaço para descrição do seu funcionamento do que foi para as redes MLP/RBF.

2.3.1 Rede SOM

O desenvolvimento de mapeamentos auto-organizáveis como um modelo neural é motivado por uma característica peculiar do cérebro humano. Muitas porções do córtex cerebral estão organizadas de uma forma que diferentes entradas sensoriais são representadas por *mapas computacionais topologicamente organizados*. Assim, mapas computacionais constituem um bloco construtivo básico na infraestrutura de processamento de informação do sistema nervoso. Um mapa computacional é definido por um *arranjo* de neurônios representando diferentes elementos processadores ou filtros, que atuam em paralelo sobre sinais portadores de informação.

A rede SOM foi proposta com o intuito de modelar algumas características essenciais dos mapas computacionais existentes no cérebro sem ter, contudo, a ambição de ser um modelo biologicamente plausível. Seu objetivo principal é transformar um sinal de entrada de dimensão qualquer em um arranjo discreto de unidades de processamento. Esta transformação deve ser realizada de forma adaptativa e mantendo relações de similaridade entre os dois espaços.

Seja $\mathcal{X}$ um espaço contínuo de dados de entrada tal que sua topologia é definida por certas relações métricas entre vetores $\mathbf{x} \in \mathcal{X}$. Do espaço $\mathcal{X}$ conhece-se apenas um conjunto finito de vetores (amostras) $\mathbf{x} \in \mathcal{X}$ organizados segundo uma densidade de probabilidade $p(\mathbf{x})$. A topologia do espaço de saída $\mathcal{A}$ é definida pelo arranjo geométrico

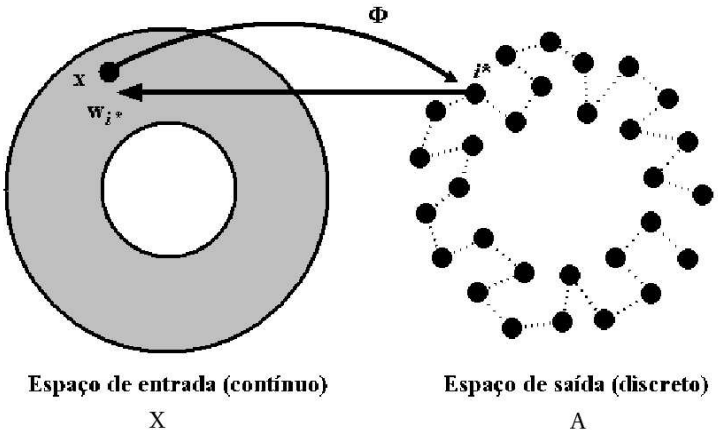


Figura 2.8 Esboço do mapeamento de características Φ e seus elementos constituintes para uma grade do tipo unidimensional.

de um conjunto de neurônios $i \in \mathcal{A}$. Seja Φ uma transformação não-linear, chamada *mapeamento de características* (*feature map*, em inglês), que leva do espaço de entrada $\mathcal{X}$ ao espaço de saída $\mathcal{A}$ (ver Figura 2.8). Matematicamente, tem-se

$$\Phi : \mathcal{X} \rightarrow \mathcal{A}. \tag{2.36}$$

Seja p a dimensão do vetor de espaço (contínuo) de entrada $\mathcal{X}$. Um vetor de entrada $\mathbf{x}(t) \in \mathcal{X} \subset \mathbb{R}^p$, selecionado de forma aleatória, é representado por

$$\mathbf{x}(t) = \begin{pmatrix} x_1(t) \\ x_2(t) \\ \vdots \\ x_p(t) \end{pmatrix} = \begin{pmatrix} x(t) \\ x(t-1) \\ \vdots \\ x(t-p+1) \end{pmatrix}. \tag{2.37}$$

O vetor de pesos associado a cada neurônio da rede tem a mesma dimensão do vetor de entrada. A representação do vetor de pesos de um neurônio i é a seguinte

$$\mathbf{w}_i = \begin{pmatrix} w_{i1} \\ w_{i2} \\ \vdots \\ w_{ip} \end{pmatrix}, \quad i = 1, 2, \ldots, q, \tag{2.38}$$

na qual q é o número total de neurônios da rede.

Dado um vetor de entrada $\mathbf{x} \in \mathcal{X}$, o algoritmo SOM primeiro identifica um determinado neurônio $i^*(\mathbf{x})$ no espaço de saída $\mathcal{A}$ em concordância com o mapeamento característico Φ. O vetor de pesos $\mathbf{w}_{i^*}$ associado ao neurônio i^* pode então ser visto como um

ponteiro para o espaço de entrada $\mathcal{X}$.

O algoritmo responsável pela formação do mapa auto-organizável tem como passo inicial a atribuição de valores iniciais aos pesos da rede. Assim procedendo, nenhum ordenamento inicial é imposto aos neurônios. Uma vez que os valores iniciais para os parâmetros da rede tenham sido devidamente escolhidos, três processos estão envolvidos na formação do mapa auto-organizável:

- *Competição.* Para cada vetor de entrada, os neurônios na rede calculam seus respectivos valores para uma certa função discriminante. Esta função provê a base para uma competição entre os neurônios visando escolher aquele que melhor represente o vetor de entrada atual. O neurônio com menor valor para a função discriminante é declarado o *vencedor* da competição.

- *Cooperação.* O neurônio vencedor e seus vizinhos no mapa discreto interagem através de uma *função vizinhança.* Essa interação é tanto mais positiva quanto mais próxima um determinado neurônio estiver do neurônio vencedor.

- *Adaptação.* Os dois processos (competição e cooperação) descritos nos itens anteriores atuam conjuntamente durante o ajuste dos pesos sinápticos da rede. O neurônio vencedor fica com a maior parcela do ajuste, mas seus vizinhos no mapa também têm seus pesos ajustados de acordo com a sua distância ao neurônio vencedor. Quanto mais próximo do vencedor, maior o ajuste.

Uma análise detalhada do processo de competição, cooperação e adaptação é feita nas próximas subseções. E continuando, são apresentadas outras duas subseções onde se comenta sobre o critério de convergência da rede SOM, e também, sobre a preservação de topologia feita por esta rede.

2.3.1.1 Competição: O Papel da Distância Euclidiana

Para encontrar o neurônio vencedor, o vetor de pesos de todos os neurônios da rede é comparado com o vetor de entrada. Essa comparação é, em geral, uma medida da distância entre cada um dos vetores de pesos e o vetor de entrada. O neurônio cujo vetor de pesos está mais próximo do vetor de entrada é considerado o *vencedor.* Matematicamente, a competição pode ser implementada em termos de distância euclidiana da seguinte forma

$$i^*(\mathbf{x}(t)) = \arg\min_{i \in \mathcal{A}} \|\mathbf{x}(t) - \mathbf{w}_i(t)\|, \tag{2.39}$$

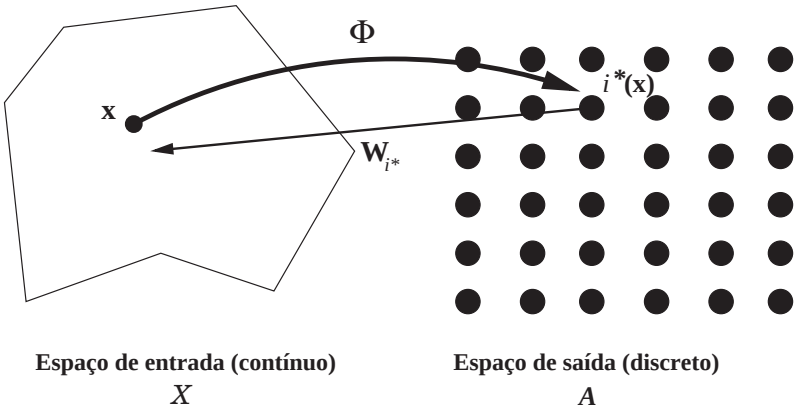


Figura 2.9 Esboço do mapeamento de características Φ e seus elementos constituintes para uma grade do tipo bidimensional.

em que $i^*(\mathbf{x}(t))$ é o índice que representa o neurônio vencedor para o padrão de entrada $\mathbf{x}(t)$. A norma euclidiana $\|\cdot\|$ é definida como

$$\|\mathbf{x}(t) - \mathbf{w}_i(t)\| = \sqrt{[\mathbf{x}(t) - \mathbf{w}_i(t)]^T[\mathbf{x}(t) - \mathbf{w}_i(t)]} = \sqrt{\sum_{j=1}^{p}[x_j(t) - w_{ij}(t)]^2}, \qquad (2.40)$$

na qual $(\cdot)^T$ denota o vetor transposto. A Equação (2.39) permite a seguinte observação:

> *O espaço contínuo a que pertencem os vetores de entrada é mapeado em um espaço de saída discreto por meio de um processo de competição entre os neurônios.*

2.3.1.2 Cooperação: O Papel da Função de Vizinhança

O neurônio vencedor $i^*(t)$ determina o centro de um grupo espacialmente localizado de neurônios no mapa; a vizinhança de i^*. O neurônio vencedor e sua vizinhança interagem lateralmente de forma cooperativa. Existe evidência neurobiológica para esta *interação lateral* entre os neurônios excitados (KOHONEN, 1997). Em particular, um neurônio que está disparando tende a excitar neurônios em sua vizinhança imediata *mais* do que aqueles mais distintos. A intensidade da interação lateral entre o vencedor i^* e um neurônio i qualquer é, em geral, descrita matematicamente na forma de uma função vizinhança $h(i^*, i; t)$. Esta função define o que se chama de vizinhança *topológica* centrada no neurônio vencedor $i^*(\mathbf{x}(t))$.

Considerando o estudo do arranjo discreto assumido pelos neurônios na rede SOM, têm-se dois modelos bastante utilizados (KOHONEN, 1997): (i) a grade unidimensional e, (ii) a grade bidimensional. Analisando o caso de que os neurônios estejam dispostos em uma grade unidimensional, tem-se que $\mathbf{r}_i(t) \in \mathbb{R}$, ou seja, a posição de um neurônio

i qualquer coincide com seu próprio índice, ou seja, $\mathbf{r}_i(t) = i$. Neste caso, cada neurônio possui apenas vizinhos à direita e à esquerda (ver Figura 2.8). Contudo, se os neurônios da rede SOM estão dispostos em uma grade bidimensional, tem-se que $\mathbf{r}_i(t) \in \mathbb{R}^2$, ou seja, a posição de um neurônio i na grade é dada pelas coordenadas (x_i, y_i) em relação a uma origem pré-fixada. Neste caso, um neurônio pode ter vizinhos à esquerda, à direita, acima, abaixo e diagonalmente (ver Figura 2.9).

Seja $\mathbf{r}_i$ a localização (coordenadas) do neurônio i em um arranjo uni-dimensional, por exemplo. Então, assume-se que $h(i^*, i; t)$ é uma função unimodal da distância lateral $\|\mathbf{r}_{i^*} - \mathbf{r}_i\|$ entre o neurônio vencedor i^* e seu vizinho i no mapa. A função vizinhança deve ainda satisfazer os seguintes requisitos:

- A função vizinhança $h(i^*, i; t)$ alcança seu valor máximo para o neurônio vencedor i^* para o qual a distância lateral $\|\mathbf{r}_{i^*} - \mathbf{r}_i\|$ é nula.
- A função vizinhança $h(i^*, i; t)$ é simétrica em relação ao neurônio vencedor.
- A amplitude de $h(i^*, i; t)$ decai monotonicamente com o aumento da distância lateral $\|\mathbf{r}_{i^*} - \mathbf{r}_i\|$, ou seja:

$$\text{Se} \quad \|\mathbf{r}_{i^*} - \mathbf{r}_i\| \longrightarrow \infty \quad \Longrightarrow \quad h(i^*, i; t) \longrightarrow 0. \tag{2.41}$$

  A Equação (2.41) é uma condição necessária para convergência.

Uma escolha comum para $h(i^*, i; t)$ que satisfaz os requisitos anteriores é a função gaussiana, ou seja,

$$h(i^*, i; t) = \exp\left(-\frac{\|\mathbf{r}_{i^*} - \mathbf{r}_i\|^2}{2\sigma^2(t)}\right), \tag{2.42}$$

sendo que o parâmetro $\sigma(t)$ define a "largura efetiva" da vizinhança topológica, i.e., ele define como os demais neurônios participam no processo de aprendizagem juntamente com o neurônio vencedor no instante atual. É importante ressaltar que o parâmetro σ usado pela rede SOM difere do seu equivalente na rede RBF pois este se refere ao raio de ação do neurônio definido pela largura da função de ativação gaussiana do mesmo.

É importante ressaltar que a forma original do algoritmo SOM descrita em Kohonen (1982) utiliza uma função vizinhança de amplitude constante (retangular):

$$h(i^*, i; t) = \begin{cases} 1, \text{ se } i \in N_{i^*}(t) \\ 0, \text{ caso contrário,} \end{cases} \tag{2.43}$$

em que $N_{i^*}(t)$, chamado de *conjunto vizinhança*, contém os neurônios vizinhos de $i^*(t)$. A função de vizinhança retangular é adotada por este trabalho pois ela apresenta um custo computacional consideravelmente menor do que a gaussiana (KOHONEN, 1998).

Tanto para a função vizinhança gaussiana quanto para a retangular, a largura da vizinhança topológica decresce monotonicamente com o passar do processo de aprendizagem. A diminuição no número de vizinhos é de fundamental importância para o ordenamento e convergência da rede. Maiores detalhes sobre como proceder para diminuir a largura da vizinhança com o tempo para as funções gaussiana e retangular serão discutidos posteriormente.

2.3.1.3 Adaptação: Ajuste dos Pesos

O ajuste dos pesos sinápticos é o último passo na formação de um mapa auto-organizável. A questão que se coloca aqui é como realizar a alteração dos pesos. Como a rede SOM é não-supervisionada, o vetor de pesos de um dado neurônio i deve ser modificado como função do estímulo de entrada apenas. Conforme dito anteriormente, os processos de competição e cooperação atuam conjuntamente durante a adaptação sináptica com o objetivo de extrair algum tipo de regularidade presente no espaço de entrada $\mathcal{X}$. A regra de aprendizagem para o algoritmo SOM baseia-se em suposições feitas por Hebb (1949), em uma tentativa de relacionar a alteração estrutural de sinapses reais com a memória e, conseqüentemente, com aprendizagem ou experiência. Assim, uma abstração matemática destas suposições foi proposta por Kohonen como uma regra recursiva para ajuste dos pesos (KOHONEN, 1998):

$$\mathbf{w}_i(t+1) = \mathbf{w}_i(t) + \alpha h(i^*, i; t)[\mathbf{x}(t) - \mathbf{w}_i(t)], \tag{2.44}$$

em que o parâmetro $0 < \alpha < 1$, chamado de *taxa de aprendizagem*, controla a intensidade com que os pesos sinápticos são modificados. Assim como no caso da largura da vizinhança topológica, a taxa de aprendizagem pode diminuir com o transcorrer do treinamento de modo a garantir convergência e, principalmente, estabilidade do mapa[2], sendo que neste trabalho é adotado uma taxa de aprendizagem constante.

2.3.1.4 Ordenamento e Convergência

A partir de uma configuração inicial em que os pesos guardam valores atribuídos aleatoriamente, o algoritmo SOM modifica maximamente os valores dos pesos em direção a uma representação que reflita a estrutura (topologia) do espaço de entrada. Para que uma configuração "organizada" e estável do mapa seja atingida, faz-se necessária uma seleção criteriosa dos parâmetros α e $\sigma(t)$. Quando uma configuração organizada final é alcançada, diz-se que o algoritmo convergiu ou atingiu um estado final.

[2]Neste contexto, estabilidade se refere à manutenção de memória previamente aprendida quando novos dados são apresentados à rede SOM.

O ponto de importância capital para o processo de convergência da rede é a diminuição da largura da vizinhança topológica. Para o caso de uma função vizinhança gaussiana, esta largura é refletida no valor do parâmetro $\sigma(t)$. A largura deve ser inicialmente alta para promover um rápido ordenamento dos pesos e decrescer de modo a garantir convergência dos mesmos. Uma escolha comum para a dependência temporal de $\sigma(t)$ tem a forma apresentada a seguir:

$$\sigma(t) = \sigma_0 \cdot \left(\frac{\sigma_f}{\sigma_0}\right)^{\frac{t}{\tau_2}}, \quad t = 0, 1, ..., \tau_2, \tag{2.45}$$

em que τ_2 é uma outra constante de tempo desta vez associada ao decaimento do parâmetro σ. Uma maneira útil de entender a influência da função vizinhança $h(i^*, i; t)$ com o passar do tempo é a seguinte. O propósito de uma largura inicial grande para $h(i^*, i; t)$ é *correlacionar* as direções de ajuste dos pesos de um grande número de neurônios da rede. À medida que a largura de $h(i^*, i; t)$ decresce, também decresce o número de neurônios cujas direções de ajuste estão correlacionadas.

Como foi explicado acima, no processo de convergência para o caso da função vizinhança tida como gaussiana, repete-se este mesmo procedimento para o caso retangular e segue que, para esta dissertação, a preferência no uso da vizinhança do tipo retangular sobre a do tipo gaussiana devido o menor custo computacional da primeira. Então, no decorrer da atualização dos pesos sinápticos o valor de $h(i^*, i; t)$ mantém-se constante e igual a um, fazendo uma atualização igualitária entre os neurônios vizinhos ao vencedor, i, e o próprio vencedor, $i^*(t)$, com a diminuição da largura da vizinhança topológica acontecendo gradativamente.

2.3.1.5 Preservação de Topologia

Pode-se expressar a propriedade de preservação de topologia da rede SOM da seguinte forma (HERTZ et al., 1991): sejam $\mathbf{x}_1$ e $\mathbf{x}_2$ dois vetores no espaço de entrada $\mathcal{X}$, $\mathbf{r}_{i_1^*}$ e $\mathbf{r}_{i_2^*}$ as coordenadas dos neurônios vencedores para $\mathbf{x}_1$ e $\mathbf{x}_2$, respectivamente. Diz-se que a rede SOM, corretamente treinada, preserva a topologia do espaço de entrada se a seguinte relação for observada

$$\|\mathbf{x}_1 - \mathbf{x}_2\| \longrightarrow 0 \quad \Longrightarrow \quad \|\mathbf{r}_{i_1^*} - \mathbf{r}_{i_2^*}\| \longrightarrow 0, \tag{2.46}$$

ou seja, se quaisquer dois vetores estão fisicamente próximos no espaço de entrada, então eles terão neurônios vencedores espacialmente próximos na rede. Dá-se a essa característica, o nome de *Propriedade de Preservação de Topologia.* Com base nesta propriedade, pode-se fazer a seguinte afirmação sobre o algoritmo SOM:

O espaço contínuo $\mathcal{X}$ é mapeado em um espaço discreto de saída $\mathcal{A}$ por

um processo de competição-cooperação entre as unidades da rede, de forma tal que a sua topologia é preservada.

Devido à propriedade de preservação de topologia, a rede SOM é capaz de construir uma **aproximação do espaço de entrada**, ou seja, ela constrói uma aproximação discreta do espaço de entrada, na qual cada neurônio da rede representa uma determinada região do espaço de entrada que define sua **região de atração** ou **campo receptivo**. Esta região é conhecida também como **célula de Voronoi**. Assim, uma das principais aplicações da rede SOM é a categorização de dados não-rotulados em agrupamentos (clusters) e sua posterior utilização na classificação de vetores de características que não estavam presentes durante o treinamento.

Com o uso das propriedades da rede SOM, tais como competição e cooperação, esta rede neural é capaz de implementar uma projeção Φ (Eq. (2.36)) que preserve relações de proximidade espacial entre os dados de entrada, ou seja, o mapeamento preserva a topologia do espaço de entrada no espaço de saída (HAYKIN, 1994), conforme ilustrado na Figura 2.8, na qual $dim(\mathcal{X}) = p = 2$ e $dim(\mathcal{A}) = 1$, e os pontos pretos correspondem às coordenadas dos vetores de pesos do i-ésimo neurônio. Neurônios que são vizinhos na grade unidimensional são conectados por linhas tracejadas.

2.3.2 Rede SOM em Filtragem Adaptativa

A aplicação clássica da rede SOM em filtragem adaptativa é na forma de quantizador vetorial para compressão de dados ou, equivalentemente, como algoritmo de *clusterização*, tal como relatado nos trabalhos de Yair et al. (1992), Hofmann & Buhmann (1998) e DeBodt et al. (2004). Os resultados obtidos pela rede SOM demonstram bem a sua capacidade de discretizar o espaço dos dados, captando regularidades estatísticas presentes nos dados de entrada e codificando-as nos vetores-protótipos da rede. Em Chen et al. (1993), os autores propõem o uso de algoritmo de *clusterização* no problema de equalização de canais de comunicações digital que utiliza modulação do tipo binária $\{\pm 1\}$, indicando viabilidade da aplicação desde tipo de algoritmo, na qual se baseia o funcionamento da rede SOM, neste problema em comunicações digitais.

Na literatura são encontradas alguns poucos artigos reportando aplicações da rede SOM em filtragem adaptativa. Neste campo, a rede SOM foi aplicada com sucesso em equalização de canais de comunicação, sendo utilizada no processo como um classificador não-linear em parceria com métodos clássicos de equalização linear (KOHONEN et al., 1990; RAIVIO et al., 1991, 1997, 1998; RAIVIO, 1999). É importante perceber que cronologicamente o tratamento do problema de equalização de canais como um problema de classificação foi proposta inicialmente por Gibson et al. (1989), Siu et al. (1990), Cowan

(1991), Gibson et al. (1991), e pode-se notar que a rede SOM foi utilizada neste tipo de aplicação logo em seguida. Têm-se aplicação conjunta das redes RBF e SOM para equalização de canais de comunicação via satélite (BOUCHIRED et al., 1998) tendo resultados que superam o algoritmo LMS, sendo demonstrados no mesmo trabalho.

2.4 Conclusão

Neste capítulo foram apresentadas arquiteturas clássicas das redes neurais supervisionadas, tais como MLP e RBF, e a rede competitiva de Kohonen. Foi mostrado que redes MLP e RBF podem ser entendidas como um filtro FIR linear transversal clássico, quando utilizadas somente com um neurônio linear na camada de saída, em que a camada escondida desempenha o papel de uma camada de pré-processamento ou extração de características. Esta interpretação será bastante útil nas análises que serão feitas entre os algoritmos neurais e os filtros lineares nos próximos capítulos.

É importante enfatizar que não foram encontrados registros bibliográficos reportando o uso da rede SOM em filtragem adaptativa como aproximador de funções, sendo que a aplicação mais encontrada desta rede neural foi como um classificador não-linear. Assim, busca-se nesta dissertação ampliar a área de aplicação da rede SOM em filtragem adaptativa modificando seu algoritmo de forma a habilitá-la como aproximador de funções. Isto será feito no próximo capítulo.

3 ALGORITMOS DE FILTRAGEM ADAPTATIVA BASEADOS NA REDE DE KOHONEN

De maneira geral, o principal objetivo deste capítulo é mostrar que a rede SOM pode ser usada como aproximador de funções e assim ter sua aplicabilidade no campo de filtragem adaptativa estendida para além do uso como algoritmo de quantização vetorial ou *clusterização*.

Em outras palavras, neste capítulo serão apresentados algoritmos de filtragem adaptativa baseados na rede SOM aplicados em tarefas de identificação e equalização de canais não-lineares. Pode-se adiantar aqui que isto é possível graças a estratégias de aprendizado *auto-supervisionado*, em que mapeamentos associativos são criados entre o espaço dos vetores de entrada e o respectivo espaço das saídas desejadas. Mais detalhes sobre este tópico serão dados mais adiante neste capítulo.

Os objetivos específicos deste capítulo estão abaixo listados:

- Mostrar como algoritmos lineares clássicos de filtragem adaptativa, tal como o filtro FIR/LMS, podem ser usados em conjunção com a rede SOM. Esta é a estratégia associativa por trás do algoritmo conhecido como **Mapeamento Linear Local** (*Local Linear Mapping* - LLM) (WALTER et al., 1990).

- Descrever como a rede SOM pode ser simultaneamente aplicada aos espaços de entrada e de saída, a fim de promover a quantização vetorial destes espaços, ao mesmo tempo que associa os protótipos (centróides) do espaço de entrada com os protótipos do espaço de saída. Esta é a estratégia associativa por trás do algoritmo conhecido como **Memória Associativa Temporal por Quantização Vetorial** (*Vector-Quantized Temporal Associative Memory* - VQTAM) (BARRETO; ARAúJO, 2004).

- Usar a estratégia VQTAM para propor novos algoritmos de filtragem adaptativa, a

saber:

  - **RBF Global** (*Global RBF* - GRBF) (SOUZA et al., 2005).
  - **RBF Local sobre K-vencedores** (*Local RBF over K-winners* - KRBF) (SOUZA et al., 2005).
  - **Mapeamento Linear Local sobre K-vencedores** (*Local Linear Mapping over K-winners* - KSOM) (BARRETO et al., 2003).

- Aplicar todas as arquiteturas descritas nos itens anteriores em problemas de filtragem adaptativa não-linear.

Iniciar-se-á na próxima seção a descrição dos algoritmos clássicos em aproximação de funções usando a rede SOM e os modelos propostos por este trabalho, que serão apresentados posteriormente.

3.1 Mapeamento Linear Local

O primeiro algoritmo a ser descrito é chamado **Mapeamento Linear Local** (*Local Linear Mapping*-LLM) (WALTER et al., 1990). A idéia subjacente à proposição deste algoritmo consiste em mostrar como algoritmos clássicos de filtragem adaptativa, tal como o filtro linear FIR/LMS, podem ser usados em conjunção com a rede SOM.

Assim, o filtro LLM nada mais é que uma rede SOM na qual cada neurônio tem um filtro FIR associado. Neste caso, a rede SOM propriamente dita é usada para quantizar o espaço de entrada em um número reduzido de vetores-protótipos, enquanto os coeficientes de cada filtro são calculados usando os vetores de estado (entrada) para os quais o neurônio associado é o vencedor.

De maneira mais formal, seja p a dimensão do espaço (contínuo) de entrada $\mathcal{X}$ (Figura 2.9) sobre o qual incidirá uma operação de quantização vetorial. Um elemento qualquer deste espaço é definido como o vetor de entrada $\mathbf{x}(t) \in \mathcal{X} \subset \mathbb{R}^p$, também chamado de vetor de estado, construído a partir de uma janela deslizante de comprimento p deslocando-se sobre o sinal de entrada, $\{x(t)\}_{t=1}^{N}$, ou seja,

$$\mathbf{x}(t) = \begin{pmatrix} x(t) \\ x(t-1) \\ \vdots \\ x(t-p+1) \end{pmatrix}. \tag{3.1}$$

Para realizar a quantização vetorial do espaço $\mathcal{X}$, cada neurônio i de uma rede SOM

qualquer possui um vetor de pesos $\mathbf{w}_i$, definido como

$$\mathbf{w}_i = \begin{pmatrix} w_{i1} \\ w_{i2} \\ \vdots \\ w_{ip} \end{pmatrix}, \qquad i = 1, 2, \ldots, q, \tag{3.2}$$

na qual q é o número total de neurônios da rede, semelhante à quantidade de neurônios na camada escondida na rede MLP. Associado a cada vetor de pesos existe um vetor de coeficientes $\mathbf{a}_i \in \mathcal{X} \subset \mathbb{R}^p$ contendo os coeficientes do i-ésimo filtro FIR linear, dado por

$$\mathbf{a}_i(t) = \begin{pmatrix} a_0(t) \\ a_1(t) \\ \vdots \\ a_{p-1}(t) \end{pmatrix}, \tag{3.3}$$

na qual p é a ordem do filtro FIR correspondente, semelhante ao valor visto no Capítulo 1, e que o vetor de coeficientes $\mathbf{a}_i(t)$ tem o mesmo formato visto na Figura 1.4. Assim, os parâmetros ajustáveis do algoritmo LLM são os conjuntos de vetores de pesos $\mathbf{w}_i(t)$ e seus respectivos vetores de coeficientes $\mathbf{a}_i(t)$, para $i = 1, \ldots, q$.

Seguindo a lógica competitiva da rede SOM, somente um neurônio por vez poderá ser usado para estimar a saída do algoritmo LLM. A seleção do neurônio vencedor se dá com base na distância euclidiana do seu vetor de pesos ao vetor de entrada, ou seja,

$$i^*(t) = \arg\min_{i \in \mathcal{A}} \|\mathbf{x}(t) - \mathbf{w}_i(t)\|. \tag{3.4}$$

Uma vez determinado o neurônio vencedor, a saída do algoritmo LLM é calculada da seguinte maneira:

$$\hat{y}(t) = \sum_{j=0}^{p-1} a_{i^*,j}(t) x(t-j) = \mathbf{a}_{i^*}^T(t)\mathbf{x}(t). \tag{3.5}$$

Resta, portanto, descrever o processo de aprendizagem do algoritmo LLM. Como se poderia esperar, o processo de adaptação dos vetores-protótipos e dos vetores de coeficientes associados a todos os neurônios segue a filosofia *cooperativa* da rede SOM, em que não apenas os parâmetros do neurônio vencedor, mas também os parâmetros dos neurônios vizinhos, são ajustados a cada iteração do algoritmo. Esta filosofia é levada a cabo por meio das seguintes regras de aprendizagem:

$$\mathbf{w}_i(t+1) = \mathbf{w}_i(t) + \alpha h(i^*, i; t)[\mathbf{x}(t) - \mathbf{w}_i(t)] \tag{3.6}$$

$$\mathbf{a}_i(t+1) = \mathbf{a}_i(t) + \alpha' h(i^*, i; t)\Delta\mathbf{a}_i, \tag{3.7}$$

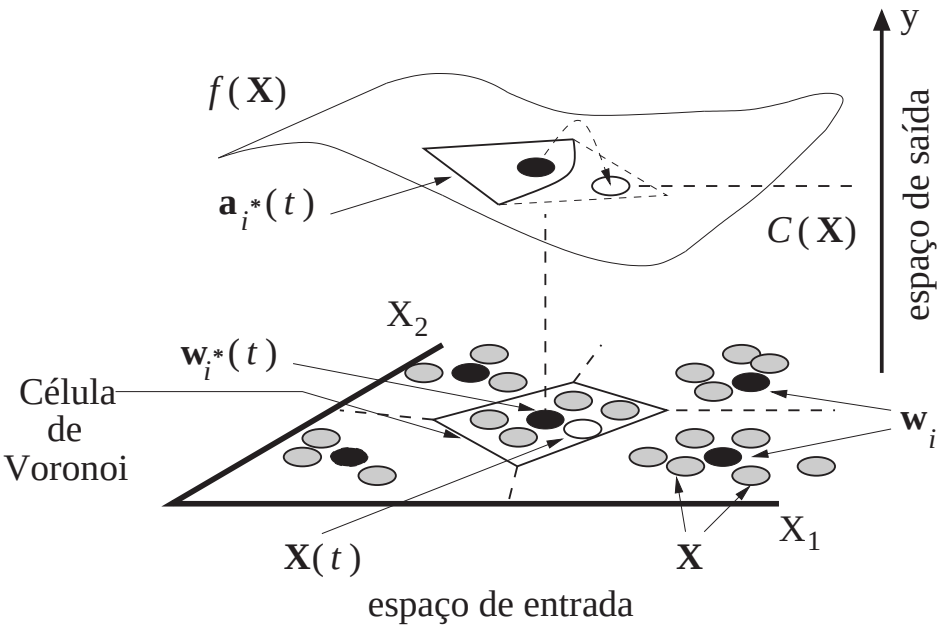


Figura 3.1 Representação da arquitetura LLM

na qual as constantes α e α' definem as taxas de aprendizagem dos vetores-protótipos e dos vetores de coeficientes dos neurônios, respectivamente, e $h(i^*, i; t)$ é a função vizinhança (ver Equações (2.42) e (2.43)) definida entre o neurônio vencedor i^* e os demais neurônios i na sua vizinhança topológica.

Para completar a descrição da Equação (3.7), tem-se ainda que especificar o termo de ajuste dos coeficientes $\Delta\mathbf{a}_i$. Ele é escolhido para minimizar o erro de estimação quadrático definido por:

$$\sum_t \left[y(t) - \mathbf{a}_{i^*}^T(t)\mathbf{x}(t)\right]^2 = \sum_t \left[y(t) - \hat{y}(t)\right]^2, \tag{3.8}$$

e é dado por uma regra de correção do erro do tipo LMS, representada por

$$\Delta\mathbf{a}_i = [y(t) - \mathbf{a}_i^T(t)\mathbf{x}(t)], \tag{3.9}$$

sendo $y(t)$ a saída real do mapeamento que se deseja aproximar.

Uma vez apresentado o algoritmo LLM, é importante e construtivo tecer alguns comentários sobre as principais características deste algoritmo, que o diferenciam das abordagens lineares tradicionais.

- O algoritmo LLM pode ser entendido como um filtro FIR cujo vetor de coeficientes a ser usado no instante t é escolhido entre q vetores de coeficientes, $\{\mathbf{a}_i(t)\}_{i=1}^q$, disponíveis naquele mesmo instante. A escolha de qual vetor de coeficientes usar é regida pela Equação (3.4). É importante perceber que se apenas 1 neurônio for utilizado (i.e. $q = 1$), a Equação (3.4) pode ser dispensada e o algoritmo LLM se

torna equivalente ao filtro linear FIR/LMS.

- Do ponto de vista computacional, pode-se dizer que o algoritmo LLM implementa uma *aproximação linear por região* da função não-linear que se deseja aproximar[1]. Ou seja, o sinal de entrada é vetorizado e o conjunto de vetores resultantes é quantizado pela rede SOM através de seus vetores-protótipos $\{\mathbf{w}_i\}_{i=1}^{q}$. Estes protótipos definem as coordenadas dos centróides de q regiões do espaço de entrada $\mathcal{X}$, chamadas de *células de Voronoi* (PRINCIPE et al., 2000) (ver Figura 3.1). Por sua vez, cada célula de Voronoi define o *campo receptivo* ou *região de atração* do i-ésimo neurônio, ou seja, a região do espaço de entrada para a qual o neurônio i é sempre escolhido vencedor. O que o algoritmo LLM faz é associar um vetor de coeficientes $\mathbf{a}_i$ a cada célula de Voronoi, tal que cada conjunto de coeficientes define um hiperplano aproximador da função não-linear de interesse.

[1]No presente trabalho, esta função corresponde ao modelo direto (identificação) ou o modelo inverso (equalização) do canal de comunicação.

Por fim, o sumário do funcionamento do algoritmo LLM é apresentado em seguida.

Sumário - Algoritmo LLM

- Para um dado instante de tempo t executar as seguintes operações:

  1. Definir o número de neurônios;
  2. Escolha do neurônio vencedor:

$$i^*(t) = \arg\min_{i \in \mathcal{A}} \|\mathbf{x}(t) - \mathbf{w}_i(t)\|,$$

  3. Atualização dos vetores de pesos e de coeficientes:

$$\begin{aligned} e_i(t) &= y(t) - \mathbf{a}_i^T(t)\mathbf{x}(t), \\ \Delta\mathbf{a}_i(t) &= e_i(t)\mathbf{x}(t), \\ \mathbf{w}_i(t+1) &= \mathbf{w}_i(t) + \alpha h(i^*, i; t)[\mathbf{x}(t) - \mathbf{w}_i(t)], \\ \mathbf{a}_i(t+1) &= \mathbf{a}_i(t) + \alpha' h(i^*, i; t)\Delta\mathbf{a}_i(t), \end{aligned}$$

  4. Cálculo da saída usando os coeficientes associados ao neurônio vencedor:

$$\hat{y}(t) = \mathbf{a}_{i^*}^T(t)\mathbf{x}(t).$$

Na próxima seção, será apresentado um outro algoritmo que utiliza a filosofia de treinamento da rede SOM, mas que, diferentemente do algoritmo LLM, não faz uso de algoritmos lineares clássicos de filtragem. Este algoritmo é de grande importância neste trabalho e será discutido com detalhes.

3.2 Memória Associativa Temporal por Quantização Vetorial

O algoritmo a ser descrito nesta seção, chamado de **Memória Associativa Temporal por Quantização Vetorial** (*Vector-Quantized Temporal Associative Memory* - VQTAM) (BARRETO; ARAúJO, 2004), utiliza a rede SOM para implementar a quantização vetorial simultânea dos espaços de entrada e saída, a partir dos pares entrada-saída $\{\mathbf{x}(t), y(t)\}$, $t = 1, \ldots, N$.

Em Barreto & Araújo (2004), o algoritmo VQTAM foi aplicado a problemas de modelagem e controle não-linear, predição de séries temporais, controle preditivo e aprendizagem de trajetórias robóticas. É importante enfatizar que, em todas estas tarefas, o

algoritmo VQTAM foi usado de modo *off-line*, em que primeiramente a rede era treinada com dados já disponíveis e depois testada (sem ajustes dos pesos) na tarefa de interesse. Nesta dissertação, o algoritmo VQTAM será pela primeira vez utilizado em tarefas que exigem um modo de aprendizado contínuo (*on-line*), tais como identificação e equalização de canais estacionários.

O algoritmo VQTAM é uma simples extensão da rede SOM original, em que o vetor de entrada $\mathbf{x}(t)$ passa a ser composto de duas partes, ou seja:

- A primeira parte, representada por $\mathbf{x}^{in}(t)$, corresponde à informação de entrada do mapeamento dinâmico que se quer aproximar.
- A segunda, representada por $\mathbf{x}^{out}(t)$, corresponde à informação de saída desse mesmo mapeamento.

Como conseqüência, os vetores de pesos dos neurônios também têm suas dimensões aumentadas. De maneira mais formal, tem-se a seguinte nova representação para $\mathbf{x}(t)$ e $\mathbf{w}_i(t)$:

$$\mathbf{x}(t) = \begin{pmatrix} \mathbf{x}^{in}(t) \\ \mathbf{x}^{out}(t) \end{pmatrix} \quad \text{e} \quad \mathbf{w}_i(t) = \begin{pmatrix} \mathbf{w}_i^{in}(t) \\ \mathbf{w}_i^{out}(t) \end{pmatrix}. \tag{3.10}$$

Dependendo das variáveis escolhidas para construir os vetores $\mathbf{x}^{in}(t)$ e $\mathbf{x}^{out}(t)$, pode-se usar o algoritmo SOM para aprender os modelos direto ou inverso de um dado sistema (e.g. canal de comunicação). Por exemplo, se o problema em mãos for identificação de canais (Figura 1.3(a)), então as seguintes definições se aplicam:

$$\mathbf{x}^{in}(t) = [x(t)\ x(t-1)\ \ldots\ x(t-p+1)]^T, \tag{3.11}$$

$$\mathbf{x}^{out}(t) = y(t), \tag{3.12}$$

em que $x(t)$ é o símbolo transmitido no instante t, $y(t)$ é a saída do canal correspondente, e $p > 1$ é a ordem do modelo direto.

Se a equalização do canal é requerida, então as seguintes definições se aplicam para que se possa aprender o mapeamento entrada-saída inverso:

$$\mathbf{x}^{in}(t) = [y(t)\ y(t-1)\ \ldots\ y(t-l+1)]^T, \tag{3.13}$$

$$\mathbf{x}^{out}(t) = x(t), \tag{3.14}$$

em que $l > 1$ é a ordem do modelo inverso.

Durante a execução do algoritmo, o neurônio vencedor na etapa de tempo t é deter-

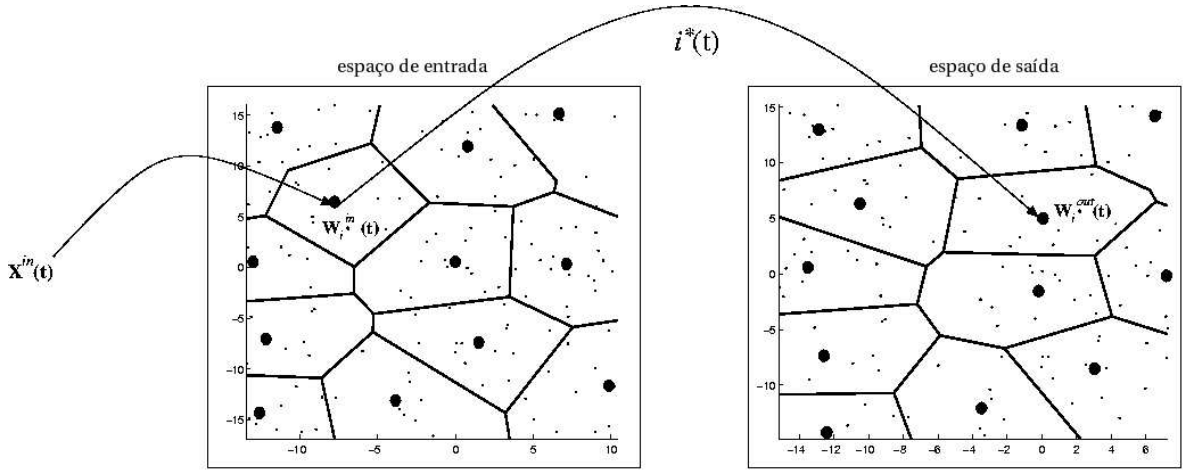


Figura 3.2 Representação da arquitetura VQTAM.

minado com base em $\mathbf{x}^{in}(t)$ apenas, logo,

$$i^*(t) = \arg\min_{i \in \mathcal{A}} \{\|\mathbf{x}^{in}(t) - \mathbf{w}_i^{in}(t)\|\}. \tag{3.15}$$

Na atualização dos pesos, os termos $\mathbf{x}^{in}(t)$ e $\mathbf{x}^{out}(t)$ são utilizados nas seguintes equações:

$$\mathbf{w}_i^{in}(t) = \mathbf{w}_i^{in}(t) + \alpha h(i^*, i; t)[\mathbf{x}^{in}(t) - \mathbf{w}_i^{in}(t)], \tag{3.16}$$

$$\mathbf{w}_i^{out}(t) = \mathbf{w}_i^{out}(t) + \alpha h(i^*, i; t)[\mathbf{x}^{out}(t) - \mathbf{w}_i^{out}(t)], \tag{3.17}$$

em que $0 < \alpha < 1$ é a taxa de aprendizagem e $h(i^*, i; t)$ é a função vizinhança definida na Seção 2.3.

É importante perceber que as regras de aprendizagem apresentadas nas Equações (3.16) e (3.17) seguem a formulação usual da rede SOM. A primeira regra executa uma quantização vetorial do espaço da entrada e a segunda atua de forma semelhante sobre o espaço da saída do mapeamento a ser aprendido. Assim, com o decorrer do treinamento, o algoritmo VQTAM aprende a associar os vetores-protótipos $\mathbf{w}_i^{in}(t)$ que formam o espaço quantizado de entrada com os vetores-protótipos $\mathbf{w}_i^{out}(t)$ que formam o espaço quantizado de saída (ver Figura 3.2).

Para que a estratégia de *memória associativa* implementada pelo algoritmo VQTAM seja útil em filtragem adaptativa, é necessário que a saída referente a um novo vetor de entrada possa ser estimada. Isto é feito a partir do vetor-protótipo do espaço de saída $\mathbf{w}_i^{out}(t)$ associado ao neurônio vencedor $i^*(t)$, ou seja,

$$\hat{\mathbf{y}}(t) \equiv \mathbf{w}_{i^*}^{out}(t), \tag{3.18}$$

na qual o neurônio vencedor $i^*(t)$ é determinado segundo a Equação (3.15).

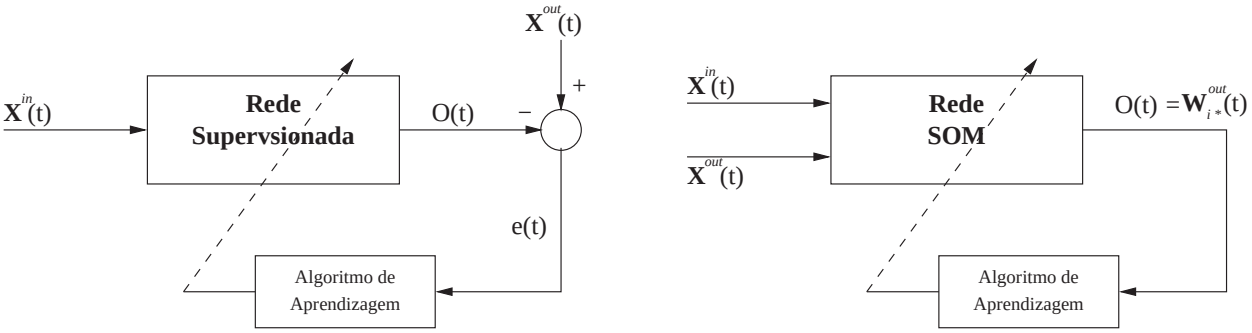


Figura 3.3 Diferença entre a abordagem supervisionada e a não-supervisionada na aproximação de funções.

É importante ressaltar alguns pontos importantes relativos à técnica VQTAM:

(*i*) O neurônio vencedor $i^*(t)$ é o elemento responsável por "associar" ou "conectar" a porção de entrada do mapeamento, representada por $\mathbf{x}^{in}(t)$, com a porção de saída $\mathbf{x}^{out}(t)$. Esta associação fica codificada nos respectivos vetores de pesos, $\mathbf{w}_i^{in}(t)$ e $\mathbf{w}_i^{out}(t)$.

(*ii*) Existe uma diferença *conceitual* muito importante entre a estratégia associativa do algoritmo VQTAM em relação àquela comumente usada no treinamento de redes supervisionadas (MLP ou RBF), que é baseada na redução *explícita* do erro de aproximação. Na abordagem supervisionada, o vetor $\mathbf{x}^{in}(t)$ é utilizado na entrada da rede, enquanto o vetor $\mathbf{x}^{out}(t)$ é utilizado na saída (ver Figura 3.3) para calcular o erro de aproximação usado para guiar o ajuste dos pesos da rede. Quando se usa o algoritmo VQTAM, o vetor $\mathbf{x}^{out}(t)$ é apresentado na entrada da rede juntamente com o vetor $\mathbf{x}^{in}(t)$, sem o cálculo *explícito* do erro de aproximação.

(*iii*) O algoritmo VQTAM, embora aplicado nesta dissertação apenas com a rede SOM, pode ser utilizada igualmente por outras RNAs competitivas, tais como a rede *Neural-Gas* (MARTINETZ; SCHULTEN, 1994).

(*iv*) A técnica VQTAM pode ser igualmente usada por redes competitivas crescentes, tais como a rede *Growing Neural-Gas* (FRITZKE, 1994) e *Growing SOM* (BAUER; VILLMANN, 1997), a fim de construir algoritmos neurais de filtragem não-linear que não necessitam de uma especificação prévia do número de neurônios.

Para finalizar, o sumário do algoritmo VQTAM é apresentado em seguida.

Sumário - Algoritmo VQTAM

- Para um dado instante de tempo t executar as seguintes operações:

  1. Definir o número de neurônios;
  2. Escolha do neurônio vencedor:

$$i^*(t) = \arg\min_{i \in \mathcal{A}} \|\mathbf{x}^{in}(t) - \mathbf{w}_i^{in}(t)\|,$$

  3. Atualização dos vetores de peso de entrada e saída para todos os neurônios da rede:

$$\begin{aligned} \mathbf{w}_i^{in}(t+1) &= \mathbf{w}_i^{in}(t) + \alpha h(i^*, i; t)[\mathbf{x}^{in}(t) - \mathbf{w}_i^{in}(t)], \\ \mathbf{w}_i^{out}(t+1) &= \mathbf{w}_i^{in}(t) + \alpha h(i^*, i; t)[\mathbf{x}^{out}(t) - \mathbf{w}_i^{out}(t)], \end{aligned}$$

  4. Cálculo da saída estimada:

$$\hat{\mathbf{y}}(t) \equiv \mathbf{w}_{i^*}^{out}(t).$$

Na próxima seção, serão apresentadas duas novas estratégias de projeto de algoritmos de filtragem adaptativa baseados na rede RBF, cujos parâmetros ajustáveis são automaticamente extraídos a partir do algoritmo VQTAM.

3.3 Projeto de Modelos RBF Usando VQTAM

Conforme discutido na seção anterior, o algoritmo VQTAM pode ser usado para os propósitos de aproximação de funções. Contudo, visto que ele é basicamente um método de quantização vetorial, ele pode necessitar de um número relativamente alto de neurônios para ter desempenho satisfatório. Nesta seção, é mostrado como projetar modelos RBF a partir de uma rede SOM treinada sob o esquema VQTAM.

Inicialmente, deriva-se um modelo RBF global para, em seguida, propor um modelo que usa menos neurônios, sendo por isso chamado de modelo RBF local. As relações dos modelos propostos com estratégias clássicas de projeto de redes RBF são também discutidas.

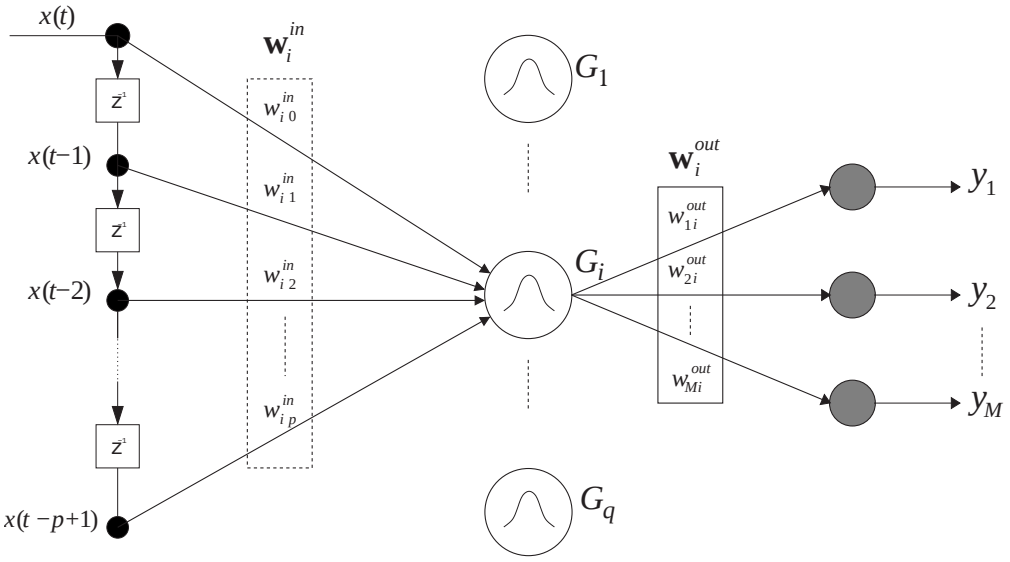


Figura 3.4 Filtro não-linear implementado por uma rede GRBF com q funções de base e M neurônios na camada de saída.

3.3.1 Um Modelo RBF Global

Assumindo que uma rede SOM com q neurônios foi treinada segundo o algoritmo VQTAM, o valor da saída de um modelo RBF geral com q funções de base e $M \geq 1$ neurônios de saída pode ser implementado como segue:

$$\hat{\mathbf{y}}(t) \equiv \hat{\mathbf{y}}(\mathbf{w}_i^{in}, \mathbf{w}_i^{out}, \mathbf{x}^{in}(t)) = \frac{\sum_{i=1}^{q} \mathbf{w}_i^{out} G_i(\mathbf{w}_i^{in}, \mathbf{x}^{in}(t))}{\sum_{i=1}^{q} G_i(\mathbf{w}_i^{in}, \mathbf{x}^{in}(t))} \tag{3.19}$$

$$= \begin{pmatrix} \dfrac{\sum_{i=1}^{q} w_{1i}^{out}\, G_i(\mathbf{w}_i^{in}, \mathbf{x}^{in}(t))}{\sum_{i=1}^{q} G_i(\mathbf{w}_i^{in}, \mathbf{x}^{in}(t))} \\ \vdots \\ \dfrac{\sum_{i=1}^{q} w_{Mi}^{out}\, G_i(\mathbf{w}_i^{in}, \mathbf{x}^{in}(t))}{\sum_{i=1}^{q} G_i(\mathbf{w}_i^{in}, \mathbf{x}^{in}(t))} \end{pmatrix}, \tag{3.20}$$

na qual $\hat{\mathbf{y}} = [\hat{y}_1 \quad \cdots \quad \hat{y}_M]^T$ é o vetor de saídas estimadas, $\mathbf{w}_i^{out}$ é o vetor contendo os pesos que conectam a i-ésima função de base aos M neurônios de saída (ver Figura 3.4), e $G_i(\mathbf{w}_i^{in}, \mathbf{x}^{in}(t))$ é a resposta da i-ésima função de base ao vetor de entrada atual $\mathbf{x}^{in}(t)$, ou seja,

$$G_i(\mathbf{w}_i^{in}, \mathbf{x}^{in}(t)) = \exp\left(-\frac{\|\mathbf{x}^{in}(t) - \mathbf{w}_i^{in}\|^2}{2\gamma^2}\right), \tag{3.21}$$

na qual o vetor-protótipo $\mathbf{w}_i^{in}$ é usado como centro da i-ésima função de base. A constante γ, calculada conforme a Equação (2.35), define a abertura (*spread*) da gaussiana, sendo atribuído o mesmo valor para todas as funções de base.

É importante notar que, no algoritmo GRBF, a saída é a média ponderada do vetor

$\mathbf{w}_i^{out}$ dos protótipos da saída associado com os protótipos da entrada $\mathbf{w}_i^{in}$ próximo ao dado vetor de entrada $\mathbf{x}^{in}$, como indicado pela Equação (3.19). Nesta equação, todos os q neurônios da camada oculta (funções de base) são usados para estimar a saída correspondente. Neste sentido, refere-se ao modelo RBF ora apresentado como modelo *RBF Global* (GRBF, do inglês *Global RBF*), apesar da natureza localizada de cada base gaussiana.

Algumas observações importantes sobre a sistemática de projeto da rede GRBF são destacadas a seguir:

- A formulação da rede GRBF apresentada é bastante geral, podendo ser utilizada para identificar sistemas MIMO, i.e. com múltiplas entradas e múltiplas saídas. Para os problemas de filtragem adaptativa de interesse para esta dissertação, apenas um neurônio de saída é utilizado. Para estes casos, basta fazer $M = 1$ na Equação (3.20).

- O projeto usual de uma rede RBF se dá em duas fases (PRINCIPE et al., 2000). Primeiramente, os centros das funções de base são determinados por meio de técnicas de agrupamento (*clusterização*) sobre os vetores $\mathbf{x}^{in}$ (por exemplo, usando o algoritmo K-médias) e, então, os pesos ligando a camada escondida à de saída são calculados por meio da regra LMS (ou o método da pseudo-inversa). O modelo GRBF proposto requer apenas uma única fase de aprendizagem, em que a rede SOM executa uma quantização vetorial simultânea sobre pares de entrada-saída $\{\mathbf{x}^{in}(t), \mathbf{x}^{out}(t)\}$.

- É interessante também comparar a rede GRBF com duas estratégias bastante comuns de projeto da rede RBF, a saber:

  - **Rede Neural para Regressão Generalizada** (GRNN), proposta por Specht (1991).
  - **Rede Neural Probabilística Modificada** (MPNN), proposta por Zaknich (1998).

  Os algoritmos MPNN e GRNN compartilham da mesma origem teórica e da estrutura básica da rede. No modelo GRNN, há uma função de base centrada no vetor de entrada $\mathbf{x}^{in}$, e os pesos da camada escondida para a de saída são apenas os vetores-alvo $\mathbf{x}^{out}$ que compõem os vetores pares entrada-saída de treinamento. A diferença é que o MPNN usa o algoritmo K-médias para determinar seus centros. Para ambos, a saída é simplesmente uma média ponderada dos vetores-alvo $\mathbf{x}^{out}$ de treinamento.

A vantagem imediata do GRBF sobre o GRNN é uma redução no número de funções de base necessárias para construir uma rede RBF, que é muito importante em cenários que demandam processamento em tempo-real. Uma outra vantagem do algoritmo GRBF

está no fato de ele ser mais robusto ao ruído e generalizar melhor que o modelo GRNN devido o processo de quantização vetorial feito pela estrutura VQTAM, uma vez que ele usa os vetores-protótipos $\{\mathbf{w}_i^{in}, \mathbf{w}_i^{out}\}_{i=1}^{q}$ aprendidos pela rede SOM.

3.3.2 Um Modelo RBF Local

No modelo GRBF, os pares entrada-saída $\{\mathbf{x}^{in}(t), \mathbf{x}^{out}(t)\}_{t=1}^{N}$, são mapeados em um número finito de regiões quantizadas (células de Voronoi), cada uma associada com um par de vetores-protótipos $(\mathbf{w}_i^{in}, \mathbf{w}_i^{out})$ que representa grosseiramente a dinâmica *local* da i-ésima célula de Voronoi. Para cada novo vetor de entrada, *todos* estes protótipos são usados para estimar a saída correspondente como mostrado na Equação (3.19). Neste sentido é que se refere ao modelo RBF resultante como sendo de natureza *global*, conforme mencionado na seção anterior.

Vários estudos demonstram que mapeamentos entrada-saída não-lineares podem ser eficientemente aproximados através de modelos neurais *locais* (PRINCIPE et al., 1998; ERDOGMUS et al., 2001; BARRETO et al., 2004). De acordo com esta abordagem somente poucas regiões dos espaços de entrada e saída modelados são usadas para estimar a saída para um novo vetor de entrada. Isto significa que basta um número $1 < K \ll q$ de protótipos para servir como centros das funções de base e de pesos da camada de saída de um modelo RBF.

Como estratégia de projeto de tal modelo RBF local, propõe-se aqui utilizar o conjunto dos K neurônios vencedores, simbolizados $\{i_1^*, i_2^*, \ldots, i_K^*\}$, ou seja aqueles K neurônios cujos vetores de peso estão mais próximos do vetor de entrada atual $\mathbf{x}^{in}(t)$. Matematicamente, esta idéia pode ser formulada como segue:

$$\begin{aligned}
i_1^*(t) &= \arg\min_{\forall i} \left\{\|\mathbf{x}^{in}(t) - \mathbf{w}_i^{in}\|\right\} = \arg\max_{\forall i} \left\{G_i(\mathbf{w}_i^{in}, \mathbf{x}^{in}(t))\right\} \\
i_2^*(t) &= \arg\min_{\forall i \neq i_1^*} \left\{\|\mathbf{x}^{in}(t) - \mathbf{w}_i^{in}\|\right\} = \arg\max_{\forall i \neq i_1^*} \left\{G_i(\mathbf{w}_i^{in}, \mathbf{x}^{in}(t))\right\} \\
&\vdots \qquad\qquad \vdots \qquad\qquad \vdots \\
i_K^*(t) &= \arg\min_{\forall i \neq \{i_1^*, \ldots, i_{K-1}^*\}} \left\{\|\mathbf{x}^{in}(t) - \mathbf{w}_i^{in}\|\right\} = \arg\max_{\forall i \neq \{i_1^*, \ldots, i_{K-1}^*\}} \left\{G_i(\mathbf{w}_i^{in}, \mathbf{x}^{in}(t))\right\}.
\end{aligned} \tag{3.22}$$

Isto posto, a saída estimada passa a ser calculada como segue:

$$\hat{\mathbf{y}}(t) \equiv \frac{\sum_{k=1}^{K} \mathbf{w}_{i_k^*}^{out} G_{i_k^*}(\mathbf{w}_i^{in}, \mathbf{x}^{in}(t))}{\sum_{k=1}^{K} G_{i_k^*}(\mathbf{w}_i^{in}, \mathbf{x}^{in}(t))}. \tag{3.23}$$

Assim, refere-se a esta arquitetura neural como modelo **RBF Local sobre K-vencedores** (*Local RBF over K-winners* - KRBF).

Além disso, vale ressaltar que os algoritmos VQTAM e o GRBF podem ser entendidos

como instâncias particulares do modelo KRBF, conforme demonstrado a seguir:

- Assumindo $K = 1$, então a Equação (3.23) reduz-se a

$$\hat{\mathbf{y}}(t) = \frac{\sum_{k=1}^{1} \mathbf{w}_{i_k^*}^{out} G_{i_k^*}(\mathbf{w}_i^{in}, \mathbf{x}^{in}(t))}{\sum_{k=1}^{1} G_{i_k^*}(\mathbf{w}_i^{in}, \mathbf{x}^{in}(t))} \tag{3.24}$$

$$= \frac{\mathbf{w}_{i_1^*}^{out} G_{i_1^*}(\mathbf{w}_i^{in}, \mathbf{x}^{in}(t))}{G_{i_1^*}(\mathbf{w}_i^{in}, \mathbf{x}^{in}(t))}$$

$$= \mathbf{w}_{i_1^*}^{out}(t), \tag{3.25}$$

  na qual o valor estimado da saída é o mesmo que aquele mostrado na Equação (3.18).

- Se for feito $K = q$, então a Equação (3.23) se transforma em

$$\hat{\mathbf{y}}(t) \equiv \frac{\sum_{k=1}^{q} \mathbf{w}_{i_k^*}^{out}(t) G_{i_k^*}(\mathbf{w}_i^{in}, \mathbf{x}^{in}(t))}{\sum_{k=1}^{q} G_{i_k^*}(\mathbf{w}_i^{in}, \mathbf{x}^{in}(t))}, \tag{3.26}$$

  que é computacionalmente equivalente ao modelo global descrito na Equação (3.19), uma vez que encontrar q vencedores é equivalente a usar *todos* os neurônios disponíveis para construir o modelo RBF.

Na pesquisa bibliográfica levada a cabo nesta dissertação, não foram encontrados muitos trabalhos propondo modelos locais baseados na rede RBF. Em Chng et al. (1996), por exemplo, um modelo GRNN é primeiro construído e, então, somente os centros que estão a uma certa distância $\epsilon > 0$ do vetor de entrada corrente são usados para estimar a saída. Esta idéia é basicamente a mesma daquela usada para construir o KRBF. Porém, ela padece da mesma limitação do modelo GRNN com relação ao seu alto custo computacional. Além disso, dependendo do valor de ϵ, o número de centros selecionados pode variar consideravelmente (de forma imprevisível) a cada instante de tempo. Se ϵ for muito pequeno, pode acontecer de nenhum dos centros ser selecionado no total. Isto nunca ocorre para o modelo KRBF, pois o mesmo número de K centros é selecionado a cada passo de tempo.

Mais recentemente, outro modelo RBF local foi proposto em Dablemont et al. (2003) baseado em um procedimento de dupla quantização vetorial. Primeiramente, este método quantiza a informação de entrada $\{\mathbf{x}^{in}(t)\}$ apresentada à rede SOM em um pequeno grupo (*cluster*), formando assim um mapa de características da entrada. Em seguida, é feito um novo mapeamento agora sobre as informações do espaço de entrada e saída através da apresentação de pares (ou agrupamentos) $\{\mathbf{x}^{in}(t), \mathbf{x}^{out}(t)\}$ a cada neurônio de uma segunda rede SOM, da mesma forma que é feito com a rede VQTAM, construindo portanto uma

associação dinâmica de informações entre os protótipos dos dois mapeamentos feitos, tal que q modelos locais RBF são construídos, um para cada neurônio. No algoritmo KRBF, por outro lado, somente um único modelo local RBF é construído dinamicamente a cada instante t, a partir dos K vetores-protótipos mais próximos do vetor de entrada atual, ou equivalentemente, a partir das K funções de base gaussianas de maior ativação.

Na próxima seção, será apresentado um outro modelo local, também baseado na rede VQTAM, mas que implementa um aproximador linear local cujos coeficientes são computados através do método dos quadrados mínimos.

3.4 Um Modelo Linear Local sobre K-vencedores

Após usar o modelo VQTAM para desenvolver uma estratégia de projeto de modelos locais não-lineares com base na arquitetura RBF, nesta seção o modelo VQTAM será usado para projetar modelos locais lineares, para uso subseqüente em filtragem adaptativa. Mais especificamente, o modelo a ser descrito utiliza uma abordagem semelhante àquela do algoritmo KRBF, no que diz respeito à escolha de K neurônios vencedores, a cada instante t, sendo por isso doravante chamado de **Mapeamento Linear Local sobre K-vencedores** (*Local Linear Mapping over K-winners* - KSOM). Porém, há diferenças entre os dois no que diz respeito à natureza do mapeamento local construído. Enquanto que no KRBF as funções de base implementam uma transformação não-linear entre o espaço de entrada e o de saída, no modelo KSOM esta transformação é de natureza linear. Mais detalhes são dados a seguir.

O modelo KSOM foi proposto por Barreto et al. (2003) para aplicação em predição de séries temporais, sendo aqui estendido para aplicações em filtragem adaptativa. Este modelo é construído a partir do modelo VQTAM. A idéia subjacente é usar os pares de vetores-protótipos $\{\mathbf{w}_{i_k^*}^{in}(t), w_{i_k^*}^{out}(t)\}_{k=1}^{K}(t)$ dos K neurônios-vencedores no instante t para construir um aproximador linear local de funções, ou seja,

$$w_{i_k^*}^{out} = \mathbf{a}^T(t)\mathbf{w}_{i_k^*}^{in}(t), \qquad k = 1, \ldots, K, \tag{3.27}$$

em que $\mathbf{a}(t) = [a_0(t)\ a_1(t)\ \cdots a_{p-1}(t)]^T$ é um vetor de coeficientes variante no tempo. A Equação (3.27) pode ser escrita na forma matricial da seguinte maneira:

$$\mathbf{p}(t) = \mathbf{R}(t)\mathbf{a}(t), \tag{3.28}$$

tal que o vetor $\mathbf{p}$, chamado de vetor de predição, e a matriz $\mathbf{R}$, chamada de matriz de regressão, são definidos como

$$\mathbf{p}(t) \;=\; [w_{i_1^*,1}^{out}(t)\ \ w_{i_2^*,1}^{out}(t)\ \ \cdots\ \ w_{i_K^*,1}^{out}(t)]^T \tag{3.29}$$

$$\mathbf{R}(t) = \begin{pmatrix} w^{in}_{i^*_1,1}(t) & w^{in}_{i^*_1,2}(t) & \cdots & w^{in}_{i^*_1,p}(t) \\ w^{in}_{i^*_2,1}(t) & w^{in}_{i^*_2,2}(t) & \cdots & w^{in}_{i^*_2,p}(t) \\ \vdots & \vdots & \vdots & \\ w^{in}_{i^*_K,1}(t) & w^{in}_{i^*_K,2}(t) & \cdots & w^{in}_{i^*_K,p}(t) \end{pmatrix}. \tag{3.30}$$

Se $p = K$ na Equação (3.30), ou seja, se a dimensão do espaço de entrada é igual ao número de neurônios vencedores utilizados, então a matriz $\mathbf{R}$ é quadrada. Neste caso, o vetor de coeficientes $\mathbf{a}$ pode ser calculado simplesmente invertendo-se $\mathbf{R}$, ou seja,

$$\mathbf{a}(t) = \mathbf{R}^{-1}(t)\mathbf{p}(t). \tag{3.31}$$

Contudo, a situação mais usual ocorre para $p < K$, ou seja, para $\mathbf{R}$ sendo uma matriz não-quadrada. Neste caso, pode-se utilizar a técnica dos *Quadrados Mínimos* (QM) ou da *Pseudo-inversa* (AGUIRRE, 2000), tornando possível a inversão da matriz $\mathbf{R}$ condicionada ao caso em que ela tenha posto completo. De acordo com esta técnica, o vetor de coeficientes $\mathbf{a}(t)$ é determinado da seguinte maneira

$$\mathbf{a}(t) = \left(\mathbf{R}^T(t)\mathbf{R}(t)\right)^{-1}\mathbf{R}^T(t)\mathbf{p}(t). \tag{3.32}$$

Uma vez calculado $\mathbf{a}(t)$ através da Equação (3.32), pode-se estimar a saída do algoritmo KSOM como sendo a de um filtro FIR (ver Figura 1.4) de ordem p, ou seja,

$$\hat{y}(t) = \mathbf{a}^T(t)\mathbf{x}^{in}(t) = \sum_{j=0}^{p-1} a_j(t)x(t-j). \tag{3.33}$$

Em suma, para cada vetor de entrada $\mathbf{x}^{in}(t)$ determina-se um vetor de coeficientes $\mathbf{a}(t)$, calculado pela Equação (3.32), que será usado para construir o estimador linear mostrado na Equação (3.33).

É importante ressaltar as diferenças do modelo KSOM em relação ao modelo LLM, que também implementa um aproximador linear local. A principal diferença está na forma como são calculados os coeficientes do filtro FIR linear de cada modelo neural. No algoritmo LLM, cada neurônio tem um vetor de coeficientes associado, resultando num conjunto de q vetores-coeficientes. Uma vez encontrado o neurônio vencedor, basta usar a Equação (3.5). Já no algoritmo KSOM um único vetor de coeficientes a cada instante t é calculado dinamicamente usando os vetores de pesos de K neurônios vencedores.

Um outro modelo linear local, muito semelhante ao KSOM, foi proposto por Principe et al. (1998). Este modelo, aqui chamado de **Modelo SOM Local Usando Quadrados Mínimos** (*Local Least-Squares SOM* - LESSOM), também seleciona K neurônios para

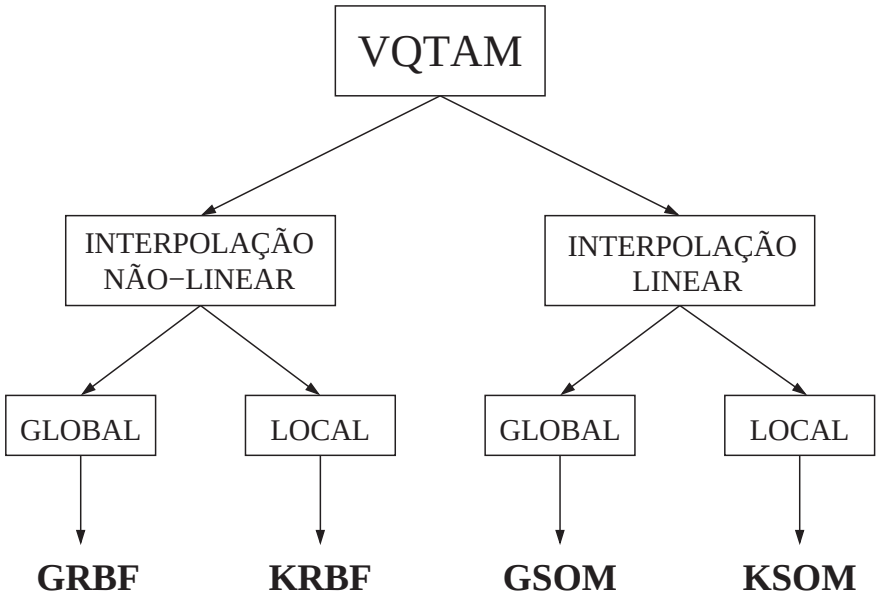


Figura 3.5 Taxonomia dos modelos derivados da técnica VQTAM.

construir o modelo linear local através das Equações (3.31) e (3.33). Contudo, em vez de selecionar os K protótipos mais próximos do vetor de entrada $\mathbf{x}^{in}(t)$, são selecionados os protótipos do neurônio vencedor e os protótipos de seus $(K-1)$ vizinhos. Por causa da propriedade de preservação de topologia da rede SOM, os K neurônios selecionados no modelo KSOM são os mesmos do modelo LESSOM. A vantagem do algoritmo KSOM é que ele pode ser utilizado por qualquer rede neural competitiva, enquanto que o algoritmo LESSOM só é válido para a rede SOM.

A título de ilustração, é importante enfatizar mais uma vez ao fim desta seção que os modelos GRBF, KRBF e KSOM são oriundos da técnica VQTAM, que permitiu o uso de algoritmos não-supervisionados competitivos em problemas de filtragem adaptativa, como aproximadores de funções não-lineares. A Figura 3.5 ilustra melhor a relação entre os modelos GRBF/KRBF/KSOM e o modelo VQTAM, em função do tipo de modelagem utilizada (global ou local). Sendo a rede KSOM um modelo local, para $K = q$ tem-se a obtenção de um modelo global que faz interpolação linear baseado na rede SOM, este modelo é o GSOM (*Global SOM*). Isto se assemelha ao caso visto entre as redes KRBF e GRBF, na qual se obteve um modelo global (GRBF) a partir de um local (KRBF), sendo que neste caso as redes realizam interpolação não-linear. A rede GSOM foi proposta inicialmente por Barreto & Araújo (2004). A seguir, discute-se uma forma mais geral de se entender modelos locais ou globais baseados na rede SOM como uma rede de especialistas.

3.5 Modelos Locais como Redes de Especialistas

Nesta seção, busca-se uma interpretação dos modelos introduzidos neste capítulo e no anterior como membros de uma classe de arquiteturas de redes neurais que utiliza mecanismos dinâmicos de *cooperação* ou de *competição* entre neurônios.

Inicialmente, no Capítulo 2 foram descritas topologias de redes com múltiplos elementos processadores (neurônios) que eram treinados de uma forma cooperativa, isto é, todos os neurônios eram treinados ao mesmo tempo com o objetivo em comum de solucionar o problema. Como exemplos, têm-se as redes MLP e RBF em que os pesos de todos os neurônios são atualizados ao mesmo tempo.

Neste capítulo, foram apresentadas outras topologias de redes cujos neurônios implementam um processo competitivo entre eles, no qual um determinado grupo se especializa em modelar (codificar ou representar) uma porção do espaço de dados de entrada e que, no processo de atualização dos pesos, nem todos os neurônios participam de uma forma igualitária, tornando o processo de aprendizagem da rede algo centralizado naqueles elementos pertencentes ao grupo dominante, em torno do neurônio vencedor. Ainda há um certo grau de cooperação, mas extremamente localizado dentro do pequeno grupo de neurônios vizinhos ao neurônio vencedor.

Dá-se o sugestivo nome de *neurônios especialistas* àqueles responsáveis pela modelagem de uma pequena porção do espaço de entrada e que, para estimar a saída do modelo, usam apenas seus próprios parâmetros (pesos) ou de um pequeno grupo de neurônios (PRINCIPE et al., 2000; HAYKIN, 1994). Como exemplos de redes que se utilizam de especialistas, doravante chamadas simplesmente de *Redes de Especialistas*, tem-se a rede SOM e todas as redes que se baseiam na sua arquitetura apresentadas por este trabalho, tais como LLM, VQTAM, GRBF, KRBF e KSOM.

De modo mais formal, Redes de Especialistas (*expert networks*) (JACOBS et al., 1991) são construídas a partir de elementos altamente especializados, tal que quando um vetor de entrada é apresentado a todos os neurônios, apenas alguns deles vão permanecer ativados graças a uma seleção coordenada por uma rede competitiva, chamado disparador. Uma arquitetura geral de uma Rede de Especialistas é mostrada na Figura 3.6. Nesta figura pode-se perceber que a saída de uma Rede de Especialistas genérica pode ser calculada como

$$\hat{y}(t) = \sum_{k=1}^{K} g_k(t)\hat{y}_k(t), \tag{3.34}$$

em que g_k é um fator de ponderação que assume valor Um (1) ou Zero (0), indicando se o k-ésimo especialista vai ou não contribuir com sua parcela ($\hat{y}_k$) para a saída da Rede de Especialistas, respectivamente.

Em geral, a Rede de Seleção (ou Rede *gate*) é implementada de duas maneiras, a fim de escolher os elementos especialistas. Na primeira ela escolhe um só elemento especialista (ou vencedor) que fica responsável sozinho pela geração da saída. A este tipo de procedimento dá-se o nome de mecanismo de seleção (*Winner-Take-All* - WTA). A segunda abordagem lança mão de procedimentos que permitem a seleção de mais de um especialista (talvez até todos eles!) para estimar a saída da rede. A este procedimento dá-se o nome de mecanismo de seleção (*K-Winners-Take-All* - KWTA).

O objetivo do restante desta seção é descrever cada um dos algoritmos descritos nas seções anteriores em função da arquitetura geral de Redes de Especialistas mostrada na Figura 3.6. Acredita-se que esta iniciativa permitirá aumentar a compreensão da forma particular com que cada um dos modelos apresentados calcula a sua saída.

Rede LLM - A rede LLM é uma rede de especialistas que utiliza o procedimento de seleção WTA. Analisando a Equação (3.34), isto corresponde à seguinte escolha de valores para $g_k(t)$:

$$g_k(t) = \begin{cases} 1, & \text{se } k = i^*(t) \\ 0, & \text{se } k \neq i^*(t) \end{cases} \tag{3.35}$$

em que $i^*(t)$ é encontrado de acordo com a Equação (3.4). Isto equivale a dizer que a rede *gate* escolhe o especialista como sendo o neurônio vencedor atual. Finalmente, a saída do especialista selecionado é, no caso da rede LLM, calculada pela Equação (3.5).

Rede VQTAM - Este algoritmo funciona também segundo o mecanismo WTA de seleção de especialistas WTA. Neste caso, o índice do especialista $i^*(t)$, usado na Equação (3.35), é encontrado segundo a Equação (3.15) e a sua saída segundo a Equação (3.18).

Rede GRBF - No modelo GRBF, a rede *gate* habilita todas as saídas dos especialistas, permitindo uma participação de todos os elementos no cálculo da saída estimada, conforme mostrado na Equação (3.19). Contudo, as funções de base gaussiana restringem automaticamente o grau de participação de cada especialista na Equação (3.34).

Rede KRBF - Neste modelo o grau de participação dos especialistas fica ainda mais restrito através da seleção, por parte da rede *gate*, de apenas K funções de base (especialistas) cujos centros estejam mais próximos do vetor de dados de entrada atual (ver Equação (3.22)). Isto equivale a fazer $g_k(t) \equiv g_{i_k^*}(t) = w_{i_k^*}^{out}$, $k = 1, \ldots, K$, na Equação (3.34). As saídas $g_{i_k^*}(t)$ correspondentes são dadas por:

$$\hat{y}_k(t) \equiv \hat{y}_{i_k^*}(t) = \frac{G_{i_k^*}(\mathbf{w}_i^{in}, \mathbf{x}^{in}(t))}{\sum_{k=1}^{K} G_{i_k^*}(\mathbf{w}_i^{in}, \mathbf{x}^{in}(t))}, \qquad k = 1, \ldots, K. \tag{3.36}$$

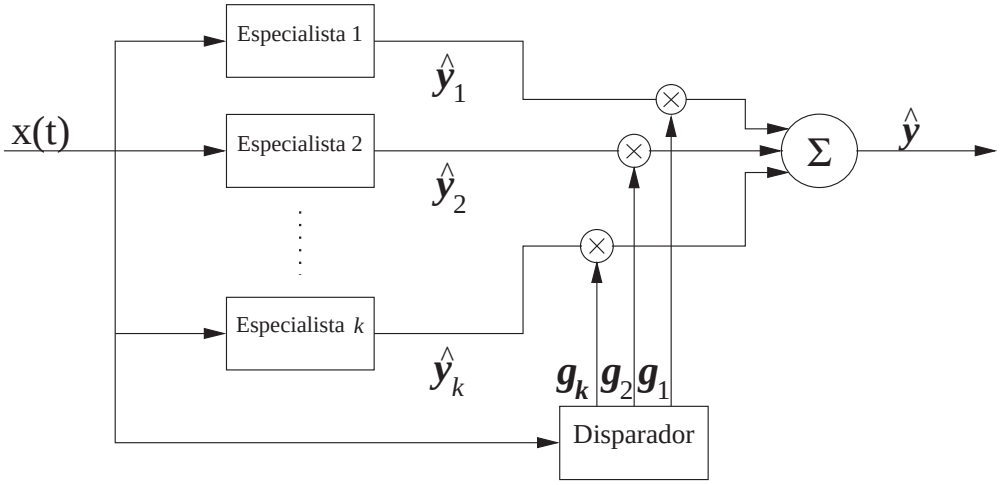


Figura 3.6 Uma rede de especialistas.

Rede KSOM - Por fim, o algoritmo KSOM tem seus K especialistas selecionados segundo o mesmo mecanismo de seleção (KWTA) do algoritmo KRBF. Com os vetores-protótipos destes K especialistas, o vetor de coeficientes $\mathbf{a}(t)$ é calculado de acordo com a Equação (3.32). A saída da rede de especialistas é então dada pela Equação (3.33).

3.6 Conclusão

Neste capítulo, pôde-se estudar os modelos neurais citados anteriormente que são baseados na rede de Kohonen e que serão utilizados nos problemas propostos por este trabalho nos últimos capítulos. Cada um dos algoritmos vistos adotaram a forma de treinamento da rede SOM.

Em primeiro lugar, foi estudado o algoritmo LLM onde foi utilizado um vetor de coeficientes, escolhido a partir do índice do neurônio vencedor, para poder aproximar a saída através de um processo de interpolação linear. O algoritmo VQTAM foi apresentado em seguida, cuja arquitetura tornou possível a aplicação da rede SOM como aproximador de funções em mapeamentos dinâmicos e o uso desta no problema de filtragem adaptativa.

E continuando com os demais algoritmos, a rede RBF utilizou a mesma técnica VQTAM para poder gerar mais dois modelos em que usam informação global e local mapeados pelos protótipos da rede. As redes GRBF e KRBF são arquiteturas que realizam aproximação de funções também, e que utilizam funções de base gaussianas para estimar a saída, diferente do VQTAM que adota uma quantização do espaço de saída para poder definir o valor estimado. Mostrou-se que os algoritmos VQTAM e GRBF podem ser entendidos como casos particulares do algoritmo KRBF.

Por último, o algoritmo VQTAM é aplicado ao problema de filtragem adaptativa definindo modelos locais lineares (filtros FIR locais) através dos protótipos dos pesos gerados pela quantização vetorial simultânea dos espaços de dados de entrada e de saída. Outro fato importante foi a interpretação de todas as redes competitivas usadas neste capítulo como sendo modelos de redes de especialistas.

No próximo capítulo, será abordado o uso das redes neurais vistas neste capítulo, assim como os demais algoritmos vistos nos capítulos anteriores, aplicados ao problema de identificação de canais não-lineares.

4 IDENTIFICAÇÃO DE CANAIS USANDO REDES NEURAIS

4.1 Introdução

Após a introdução teórica feita em capítulos anteriores para os modelos de filtros transversais lineares e para filtros baseados em redes neurais artificiais, este capítulo se detém na apresentação dos resultados obtidos pela aplicação desses filtros no problema de identificação de canais de comunicação.

O desempenho de cada filtro estudado será avaliado principalmente segundo sua capacidade de aproximar a relação (mapeamento) entrada-saída de um canal de comunicação não-linear, cujo modelo simulado é apresentado na próxima seção. Além disso, todos os filtros em questão serão comparados conforme a sua velocidade de convergência e a sua sensibilidade (ou dependência) em relação a variações de alguns parâmetros de treinamento, tais como o número de neurônios utilizados e capacidade de generalização.

Ao longo da apresentação dos resultados, diversos pontos de discussão serão abordados com respeito aos modelos propostos, para então se chegar às conclusões finais sobre o emprego dos mesmos.

4.2 Modelo do Canal de Comunicação

O processo gerador das seqüências de variáveis aleatórias discretas de entrada $\{x(t)\}$ e de saída $\{y(t)\}$, $t = 1, \ldots, N$ é descrito a seguir.

Primeiramente, a seqüência de símbolos $\{x(t)\}$, a ser enviada pelo canal, é modelada como um processo aleatório Gauss-Markov de primeira ordem, ou seja (DONG et al., 2003; MISRA et al., 2003):

$$x(t) = ax(t-1) + b\varepsilon(t), \qquad t = 1, \ldots, N, \tag{4.1}$$

na qual $x(t) \in \mathbb{R}$ é o símbolo a ser transmitido no instante t, $\varepsilon(t) \sim \mathcal{N}(0, \sigma_\varepsilon^2)$ é uma

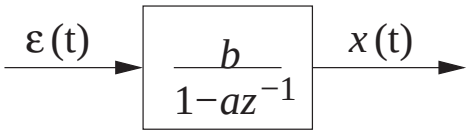


Figura 4.1 Geração do sinal auto-regressivo Gauss-Markov.

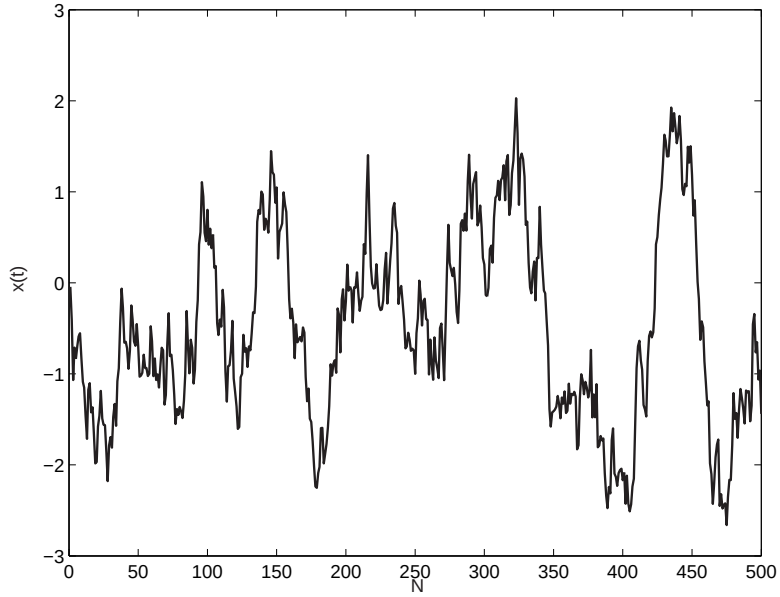


Figura 4.2 Amostra do sinal de entrada auto-regressivo Gauss-Markov.

seqüência de ruído branco gaussiano, a e b são constantes, e N é o comprimento total da seqüência. Em particular, deve-se ter $|a| < 1$ para garantir estacionariedade do processo[1].

A Equação (4.1) estabelece que a seqüência de entrada $\{x(t)\}$ deve ser entendida como a saída de um filtro IIR de primeira ordem tendo um processo gaussiano como entrada, conforme ilustrado na Figura 4.1.

Assim, a Equação (4.1) produz um sinal discreto gaussiano de média zero e potência σ_x^2, ou seja $x(t) \sim N(0, \sigma_x^2)$, tal que amostras $x(t-1)$ e $x(t+1)$ são condicionalmente independentes de $x(t)$. A potência σ_x^2 da seqüência de símbolo é definida como

$$\sigma_x^2 = \frac{b^2}{1-a^2}\,\sigma_\varepsilon^2. \tag{4.2}$$

Nota-se que, para $a = 0$ e $b = 1 \Rightarrow x(t) = \varepsilon(t)$, ou seja, o sinal de entrada reduz-se a uma seqüência de ruído branco gaussiano. Uma realização de x para a=0,95 e b=0,10 é mostrada na Figura 4.2.

Uma vez definido o sinal de entrada, passa-se à especificação do canal propriamente dito. Esta fase tem início com a adição de ruído a cada observação discreta do sinal de

[1]Sem perda de generalidade, assume-se que o tempo de processamento entre símbolos é maior que o tempo de processamento dos filtros adaptativos a serem simulados. Além disso, não se adota nenhum tipo de codificação para o sinal a ser transmitido.

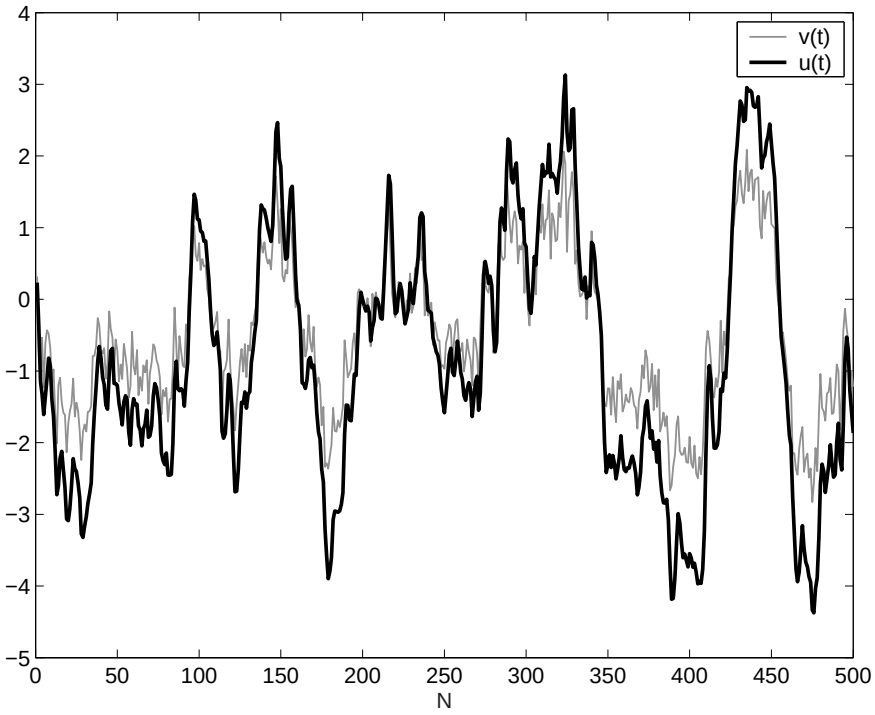


Figura 4.3 Efeito do filtro linear e invariante no tempo, especificado por meio de sua resposta ao impulso $\mathbf{h} = [1.0 \;\; 0.8 \;\; 0.5]^T$.

entrada, ou seja,

$$v(t) \;\; = \;\; x(t) + w(t), \qquad t = 1, 2, \ldots, N, \tag{4.3}$$

na qual $x(t)$ é o símbolo transmitido pelo canal, descrito acima, e $w \sim \mathcal{N}(0, \sigma_w^2)$ define as amostras de ruído branco gaussiano cuja variância é σ_w^2. Assume-se que a seqüência de dados $\{x(t)\}$ e a seqüência de ruído $\{w(t)\}$ são conjuntamente independentes.

O sinal $v(t)$ resultante é então processado por um filtro FIR linear por meio da seguinte expressão

$$u(t) = \frac{\mathbf{h}^T \mathbf{v}(t)}{\|\mathbf{h}\|}, \tag{4.4}$$

em que $\mathbf{v}(t) = [v(t) \; v(t-1) \; \cdots \; v(t-p+1)]^T$ é um vetor contendo os p símbolos ruidosos mais recentes e $\mathbf{h} \in \mathbb{R}^p$ é a resposta ao impulso do canal, considerada invariante no tempo. Nesta dissertação, adota-se $\mathbf{h} = [1.0 \;\; 0.8 \;\; 0.5]^T$, escolhida em razão de seu uso prévio em Gibson et al. (1989). O termo N representa o comprimento total da seqüência de símbolos gerada pelo modelo de canal não-linear adotado. O efeito resultante desta transformação linear sobre o sinal $v(t)$ é ilustrada na Figura 4.3.

Finalmente, uma seqüência de saída $y(t)$ não-linear é obtida ao aplicar uma transformação não-linear, misto de polinomial com sigmoidal, sobre sinal $u(t)$ (REBOLLO-

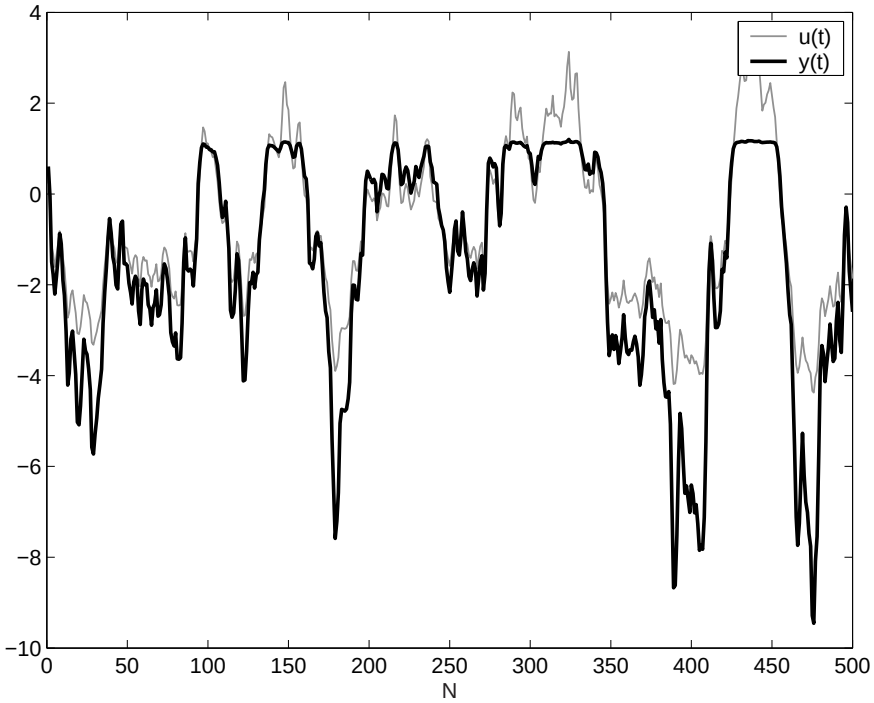


Figura 4.4 Efeito não-linear imposto pelo canal simulado.

MONEDERO, 2001):

$$y(t) = G_1\left[u(t)\right] + G_2\left[u(t)\right], \tag{4.5}$$

tal que

$$G_1\left[u(t)\right] = 0.2u(t) - 0.2u^2(t) + 0.04u^3(t), \qquad \text{e} \tag{4.6}$$

$$G_2\left[u(t)\right] = 0.9\tanh\left[u(t) - 0.1u^2(t) + 0.5u^3(t) - 0.1u^4(t) + 0.5\right]. \tag{4.7}$$

O efeito saturador da não-linearidade sigmoidal sobre o sinal $u(t)$ é claramente percebido a partir da Figura 4.4.

Verifica-se que o canal simulado introduz três efeitos à seqüência de entrada $\{x(t)\}$, conforme ilustrado na Figura 4.5.

- Ruído de medição através de w na Equação (4.3).
- Memória (correlação) linear através da resposta ao impulso $\mathbf{h}$.
- Saturação imposta pela não-linearidade sigmoidal em $G_2[u(t)]$.

É importante destacar que o canal ruidoso, com memória e não-linear definido acima não tem por objetivo simular a dinâmica de um canal de telecomunicações para uma tecnologia específica, seja com- ou sem-fio. Contudo, a motivação subjacente ao uso deste canal nas simulações a serem apresentadas neste e no próximo capítulos reside no fato

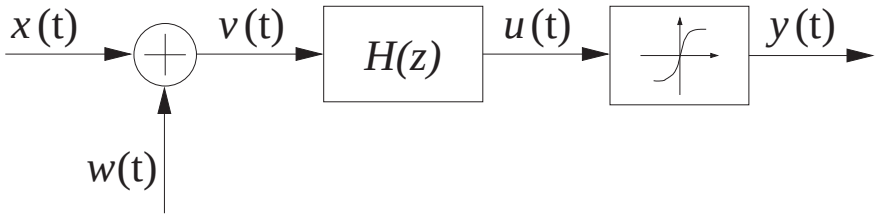


Figura 4.5 Canal ruidoso não-linear com memória.

de que ele condensa dois importantes aspectos comumente encontrados nos modernos sistemas de telecomunicações.

O primeiro destes aspectos diz respeito a canais de comunicação sem-fio, visto que o processo Gauss-Markov de primeira ordem é comumente utilizado para implementar modelos de desvanecimento Rayleigh plano ($flat$) (MISRA et al., 2003, 2004; DONG et al., 2003, 2004). O segundo aspecto diz respeito à não-linearidade imposta pela saturação dos amplificadores de potência e por conversores AD/DA.

No apêndice A, está disponibilizado o programa que foi usado para gerar o modelo do canal não-linear adotado nesta dissertação, apresentando cada passo para obtenção das seqüências de entrada e saída, $\{\mathbf{x}(t)\}$ e $\{y(t)\}$.

4.3 Simulações Computacionais

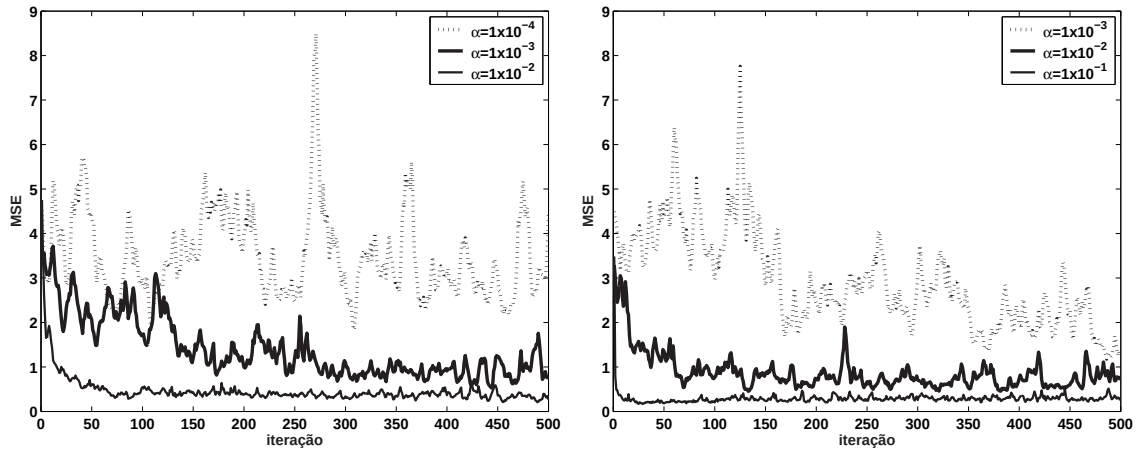
Para as simulações a seguir, foram adotados os seguintes valores para os parâmetros do canal: a=0,95, b=0,1, $\sigma_{\varepsilon}^2 = 1$, $\sigma_w^2 = 0{,}03$, $\mathbf{h} = [1 \quad 0.8 \quad 0.5]^T$. Seqüências de comprimento N=6.000 foram simuladas, tal que a porção inicial das seqüências, de comprimento ($5N/6 = 5.000$), foi usada para treinamento dos filtros, enquanto a porção restante ($N/6 = 1.000$) serviu para avaliar a capacidade de generalização (predição) dos mesmos. A dimensão do vetor de entrada de cada filtro, seja linear ou neural, foi escolhida por tentativa-e-erro como sendo $p = 5$.

Para avaliar o processo de aprendizagem dos filtros relacionados anteriormente, seja linear ou neural, é necessário o uso do *erro quadrático médio* (MSE) como medida de desempenho dos mesmos na fase de treinamento. Ele é definido como

$$MSE = \frac{1}{N}\sum_{t=1}^{N} e^2(t), \tag{4.8}$$

em que N é o comprimento total da seqüência de símbolos gerada pelo canal não-linear e $e(t)$ é o erro de estimação (ver Equação (1.4)) no instante de tempo t.

Para cada curva levantada, foram geradas 500 realizações da seqüência de entrada $\{x(t)\}$, e os valores finais de MSE (Equação (4.8)) resultam da média sobre estas 500



(a) (b)

Figura 4.6 Curvas de aprendizagem para o problema de identificação: (a) Filtro FIR/LMS e (b) Filtro FIR/LMS-Newton, para diferentes valores do passo de adaptação.

repetições.

4.3.1 Curvas de Aprendizagem

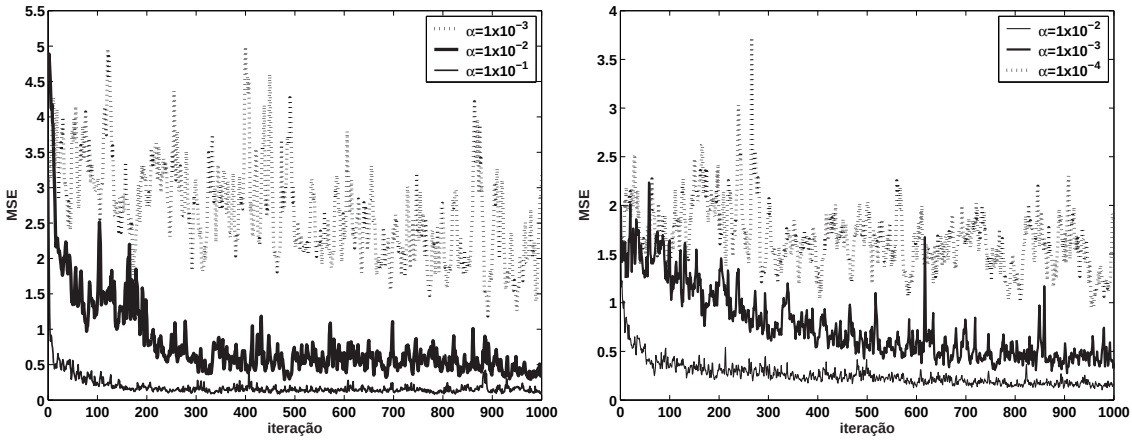
Filtros Lineares FIR/LMS e FIR/LMS-Newton - As curvas de aprendizagem para os filtros FIR/LMS e FIR/LMS-Newton quando utilizados para identificar o canal não-linear estão mostradas nas Figuras 4.6(a) e 4.6(b), respectivamente, para diferentes valores do passo de adaptação.

Os passos de adaptação escolhidos para construir as três curvas de aprendizagem para cada um dos filtros lineares foram os seguintes: $\alpha = 10^{-4}$, 10^{-3}, 10^{-2} e 10^{-1}. Estes valores foram escolhidos com base na seguinte restrição (HAYKIN, 1996; PRINCIPE et al., 2000; ABRANTES, 2000):

$$0 < \alpha < \frac{2}{\lambda_{max}} = \frac{2}{8,1666} \approx 0,24, \tag{4.9}$$

em que λ_{max} é o maior autovalor associado à matriz de auto-correlação dos vetores de entrada, para a dimensão escolhida (ou seja, $p = 5$). Para o filtro FIR/LMS-Newton, o valor do fator de esquecimento adotado é $\beta = 0,95$.

Como já era de se esperar, em virtude do exposto no Capítulo 1, o filtro FIR/LMS-Newton converge mais rapidamente para uma situação de regime permanente do que o filtro FIR/LMS. Além disso, o filtro FIR/LMS não convergiu para $\alpha \geq 10^{-2}$, mesmo satisfazendo a Equação (4.9). Isto ocorre porque, estritamente falando, esta restrição é valida apenas para processos lineares gaussianos e estacionários, servindo contudo para



(a) (b)

Figura 4.7 Curvas de aprendizagem para diferentes valores do passo de adaptação: (a) Filtro MLP e (b) Filtro LLM.

orientar a escolha da faixa de valores a serem também utilizados para os filtros lineares.

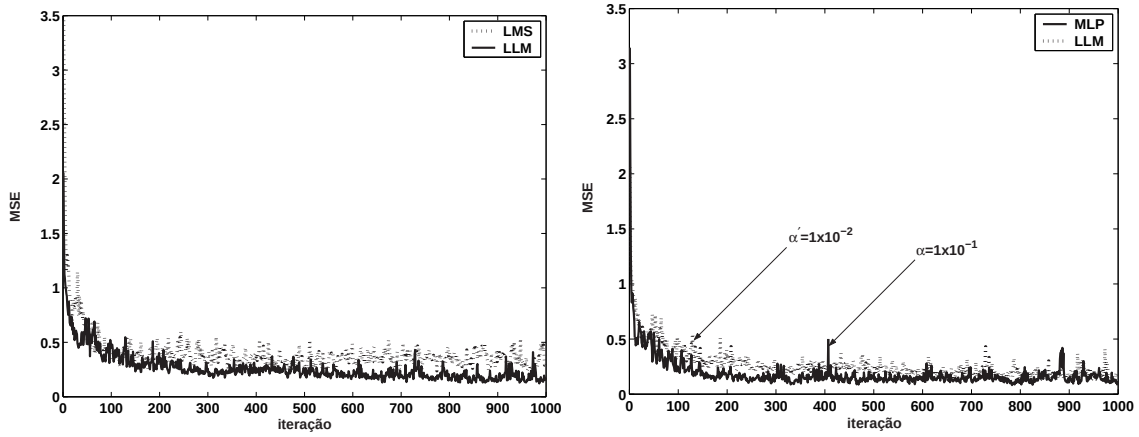
Filtros Adaptativos Neurais - Os filtros não-lineares são apresentados a seguir de acordo com o tipo de rede que foi adotada para realização da tarefa de identificação de canal. Primeiro, os resultados para o filtro MLP são apresentados e, em seguida, os resultados do filtros LLM e do VQTAM.

O desempenho de cada filtro neural é ilustrado através de um gráfico contendo três curvas de aprendizagem, que correspondem aos maiores valores permitidos para os passos de adaptação.

Filtro MLP - Para este filtro foi adotado $q = 10$ neurônios na camada escondida, todos com função de ativação tangente hiperbólica. Este número foi escolhido apenas por conveniência para se comparar os resultados dos modelos de filtros não-lineares. Contudo, será visto mais à frente que não haverá ganho de desempenho considerável para valores muito altos de q, devido à ocorrência de *over-fitting*, perceptível apenas durante a fase de generalização.

O único neurônio de saída tem ativação linear. Os pesos inicias de todos os neurônios foram escolhidos aleatoriamente e com distribuição uniforme na faixa de $\pm 0,01$. Os resultados obtidos da simulação estão mostrados na Figura 4.7(a).

Comparando o filtro MLP com o filtro FIR/LMS, pode-se perceber que aquele é mais robusto que este em relação aos valores permitidos para o passo de adaptação, mesmo usando regras de adaptação semelhantes. A explicação está no efeito estabilizador causado pela introdução da derivada da função de ativação não-linear na regra LMS, conforme mostrado na Equação (2.14) da regra Delta Generalizada. Este efeito estabilizador ficará



(a) (b)

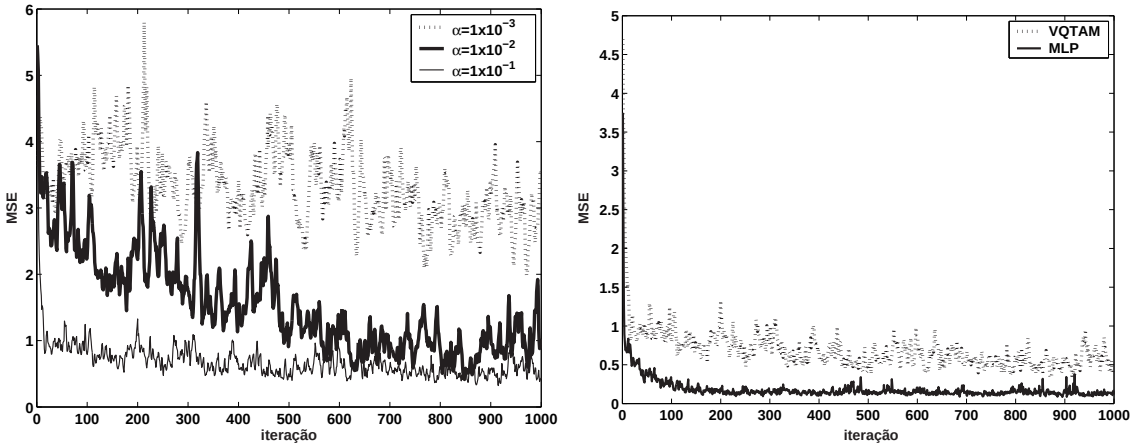
Figura 4.8 Comparação entre as curvas de aprendizagem: (a) LLM × FIR/LMS e (b) LLM × MLP, para α'=0,01 e α=0,1.

mais evidente ainda no próximo capítulo, quando for tratado o problema mais complexo da equalização de canais. Para a tarefa de identificação, o filtro MLP tem desempenho equivalente ao filtro FIR/LMS-Newton, sem precisar de informação extraída de momentos estatísticos de 2a. ordem, tal como a matriz de covariância.

Filtro LLM - As curvas de aprendizagem para este filtro, treinado com 10 neurônios, estão mostradas na Figura 4.7(b). Para estas simulações foram utilizadas os mesmos passos de adaptação utilizados pelo filtro FIR/LMS linear, ou seja, $\alpha' = 10^{-4}$, 10^{-3}, 10^{-2}, visto que o algoritmo LMS é parte integrante do filtro LLM. O passo de adaptação relativo à quantização vetorial do espaço de entrada (ver Equação (3.6)), implementado pela rede SOM, foi mantido constante e igual a α=0,1.

A título de comparação, a Figura 4.8(a) traz as curvas de aprendizagem dos filtros LLM e FIR/LMS, para os melhores casos de ambos (ou seja, α'=0,01 e α=0,01). Para esta figura, o filtro LLM é treinado com 10 neurônios e o passo de aprendizagem da quantização vetorial é α=0,1. Nota-se que o filtro LLM tem desempenho superior ao do filtro FIR/LMS, muito embora às expensas de um maior custo computacional. É feito também um comparativo entre a rede MLP e o LLM, visto na Figura 4.8(b). Nesta figura, observa-se que o desempenho do filtro LLM é equivalente ao do filtro MLP, com custo computacional também semelhante.

Filtro VQTAM - O filtro VQTAM foi treinado com 10 neurônios, para diferentes passos de adaptação, i.e. $\alpha = 10^{-4}, 10^{-3}, 10^{-2}$ e 10^{-1}. As curvas de aprendizagem resultantes estão mostradas na Figura 4.9(a). Somente, a curva de aprendizagem associada a $\alpha = 10^{-4}$ é omitida para não sobrecarregar a Figura 4.9(a).



(a) (b)

Figura 4.9 Curvas de aprendizagem: (a) Filtro VQTAM para vários passos de adaptação e (b) MLP × VQTAM, para α=0,1.

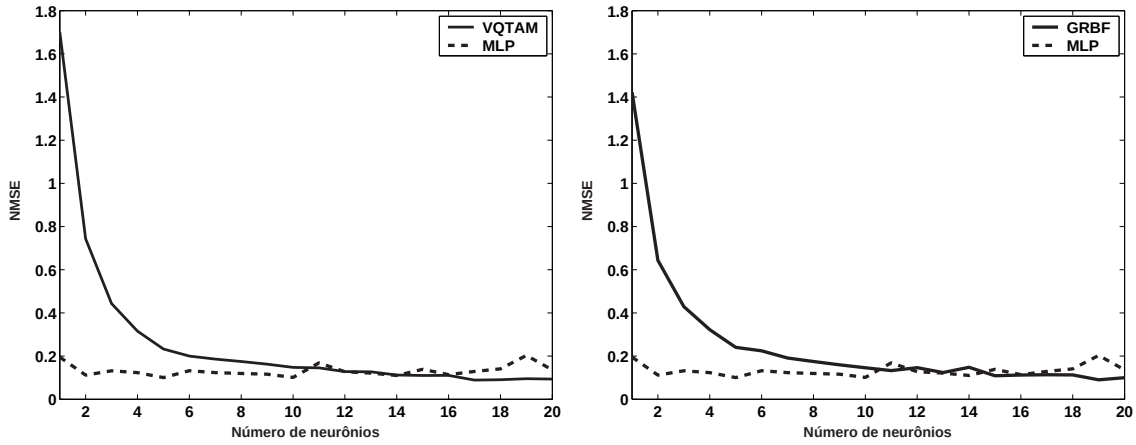
O desempenho nitidamente inferior do filtro VQTAM pode ser explicado pelo fato de se utilizar um número muito pequeno de neurônios. Para que este filtro forneça melhores resultados, deve-se utilizar muito mais neurônios, encarecendo seu custo computacional. Conforme será mostrado adiante, os modelos GRBF, KRBF e KSOM, todos construídos sobre o filtro VQTAM, têm boa capacidade de generalização em função dos métodos de interpolação por eles implementados.

Em compensação, o filtro VQTAM é o único que, ao lado do filtro MLP, se mostrou robusto às variações de α, ou seja, convergiu para todos os valores do passo de adaptação adotados. A Figura 4.9(b) mostra a comparação de desempenho para os filtros MLP e VQTAM, usando α=0,1.

4.4 Avaliação do Erro de Generalização

Nesta seção, os filtros neurais são avaliados segundo sua capacidade de generalização, ou seja, sua habilidade em fornecer (extrapolar) respostas coerentes após a etapa de treinamento, ou seja, sem ajuste dos pesos. Este tipo de análise de desempenho é importante porque serve para indicar se o filtro neural capturou corretamente a dinâmica não-linear do canal. Além disso, esta análise permite avaliar também o desempenho dos filtros baseados na rede VQTAM, tais como GRBF, KRBF e KSOM.

A avaliação do erro de generalização, feita com base no quadrático médio produzido, serve também como uma medida qualitativa da sensibilidade de cada filtro em relação a variações em alguns de seus parâmetros mais importantes, em particular, o número de



(a) (b)

Figura 4.10 Erro de generalização dos filtros em função do número de neurônios: (a) MLP × VQTAM e (b) MLP × GRBF.

neurônios (q) das redes MLP, VQTAM e GRBF e o número de vencedores (K) das redes KRBF e KSOM.

O erro de generalização é a média total do erro quadrático médio normalizado pela variância do sinal de saída, a cada etapa de realização, que no caso da identificação é o valor correspondente à variância da seqüência de símbolos $\{y(t)\}$. O valor do *erro quadrático médio normalizado* (NMSE) é calculado da seguinte forma

$$NMSE = \sqrt{\frac{\sum_{t=1}^{N} e^2(t)}{N \cdot \sigma_y^2}} = \sqrt{\frac{\hat{\sigma}_e^2}{\sigma_y^2}}, \tag{4.10}$$

em que σ_y^2 é a variância do sinal de saída, $\hat{\sigma}_e^2$ é a variância dos erros e N é o tamanho da seqüência de erros.

O primeiro resultado interessante obtido a partir da avaliação do erro de generalização é mostrado na Figura 4.10(a). Nesta figura, o filtro MLP é comparado com o filtro VQTAM. Para cada valor de q, variando de 1 até 20, cada filtro é treinado 500 vezes, para 500 realizações distintas das seqüências de entrada e saída. Pode-se notar que, à medida que o número de neurônios cresce, o desempenho do filtro VQTAM melhora, a ponto de superar o filtro MLP a partir de $q = 12$. O filtro MLP, por outro lado, atinge seu melhor desempenho para $q = 5$. Deste valor em diante o erro de generalização tem uma tendência de crescimento.

Este comportamento do filtro MLP confirma um resultado bastante conhecido pela comunidade científica, conhecido como *over-fitting*, que afirma que um aumento no número de neurônios na camada escondida não implica necessariamente em maior capacidade de

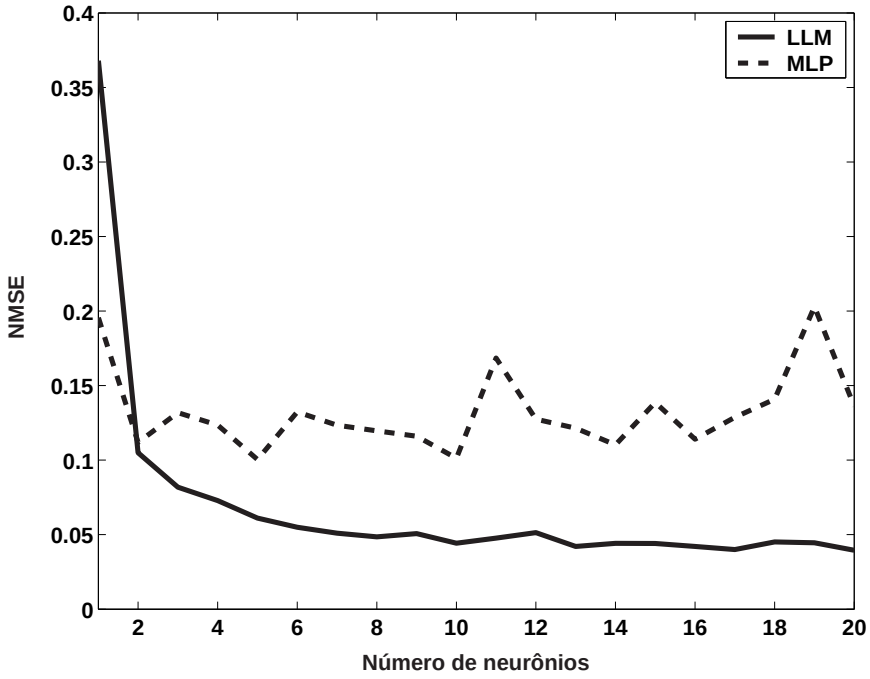


Figura 4.11 Erro de generalização dos filtros MLP e LLM em função do número de neurônios.

generalização (PRINCIPE et al., 2000; HAYKIN, 1994). Já para o filtro VQTAM, por ser baseado num quantizador vetorial, a existência de muitos neurônios significa que mais vetores-protótipos estão povoando os espaços de entrada e de saída, reduzindo assim o erro de quantização que, neste filtro, se confunde com o erro de generalização.

O mesmo tipo de comparação foi levada a cabo para os filtros MLP e GRBF, com os resultados sendo mostrados na Figura 4.10(b). Verifica-se que, para a tarefa de identificação de canais, não houve maiores diferenças entre o comportamento do filtro GRBF e do filtro VQTAM, sendo que os dois filtros inicialmente superam o MLP a partir de $q = 11$ e $q = 12$ neurônios, respectivamente.

Outro resultado é visto na Figura 4.11, na qual são comparadas as simulações dos filtros LLM e MLP para o erro de generalização (NMSE) em função do número de neurônios. O filtro LLM apresentou resultados melhores do que o MLP a partir do valor $q = 2$, configurando uma diminuição rápida do NMSE. Este resultado demonstra que o processo de quantização vetorial feito pelo algoritmo LLM no espaço de entrada auxiliou na obtenção de bons resultados na estimação da saída, sendo melhores do que a rede MLP, que apresentou um crescimento no valor do NMSE devido ao processo de *over-fitting* que esta rede sofre com o aumento de q.

A última bateria de simulações deste capítulo visa avaliar como o erro de generalização para os filtros KRBF e KSOM varia em função de K. Os resultados estão mostrados na Figura 4.12. Para obter as curvas mostradas nesta figura, o número de neurônios usado

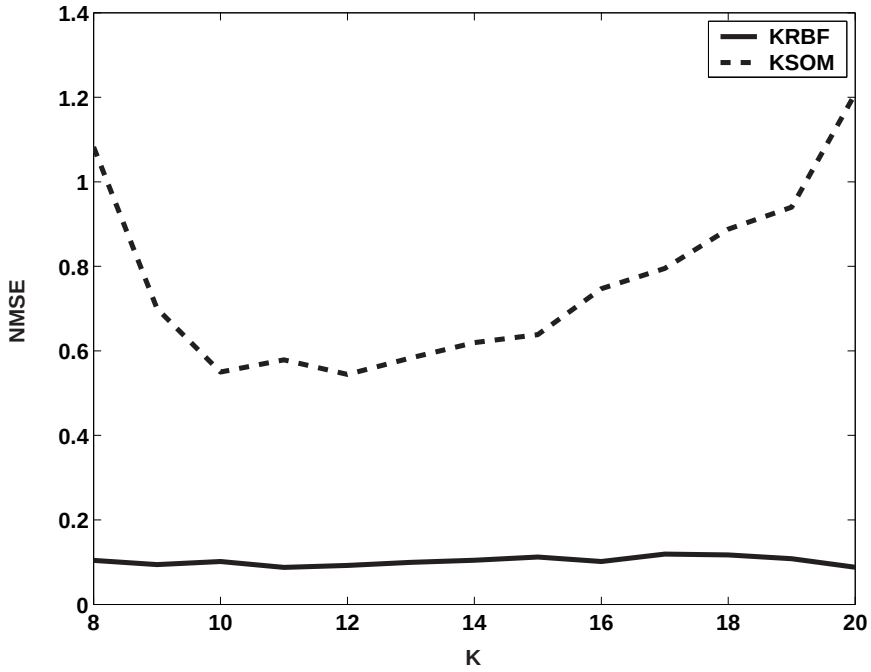


Figura 4.12 Erro de generalização para os filtros KSOM e KRBF em função de K.

pelo filtro VQTAM é fixado em 20 neurônios, enquanto o valor de K é variado de 8 até 20. Para cada valor de K, 500 rodadas de treinamento são então realizadas e o erro de generalização médio é calculado.

Analisando a Figura 4.12, observa-se que o erro de generalização do filtro KSOM se mantém sempre maior que o erro do filtro KRBF. Além disso, o filtro KSOM é mais sensível (menos robusto) à variação de K, comportamento este demonstrado pela forma de U da curva do seu erro de generalização. Em outras palavras, quando comparada com a curva do filtro KSOM, a curva do erro de generalização do filtro KRBF é praticamente constante. O valor mínimo de K para o filtro KRBF foi estipulado em $K = 11$, enquanto que para o filtro KSOM foi $K = 12$. A explicação para este comportamento do filtro KSOM é que a partir do valor mínimo de K, com o aumento do número de neurônios vencedores a serem escolhidos, o algoritmo além de definir os índices dos neurônios mais próximos, ele acrescenta ao conjunto índices de neurônios que se encontram mais distantes do vetor de entrada atual, montando conseqüentemente hiperplanos que não se aproximam bem ao valor da saída desejada. Este erro pode ser verificado quando é feito o cálculo da *pseudo-inversa* para obtenção do vetor de coeficientes a ser utilizado no processo de interpolação linear da saída desejada.

Tomando por base a simulação anterior, pode-se montar uma tabela comparativa contendo os valores obtidos para os erros de generalização de todos os filtros neurais, fixando o número de neurônios em $q = 20$, e também, dos valores obtidos para os dois filtros lineares estudados anteriormente (ver Tabela 4.1). Os resultados ali apresentados permitem

Tabela 4.1 Erro de generalização (NMSE) dos filtros lineares e não-lineares para o problema de identificação de canais.

Filtros	**NMSE**			
	média	*mínimo*	*máximo*	*variância*
FIR/LMS	0,3214	0,0828	1,4121	0,0260
FIR/LMS-Newton	0,3278	0,3278	1.1698	0,0290
MLP ($q = 20$)	0,1170	0,0161	0,7364	0,0145
LLM ($q = 20$)	0,0396	0,0124	0,2695	$7{,}67\times10^{-4}$
VQTAM ($q = 20$)	0,0999	0,0241	0,9031	0.0105
GRBF ($q = 20$)	0,0998	0,0152	1,5579	0,0278
KRBF ($K = 11$)	0,0877	0,0112	1,0833	0,0145
KSOM ($K = 12$)	0,5295	0,0881	1,4649	2,9726

concluir que todos os algoritmos baseados na rede SOM têm desempenho melhor do que o filtro MLP convencional na tarefa de identificação de canais não-lineares. Especificamente, é importante destacar nessa tabela o desempenho superior do filtro KRBF, comparado com os modelos que adotam a estrutura VQTAM para construção do quantizador vetorial.

4.5 Conclusão

Este capítulo avaliou o desempenho de filtros lineares e neurais na tarefa de identificação de canais não-lineares. Tais filtros foram avaliados segundo sua velocidade de aprendizado (convergência) e sua capacidade de generalização (extrapolação). Em particular, buscou-se comparar o desempenho do filtro neural clássico, baseado na rede MLP, com aqueles baseados na rede SOM (GRBF, KRBF e KSOM).

Como conclusão geral, pode-se afirmar que, apesar de ter desempenho apenas razoável durante a fase de treinamento usando um número pequeno de neurônios, o filtro VQTAM e os filtros KRBF e GRBF superam o filtro MLP na fase de generalização para um número ligeiramente maior de neurônios. A desvantagem está no aumento do custo computacional resultante. Porém, no caso do filtro VQTAM e seus derivados, há a garantia de que o desempenho tende a melhorar com o aumento do número de neurônios.

É importante ressaltar também que valores ótimos para alguns dos parâmetros dos filtros neurais, tais como o número de neurônios (q) e o número de vencedores (K) foram encontrados empiricamente através de extensas simulações. Um dos futuros tópicos a serem explorados como desdobramentos desta dissertação consiste em estudar mecanismos automáticos de determinação do número ótimo de neurônios para o filtro VQTAM. Uma possível estratégia neste sentido seria lançar mão de redes competitivas do tipo crescentes (*growing*) (FRITZKE, 1994), que adicionam neurônios na rede em função de alguma mé-

trica de avaliação. Mais detalhes sobre perspectivas de trabalhos serão apresentados no Capítulo 6.

O próximo capítulo avalia os mesmos filtros na tarefa de modelagem inversa do canal de comunicação (equalização), tarefa esta mais complexa que a de identificação de canais.

5 EQUALIZAÇÃO DE CANAIS USANDO REDES NEURAIS

5.1 Introdução

Neste capítulo, são apresentados os resultados obtidos a partir da aplicação dos filtros lineares e neurais ao problema de equalização de canais de comunicação. Por se tratar de um *problema inverso*, sabe-se que esta tarefa apresenta um grau de dificuldade maior em relação ao problema de identificação de canais e, por isso, impõe maiores desafios aos filtros.

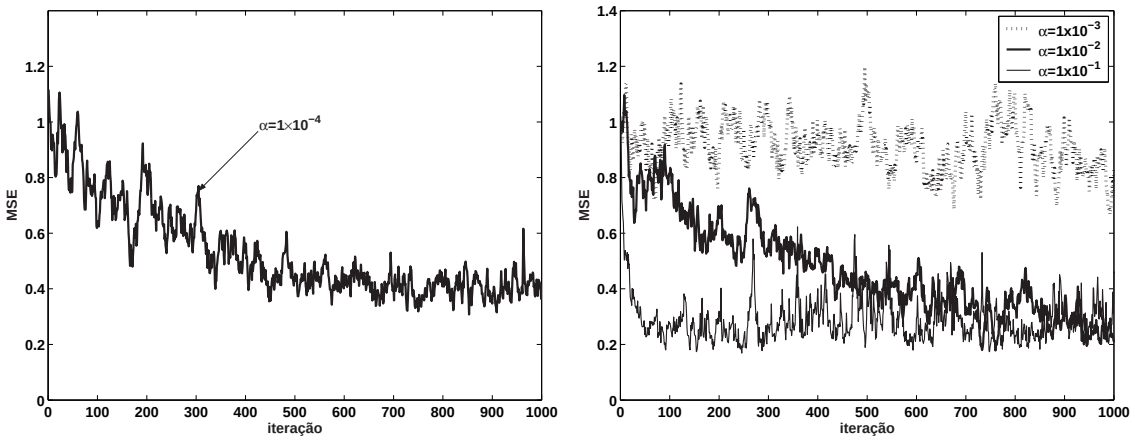
Seguindo a mesma metodologia de avaliação do capítulo anterior, todos os filtros em questão serão avaliados conforme a sua velocidade de convergência e a sua sensibilidade (ou dependência) em relação a variação de algum parâmetro de treinamento, tal como o número de neurônios utilizados.

Ao longo da apresentação dos resultados, diversos pontos de discussão serão abordados com respeito aos modelos propostos, para então se chegar às conclusões finais sobre o emprego dos mesmos.

5.2 Simulações Computacionais

Os filtros lineares e as redes neurais são avaliados conforme a velocidade do seu aprendizado, ou seja, como a convergência se comporta através de uma queda lenta ou abrupta dos valores do erro médio quadrático visto na curva de aprendizagem do algoritmo no decorrer do tempo. Cada algoritmo apresenta de uma forma singular o cálculo do erro que será usado na análise da convergência.

Os parâmetros gerais, tais como o tamanho da seqüência de treinamento e teste, e o tamanho do vetor de entrada $\mathbf{x}(t)$ são os mesmos do caso da identificação de canais. Para análise dos casos, segue-se o mesmo estilo apresentado no Capítulo 4, começando a falar sobre os resultados dos filtros lineares e, em seguida, comentando sobre as redes neurais.



(a) (b)

Figura 5.1 Curvas de aprendizagem: (a) Filtro FIR/LMS e (b) Filtro FIR/LMS-Newton, para diferentes valores do passo de adaptação.

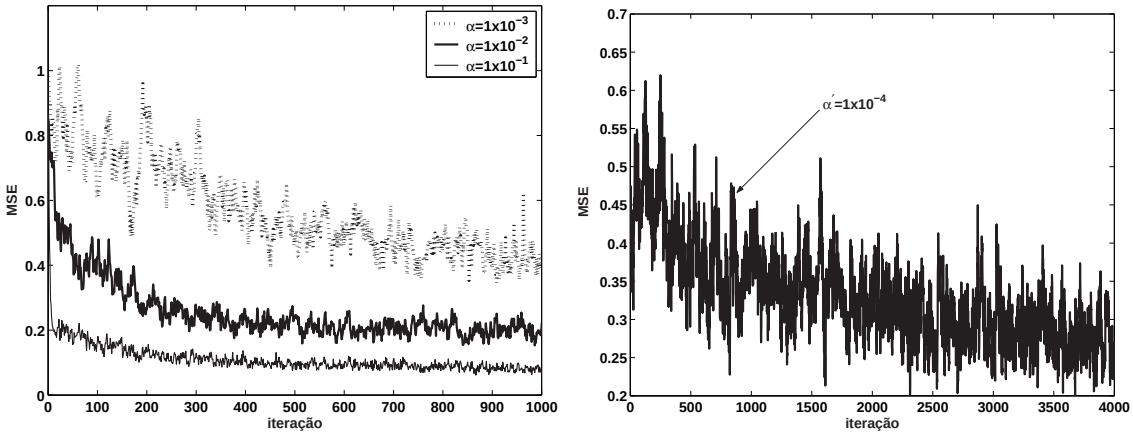
5.2.1 Curvas de Aprendizagem

O mesmo canal não-linear apresentado no capítulo anterior é simulado aqui, mantendo-se também os mesmos parâmetros de simulação. A título de completude, estes parâmetros e os demais itens que configuram a simulação do canal e o treinamento dos filtros são apresentados novamente a seguir.

Os parâmetros do canal não-linear são os seguintes: a=0,95, b=0,1, $\sigma_{\varepsilon}^2 = 1$, $\sigma_w^2 = 0{,}03$, $\mathbf{h} = [1 \quad 0{,}8 \quad 0{,}5]^T$. , Seqüências de comprimento N=6.000 foram simuladas, tal que a porção inicial das seqüências, de comprimento ($5N/6 = 5.000$), foi usada para treinamento dos filtros, enquanto a porção restante ($N/6 = 1.000$) serviu para avaliar a capacidade de generalização (predição) dos mesmos. A dimensão do vetor de entrada de cada filtro, seja linear ou neural, foi escolhida por tentativa-e-erro como sendo $p = 5$.

Para cada curva levantada, foram geradas 500 realizações da seqüência de entrada $\{x(t)\}$, e os valores finais de MSE resultam da média sobre estas 500 repetições.

Filtros Lineares FIR/LMS e FIR/LMS-Newton - Apesar de os filtros FIR/LMS e FIR/LMS-Newton terem sido originalmente propostos para lidar com processos lineares, gaussianos e estacionários, nada impede que seus desempenhos também sejam avaliados para o presente caso. As curvas de aprendizagem para estes filtros, quando utilizados para equalizar o canal não-linear, estão mostradas nas Figuras 5.1(a) e 5.1(b), respectivamente, para diferentes valores do passo de adaptação (α=10^{-1}, 10^{-2}, 10^{-3} e 10^{-4}). Para o filtro FIR/LMS-Newton, foi adotado o mesmo valor do fator de esquecimento utilizado no problema de identificação de canais, ou seja, $\beta = 0,95$.



(a) (b)

Figura 5.2 Curvas de aprendizagem para diferentes valores do passo de adaptação: (a) Filtro MLP e (b) Filtro LLM.

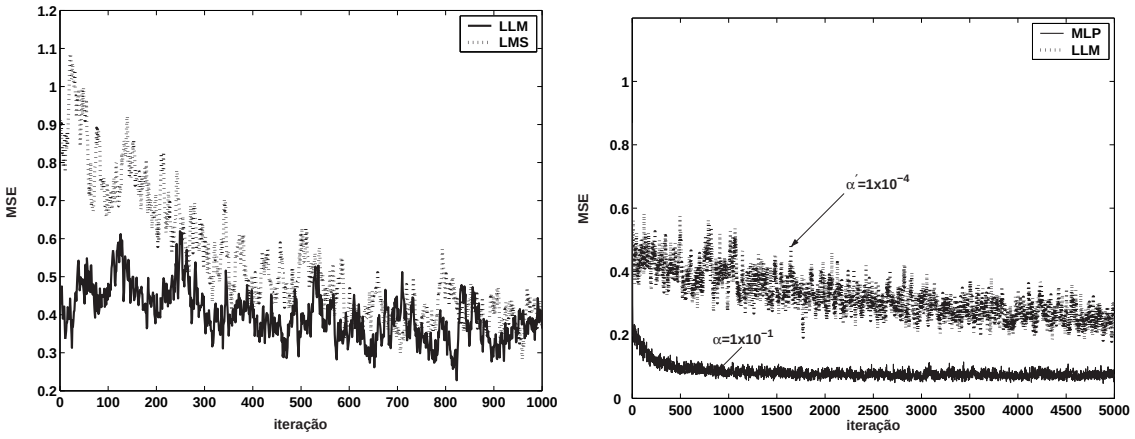
Pode-se perceber que o filtro FIR/LMS só convergiu para $\alpha = 10^{-4}$, mesmo assim muito lentamente, estabilizando em um valor alto ($\sim$0,4). Já o filtro FIR/LMS-Newton consegue convergir para os três menores valores atribuídos a α, mesmo assim estabilizando em patamares elevados do erro. Do exposto, conclui-se que o problema de equalização torna mais evidente a inadequação dos filtros FIR/LMS e FIR/LMS-Newton à modelagem de sistemas não-lineares.

Filtros Adaptativos Neurais - Os filtros não-lineares são apresentados a seguir de acordo com o tipo de rede que foi adotada para realização da tarefa de equalização de canal. Primeiro, os resultados para o filtro MLP são apresentados e, em seguida, os resultados do filtros LLM e do VQTAM.

O desempenho de cada filtro neural é ilustrado através de um gráfico contendo três curvas de aprendizagem, que correspondem aos maiores valores permitidos para os passos de adaptação.

Filtro MLP - Para este filtro foi adotado $q = 10$ neurônios na camada escondida, todos com função de ativação tangente hiperbólica. Este número foi escolhido apenas por conveniência para se comparar os resultados dos modelos de filtros não-lineares, visto que quanto mais neurônios nesta camada menor será o erro durante o treinamento. Contudo, será visto mais à frente que não haverá ganho de desempenho considerável para valores muito altos de q, devido à ocorrência de *over-fitting*, perceptível apenas durante a fase de generalização.

O único neurônio de saída tem função de ativação linear. Os pesos inicias de todos os neurônios foram escolhidos aleatoriamente e com distribuição uniforme na faixa de $\pm 0,01$.



(a) (b)

Figura 5.3 Comparação entre as curvas de aprendizagem: (a) LLM × FIR/LMS e (b) LLM × MLP, para α'=0,0001 e α=0,1.

Os resultados obtidos da simulação estão mostrados na Figura 5.2(a).

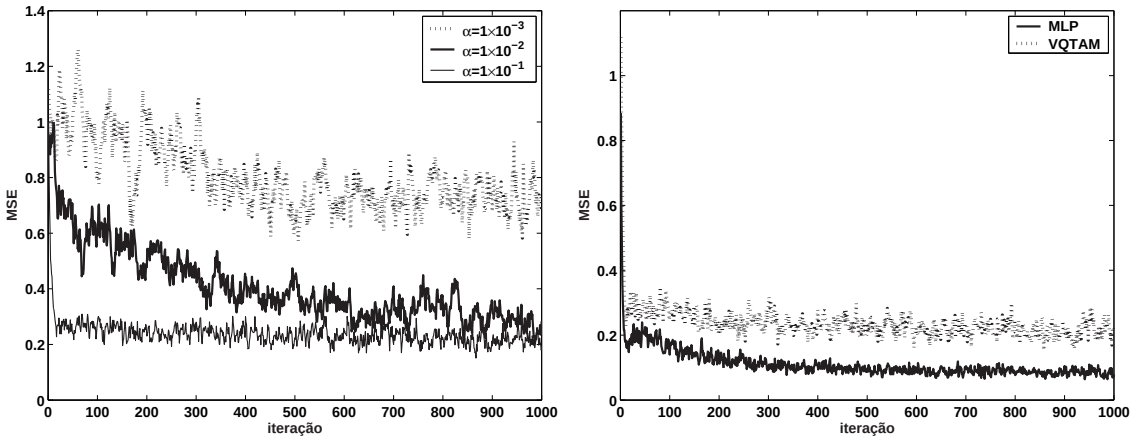
De modo similar ao ocorrido no Capítulo 4, o filtro MLP mostrou-se devidamente mais robusto que os filtros lineares, permitindo utilizar valores mais altos para o passo de adaptação, acelerando o processo de convergência. O erro também estabiliza em patamares bem inferiores àqueles obtidos para o filtro FIR/LMS-Newton.

A justificativa para este comportamento é a mesma, ou seja, a não-linearidade da função de ativação dos neurônios do filtro agrega maior poder computacional e estabilidade ao processo de aprendizagem, em função do efeito estabilizador causado pela introdução da derivada da função de ativação na regra LMS, dando margem ao aparecimento da regra Delta Generalizada, descrita na Equação (2.14).

Filtro LLM - A única curva de aprendizagem para este filtro, treinado com 10 neurônios, está mostrada na Figura 5.2(b). Isto ocorre porque o filtro LLM é baseado no algoritmo de adaptação LMS, que se mostrou instável para o problema de equalização. Assim, o único valor permitido para o passo de adaptação foi $\alpha' = 10^{-4}$ e $\alpha = 10^{-4}$. O passo de adaptação relativo à quantização vetorial do espaço de entrada (ver Equação (3.6)), implementada pela rede SOM, foi mantido constante e igual a α=0,1.

A título de comparação, a Figura 5.3(a) traz as curvas de aprendizagem dos filtros LLM e FIR/LMS, para $\alpha' = 10^{-4}$. Para esta figura, o filtro LLM continuou sendo treinado com 10 neurônios e passo de aprendizagem da quantização vetorial igual a α=0,1. Nota-se que a única diferença está no fato de o filtro LLM convergir mais rápido que o filtro LMS, muito embora ambos convirjam para o mesmo patamar de erro.

É feito também um comparativo entre a rede MLP e o LLM, visto na Figura 5.3(b).



(a) (b)

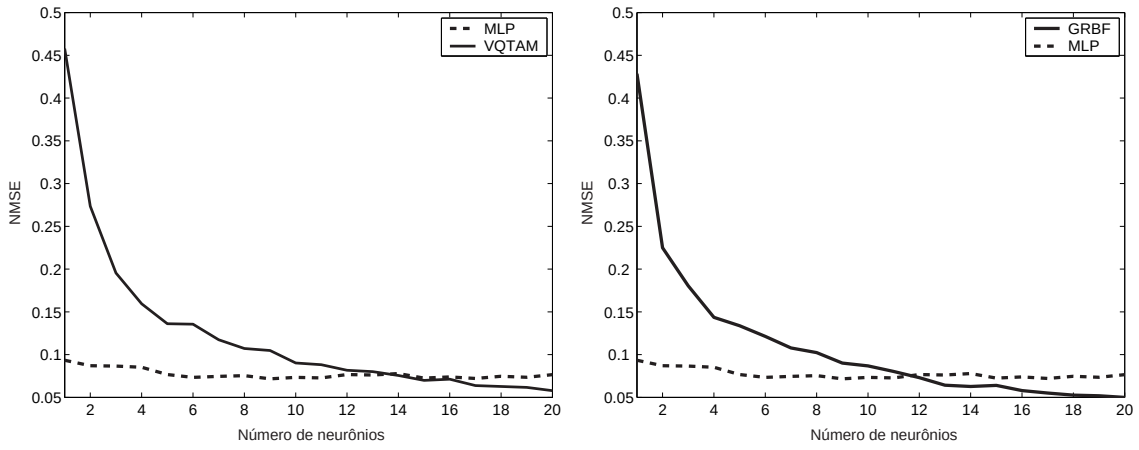
Figura 5.4 Curvas de aprendizagem: (a) Filtro VQTAM para vários passos de adaptação e (b) MLP × VQTAM, para α=0,1.

Nesta figura, observa-se que o desempenho do filtro LLM é bem inferior ao do filtro MLP, tanto em velocidade de convergência quanto no patamar final atingido para o erro.

Filtro VQTAM - O filtro VQTAM foi treinado com 10 neurônios, para diferentes passos de adaptação, i. e. $\alpha = 10^{-4}$, 10^{-3}, 10^{-2} e 10^{-1}. As curvas de aprendizagem resultantes estão mostradas na Figura 5.4(a). Somente a curva de aprendizagem associada a $\alpha = 10^{-4}$ é omitida para não sobrecarregar a figura.

O filtro VQTAM é o único que, ao lado do filtro MLP, se mostrou robusto às variações de α, ou seja, convergiu para todos os valores do passo de adaptação adotados. A Figura 5.4(b) mostra a comparação de desempenho entre os filtros MLP e VQTAM, usando α=0,1.

Embora tenham praticamente a mesma velocidade de convergência, o valor final do erro para o filtro MLP é menor do que o valor para o filtro VQTAM. Isto pode ser explicado pelo fato de se ter usado um número muito pequeno de neurônios na simulação de ambos os filtros. Para que o filtro VQTAM forneça melhores resultados deve-se utilizar muito mais neurônios, encarecendo seu custo computacional. Conforme será mostrado adiante, os modelos GRBF, KRBF e KSOM, todos construídos sobre o filtro VQTAM, têm boa capacidade de generalização em função dos métodos de interpolação por eles implementados.



(a) (b)

Figura 5.5 Erro de generalização dos filtros em função do número de neurônios: (a) MLP × VQTAM e (b) MLP × GRBF.

5.3 Avaliação do Erro de Generalização

Nesta seção, os filtros neurais são avaliados segundo sua capacidade de generalização, ou seja, sua habilidade em fornecer respostas coerentes com o problema de equalização após a etapa de treinamento. Serão apresentados os desempenhos dos modelos propostos por este trabalho e dos demais modelos que já são utilizados como aproximadores de funções na literatura científica.

Conforme mencionado no capítulo anterior, este tipo de análise de desempenho é importante porque serve para indicar se o filtro neural capturou corretamente a dinâmica inversa do canal. Além disso, esta análise permite avaliar também o desempenho dos filtros baseados na rede VQTAM, tais como GRBF, KRBF e KSOM.

A avaliação do erro de generalização, feita com base no quadrático médio produzido, serve também como uma medida qualitativa da sensibilidade de cada filtro em relação a variações em alguns de seus parâmetros mais importantes, em particular o número de neurônios (q) das redes MLP, VQTAM e GRBF e o número de vencedores (K) das redes KRBF e KSOM.

O primeiro resultado interessante obtido a partir da avaliação do erro de generalização é mostrado na Figura 5.5(a). Nesta figura, o filtro MLP é comparado com o filtro VQTAM. Para cada valor de q, variando de 1 até 20, cada filtro é treinado 500 vezes, para 500 realizações distintas das seqüências de entrada e saída. Assim como verificado para o problema de identificação de canais, pode-se notar que, à medida que o número de neurônios cresce, o desempenho do filtro VQTAM melhora, a ponto de superar o filtro MLP a

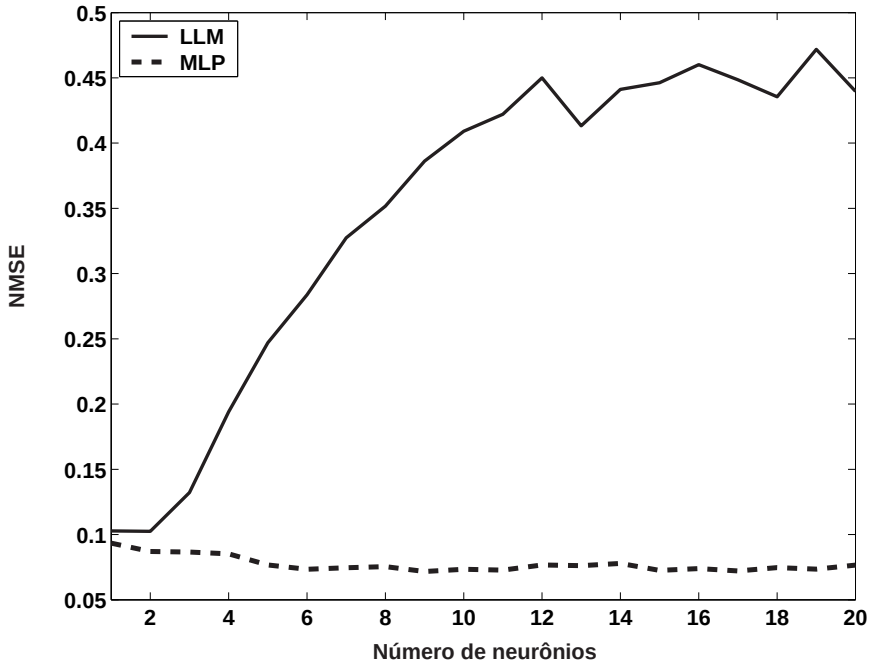


Figura 5.6 Erro de generalização dos filtros MLP e LLM em função do número de neurônios.

partir de $q = 14$. O filtro MLP, por outro lado, atinge seu melhor desempenho para $q = 6$. Deste valor em diante o erro de generalização tem uma tendência de crescimento, devido à ocorrência de *over-fitting*. Já para o filtro VQTAM, por ser baseado num quantizador vetorial, a existência de muitos neurônios significa que mais vetores-protótipos estão povoando os espaços de entrada e de saída, reduzindo assim o erro de generalização.

O mesmo tipo de comparação foi feita para os filtros MLP e GRBF, conforme mostrado na Figura 5.5(b). Verifica-se que, para a tarefa de equalização de canais, o filtro GRBF supera o filtro MLP já a partir de $q = 12$, com menos neurônios que o exigido para o filtro VQTAM superar o MLP. Isto demonstra que a interpolação promovida pelo filtro GRBF é eficiente.

Outro resultado foi obtido com as simulações dos filtros LLM e MLP, e é visto na Figura 5.6. A análise feita com base nestas simulações demonstra que o filtro LLM apresentou um comportamento diferente daquilo que era esperado, com base no seu resultado obtido anteriormente para o problema de identificação de canais. Isto foge do compromisso visto inicialmente na Figura 4.11, em que a quantização vetorial do sinal de entrada auxiliara na escolha do melhor vetor de coeficientes a ser usado na interpolação linear da saída desejada.

As últimas simulações deste capítulo visam avaliar como o erro de generalização para os filtros KRBF e KSOM varia em função de K. Os resultados estão mostrados na Figura 5.7. Para obter as curvas mostradas nesta figura, o número de neurônios usado pelo filtro VQTAM é fixado em 20 neurônios, enquanto o valor de K é variado de 8 até 20.

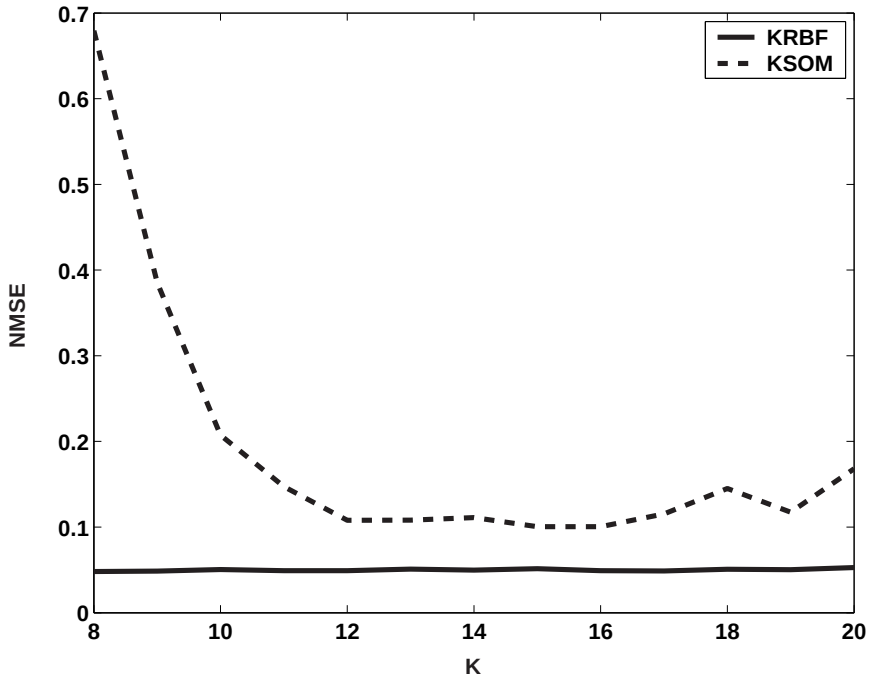


Figura 5.7 Erro de generalização para os filtros KSOM e KRBF em função de K.

Para cada valor de K, 500 rodadas de treinamento são realizadas e o erro de generalização médio é calculado.

Analisando a Figura 5.7, observa-se que o erro de generalização do filtro KSOM se mantém sempre maior que o erro do filtro KRBF. Além disso, o filtro KSOM é bem mais sensível (menos robusto!) à variação de K, comportamento este demonstrado pela forma de U da curva do seu erro de generalização. Em outras palavras, quando comparada com a curva do filtro KSOM, a curva do erro de generalização do filtro KRBF é praticamente constante. O valor mínimo de K para o filtro KRBF foi estipulado em $K = 8$, enquanto que para o filtro KSOM foi $K = 15$. Nesta simulação, pôde-se ver o mesmo comportamento da rede que no caso da identificação de canais, onde o algoritmo KSOM apresentou resultados piores com o aumento do valor de K, ou seja, este algoritmo sofreu as mesmas limitações em termos da escolha dos K neurônios vencedores para calcular os vetores de coeficientes que auxiliaram na interpolação linear da saída desejada.

Tomando por base a simulação anterior, pode-se montar uma tabela comparativa contendo os valores obtidos para os erros de generalização de todos os filtros neurais, fixando o número de neurônios em $q = 20$, e também, dos valores obtidos para os dois filtros lineares estudados anteriormente (ver Tabela 5.1). Os resultados apresentados nesta tabela permitem concluir que todos os algoritmos baseados na rede SOM têm desempenho melhor do que o filtro MLP convencional na tarefa de equalização de canais não-lineares. É importante destacar nessa tabela o desempenho superior do filtro KRBF.

Tabela 5.1 Erro de generalização (NMSE) dos filtros lineares e não-lineares para o problema de equalização de canais.

Filtros	**NMSE**			
	média	*mínimo*	*máximo*	*variância*
FIR/LMS	0,0925	0,0324	0,3237	0,0011
FIR/LMS-Newton	0,1460	0,0229	1,8880	0,0337
MLP ($q = 20$)	0,0785	0,0185	1,3479	0,0065
LLM ($q = 20$)	0,1360	0,0438	0,3537	0,0026
VQTAM ($q = 20$)	0,0583	0,0265	0,1428	$3{,}33\times10^{-4}$
GRBF ($q = 20$)	0,0500	0,0177	0,2198	$6{,}79\times10^{-4}$
KRBF ($K = 8$)	0,0487	0,0190	0,1723	$4{,}67\times10^{-4}$
KSOM ($K = 15$)	0,0991	0,0251	0,7721	0,0076

5.4 Conclusão

Este capítulo avaliou o desempenho de filtros lineares e neurais na tarefa de equalização de canais não-lineares. Tais filtros foram avaliados segundo sua velocidade de convergência e sua capacidade de generalização. Em particular, buscou-se comparar o desempenho dos filtros neural clássico, baseado na rede MLP, com aqueles baseados na rede SOM (GRBF, KRBF e KSOM).

Como conclusão geral, pode-se afirmar que o problema de equalização adaptativa serviu para enfatizar os resultados gerais obtidos para o problema de identificação de canais, ou seja, que os filtros baseados na rede SOM são alternativas viáveis ao filtro MLP e RBF tradicionais. Os filtros lineares tiveram desempenho abaixo da média para o problema de equalização aqui estudado.

Para o problema de equalização em particular, o filtro VQTAM teve melhor desempenho que os filtros FIR/LMS, FIR/LMS-Newton e LLM durante a fase de treinamento, apesar de não ter superado o filtro MLP. Por outro lado, os filtros VQTAM, GRBF e KRBF superam o filtro MLP na fase de generalização para um número ligeiramente maior de neurônios. A desvantagem está no aumento do custo computacional resultante. Porém, no caso do filtro VQTAM e derivados, há a garantia de que o desempenho tende a melhorar.

O próximo capítulo traz um resumo dos tópicos abordados e dos principais resultados e conclusões obtidas. Discute-se também alguns possíveis temas para futuros trabalhos na área.

6 CONCLUSÕES E PERSPECTIVAS

Os algoritmos apresentados neste trabalho foram avaliados conforme as curvas de aprendizagem e do erro de generalização. Iniciando com análise dos filtros lineares, pôde-se ver que a aplicação dos mesmos neste tipo de canal não-linear foi limitada, com base na convergência dos algoritmos em relação aos valores adotados para as taxas de aprendizagem. Os resultados obtidos para o erro de generalização estiveram acima dos valores correspondentes para os algoritmos propostos por este trabalho.

Os resultados obtidos para os algoritmos clássicos de aproximação de funções usando a rede SOM, tais como LLM e VQTAM, demonstraram bem o emprego da técnica de quantização vetorial da informação de entrada e saída, sendo que no caso do filtro LLM a sua capacidade de realizar somente a quantização da informação da entrada levou-o a resultados satisfatórios no problema de identificação de canais, mas obteve resultados não muito satisfatórios na tarefa mais difícil que é a equalização de canais, ofuscando os resultados obtidos no primeiro problema. Passando para o filtro VQTAM, pôde-se ver resultados satisfatórios nos dois casos, e verificou-se nos gráficos do erro de generalização uma diminuição no valor do NMSE com o aumento no número de neurônios q, demonstrando assim a evolução dada à quantização vetorial da informação de entrada/saída feito pelo mesmo filtro. Esta característica do filtro VQTAM vai de encontro com o filtro MLP, já que este demonstrou um crescimento do valor do NMSE em relação a q, explicitando bem a presença de "*over-fitting*" devido ao crescimento no número de neurônios utilizados pelo filtro. Para valores menores do número de neurônios, há uma redução no esforço computacional e, nesta situação, o filtro MLP se apresenta como a melhor opção pois ele demonstra uma diminuição no valor do NMSE, tendo um rendimento superior ao do filtro VQTAM.

Passando para os modelos propostos, têm-se os filtros que realizam interpolação não-linear através da função gaussiana e que adotam a técnica VQTAM. Pôde-se ver que os filtros GRBF e KRBF apresentaram resultados satisfatórios nos dois problemas. Os filtros VQTAM e GRBF são modelos particulares do filtro KRBF para o valor de $K = 1$ e $K = q$, respectivamente. Os resultados dos dois primeiros (VQTAM e GRBF) foram bem

próximos e o filtro KRBF foi o que apresentou os menores valores, demonstrando bem a eficiência do processo de interpolação feito pela função não-linear gaussiana. Um ponto importante a se comentar sobre o filtro KRBF é que ele demonstrou robustez à variação no valor de K, sendo capaz de manter sempre uma característica constante no valor do NMSE visto no gráfico.

O último modelo proposto por este trabalho e que utiliza também a técnica VQTAM, foi o filtro KSOM. Este filtro realiza interpolação linear com base nas informações armazenadas nos protótipos de entrada/saída. Como foi visto nos gráficos do erro de generalização em função do parâmetro q, este filtro sofreu restrições devido o aumento no número de neurônios vencedores que dificultou na obtenção dos coeficientes a partir do cálculo da *pseudo-inversa* para então montar os hiperplanos e poder assim calcular a saída estimada, caracterizando uma não-robustez a variação de K. Analisando os problemas de identificação e equalização de canais, os resultados obtidos não foram satisfatórios, pois demonstrou-se uma limitação no processo de interpolação linear na qual este filtro se baseia, devido à obtenção de erro de aproximação na estimação da saída, figurando assim as suas respostas sempre acima dos valores obtidos pelo filtro MLP.

Analisando todas as respostas obtidas pelos filtros lineares e não-lineares, conclui-se que os modelos propostos baseados na interpolação não-linear, tais como GRBF e KRBF, apresentaram resultados melhores que os resultados do filtro MLP, o que era esperado por este trabalho.

Para finalizar, serão listadas as futuras contribuições que possibilitarão a continuidade deste trabalho e, com isso, demonstrarão o aproveitamento do conhecimento adquirido no decorrer dos estudos desta dissertação. Segue abaixo:

- implementação de um modelo auto-regressivo linear local usando uma versão crescente (*growing*) do Mapa de Operadores (Operator Map) para extração de características de sinais de EEG, e compará-lo com modelos auto-regressivos alternativos, tais como auto-regressivos lineares uni- e multi-variável;
- modificação do modelo do filtro LLM para obter uma estrutura que seja capaz de fazer interpolação não-linear;
- implementação de técnicas de predição de séries temporais baseadas na rede auto-organizável de Kohonen;
- análise do efeito do número de entradas, ou seja, o tamanho da seqüência apresentada ao filtro;
- implementação de mapas auto-organizáveis com meta-dinâmica;

- realização de experimentos com outros tipos de canais;
- emprego em equalização cega;
- adoção de um processo de definição automática de parâmetros dos filtros;
- outras propostas para otimização de parâmetros dos filtros.

APÊNDICE A – Algoritmo de Geração dos Sinais de Entrada e Saída do Canal não-Linear

Segue abaixo o programa (feito em Matlab®) utilizado para gerar a seqüência de símbolos de entrada $\{x(t)\}$, o canal não-linear e a saída do canal $\{y(t)\}$:

```
% ————————————————————————————————— %
function [X,Y,N_train,N_test]=nlsignal(N)

   % N: NÚMERO TOTAL DE AMOSTRAS DE SINAL
N_distribution=[5/6 1/6];
N_train=floor(N_distribution(1)*N);
N_test=floor(N_distribution(2)*N);

   % ————————————————————————————————— %
% 1.  GERAÇÃO DO SINAL. Sinal X, modelo Gauss-Markov AR,
% potência X_P, N amostras

   % 1.1 Filtro FIR, modelo AR de ordem 1
X_tau=20; Constante de Tempo Associada com a
X_a = exp(-1/X_tau); % Coeficiente a
X_N_transiente=5*X_tau;
X_b=.1; % Coeficiente b

   % 1.2 Ruido branco de entrada
E_P=1; % Potencia do Ruido
E=sqrt(E_P)*randn(1,N + X_N_transiente); % Ruido branco N(0,Pe);
```

```
   % 1.3 Sinal Gauss-Markov, normalizado pela potencia S_P
X_P_filtro = X_b^2/(1 - X_a^2)*E_P; \% Potencia da saida do filtro
X_P=1; % Potencia do Sinal de Entrada
tmp=filter([X_b],[1 - X_a],E);
X=sqrt(X_P/X_P_filter)*tmp(X_N_transient:X_N_transient + N-1);


   % ---------------------------------------------------------- %
% 2.  TRANSFORMAÇÃO DO CANAL . Ruidoso, não-linear, com
% memória.  Saída Y, N amostras


   % 2.1 Geração do ruído.  Ruído W, modelo AWGN, potência W_P, N amostras
W_P=0.03; % Potencia do Ruido
W=sqrt(W_P)*randn(1,N);


   % 2.2 Ruído mais Sinal.  Sinal ruidoso V, N amostras
V = X + W;


   % 2.3 Efeito do canal LTI FIR. Saída U, (última) N amostras
C_h=[1.0 0.8 0.5];  % Artigo do Cowan
C_h=C_h./norm(C_h); % Resposta ao impulso do canal C_h
U=filter(C\_h,[1],V);


   % 2.4 Efeito do canal não-linear.  Saída Y, N amostras
G_1 = 0.2*U - 0.2*U.^2+ 0.04*U.^3;
G_2 = 0.9*atan(U - 0.1*U.^2 + 0.5*U.^3 - 0.1*U.^4 + 0.5);
Y = G_1 + G_2;


   % 2.5 Variância dos dados de entrada e saída
X_sigma=var(X); \% Variancia de Entrada
Y_sigma=var(Y); % Variancia de Saida
```

Referências

ABRANTES, S. A. *Processamento Adaptativo de Sinais*. Lisboa, Portugal: Fundação Calouste Gulbenkian, 2000.

ADALI, T.; LIU, X.; Sï¿½MEZ, M. K. Conditional distribution learning with neural networks and its application to channel equalization. *IEEE Transactions on Signal Processing*, v. 45, n. 4, p. 1051–1064, 1997.

AGUIRRE, L. *Introdução à Identificação de Sistemas*. Belo Horizonte, MG: Editora UFMG, 2000.

AL-MASHOUQ, K. A.; REED, I. S. The use of neural nets to combine equalization with decoding for severe intersymbol interference channels. *IEEE Transactions on Neural Networks*, v. 5, n. 6, p. 982–988, 1994.

BARRETO, G. et al. Previsï¿½ de sï¿½ies temporais nï¿½-estacionï¿½ias usando modelos locais baseados em redes neurais competitivas. In: *Anais do VI Simpï¿½io Brasileiro de Automaï¿½o Inteligente (SBAI'03)*. [S.l.: s.n.], 2003. p. 941–946.

BARRETO, G. et al. Nonstationary time series prediction using local models based on competitive neural networks. *Lecture Notes in Computer Science*, v. 3029, p. 1146–1155, 2004.

BARRETO, G. A.; ARAúJO, A. F. R. Identification and control of dynamical systems using the self-organizing map. *IEEE Transactions on Neural Networks*, v. 15, n. 5, p. 1244–1259, 2004.

BARRETO, G. A.; ARAúJO, A. F. R.; RITTER, H. J. Self-organizing feature maps for modeling and control of robotic manipulators. *Journal of Inteligent and Robotic Systems*, v. 36, n. 4, p. 407–450, 2003.

BAUER, H.-U.; VILLMANN, T. Growing a hypercubical output space in a self-organizing feature map. *IEEE Transactions on Neural Networks*, v. 8, n. 2, p. 218–226, 1997.

BLUM, E. K.; LI, L. K. Approximation theory and feedforward networks. *Neural Networks*, v. 4, p. 511–515, 1991.

BOUCHIRED, S. et al. Equalization of satellite mobile communication channels using combined self-organizing maps and RBF networks. In: *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*. Seattle: [s.n.], 1998. v. 6, p. 3377–3379.

CHANG, P.-H.; WANG, B.-C. Adaptive decision feedback equalization for digital satellite channels using multilayer neural networks. *IEEE Journal on Selected Areas in Communications*, v. 13, n. 2, p. 316–324, 1995.

CHEN, S.; MULGREW, B.; GRANT, P. A clustering technique for digital communications channel equalization using radial basis function. *IEEE Transactions on Neural Networks*, v. 4, n. 4, p. 570–579, 1993.

CHEN, T.; CHEN, H. Approximation of continuous functional by neural networks with application to dynamic systems. *IEEE Transactions on Neural Networks*, v. 4, n. 6, p. 910–918, 1993.

CHNG, E.-S.; YANG, H. H.; SKARBEK, W. Reduced complexity implementation of the bayesian equaliser using local rbf network for channel equalisation problem. *Electronics Letters*, v. 32, n. 1, p. 17–19, 1996.

COWAN, C. F. N. Communications equalization using non-linear adaptive networks. *Proceedings of the IEEE*, p. 199–202, 1991.

DABLEMONT, S. et al. Time series forecasting with som and local non-linear models - application to the DAX30 index prediction. In: *Proceedings of the 4th Workshop on Self-Organizing Maps (WSOM'03)*. [S.l.: s.n.], 2003. p. 340–345.

DaSILVA, M. T. M. *Equalizaï¿½o Nï¿½-linear de Canais de Comunicaï¿½o*. Dissertação (Mestrado) — Departamento de Engenharia de Telecomunicaï¿½es e Controle, Escola Politï¿½nica da Universidade de Sï¿½ Paulo, Sï¿½ Paulo, SP, 2001.

DEBODT, E. et al. On the use of self-organizing maps to accelerate vector quantization. *Neurocomputing*, v. 56, p. 187–203, 2004.

DINIZ, P. S. R.; CAMPOS, M. L. R.; ANTONIOU, A. Analysis of LMS-Newton adaptive filtering algorithms with variable convergence factor. *IEEE Transactions on Signal Processing*, v. 43, n. 3, p. 617–627, 1995.

DONG, M.; TONG, L.; SADLER, B. Optimal pilot placement for time-varying channels. In: *Proceedings of the IEEE Workshop on Signal Processing Advances in Wireless Communications (SPAWC'2003)*. Rome, Italy: [s.n.], 2003. p. 219–223.

DONG, M.; TONG, L.; SADLER, B. Optimal insertion of pilot symbols for transmissions over time-varying flat fading channels. *IEEE Transactions on Signal Processing*, v. 52, n. 5, p. 1403–1418, 2004.

ERDOGMUS, D. et al. Nonlinear channel equalization using multilayer perceptrons with information-theoretic criterion. In: *Proceedings of the IEEE Workshop on Neural Networks for Signal Processing, (NNSP)'01*. [S.l.: s.n.], 2001. p. 443–451.

FANCOURT, C. L.; PRINCIPE, J. C. Competitive principal component analysis for locally stationary time series. *IEEE Transactions on Signals Processing*, v. 46, n. 11, p. 3068–3081, 1998.

FARHANG-BOROUJENY, B. Fast lms/newton algorithms based on autoregressive modeling and their application to acoustic echo cancellation. *IEEE Transactions on Signal Processing*, v. 45, n. 8, p. 1987–2000, 1997.

FRITZKE, B. A growing neural gas network learns topologies. In: *Advances in Neural Information Processing Systems (NIPS)*. [S.l.: s.n.], 1994. p. 625–632.

GABOR, D.; WILBY, W.; WOODCOCK, R. A universal non-linear filter, predictor and simulator which optmises itself by a learning process. In: *IEE Proceedings*. [S.l.: s.n.], 1960. v. 109, p. 422–435.

GEMAN, S.; BIENENSTOCK, E.; DOURSAT, R. Neural networks and the bias/variance dilemma. *Neural Computation*, MIT Press, Cambridge, MA, USA, v. 4, n. 1, p. 1–58, 1992. ISSN 0899-7667.

GENï¿½Y, R.; LIU, T. Nonlinear modeling and prediction with feedforward and recurrent networks. *Physics D*, v. 108, p. 119–134, 1997.

GEVA, S.; SITTE, J. A constructive method for multivariate function approximation by multilayer perceptrons. *IEEE Transactions on Neural Networks*, v. 3, n. 4, p. 621–624, 1992.

GIBSON, G. J.; COWAN, C. F. N. On the decisions regions of multilayer perceptrons. *Proceedings of the IEEE*, v. 78, n. 10, p. 1590–1594, 1990.

GIBSON, G. J. et al. The application of nonlinear architectures to adaptive channel equalization. In: *Proceedings of IEEE*. Atlanta, (USA): [s.n.], 1990. p. 312.8.1–312.8.5.

GIBSON, G. J.; SIU, S.; COWAN, C. F. N. Multilayer perceptron structures applied to adaptive equalisers for data communications. *Proceedings of IEEE*, p. 1183–1186, 1989.

GIBSON, G. J.; SIU, S.; COWAN, C. F. N. The application of nonlinear structures to the reconstruction of binary signals. *IEEE Transactions on Signal Processing*, v. 39, n. 8, p. 1877–1884, 1991.

HARTMAN, E. J.; KEELER, J. D.; KOWALSKI, J. M. Layered neural networks with Gaussian hidden units as universal approximations. *Neural Computation*, v. 2, n. 2, p. 210–215, 1990.

HAYKIN, S. *Neural Networks: A Comprehensive Foundation*. Englewood Cliffs, NJ: Macmillan Publishing Company, 1994.

HAYKIN, S. *Adaptive Filter Theory*. 3. ed. Englewood Cliffs, NJ: Prentice Hall, 1996.

HEBB, D. O. *The Organization of Behavior*. New York: John Wiley, 1949.

HERTZ, J.; KROGH, A.; PALMER, R. G. *Introduction to the theory of neural computation*. Redwood City, CA: Addison-Wesley, 1991.

HOFMANN, T.; BUHMANN, J. Competitive learning algorithms for robust vector quantization. *IEEE Transactions on Signal Processing*, v. 46, n. 6, p. 1665–1675, 1998.

HORNIK, K.; STINCHCOMBE, M.; WHITE, H. Multilayer feedforward networks are universal approximators. *Neural Networks*, v. 2, n. 5, p. 359–366, 1989.

HWANG, J.-N. et al. The past, present, and future of neural networks for signal processing. *IEEE Signal Processing Magazine*, v. 14, n. 6, p. 28–48, 1997.

IBNKAHLA, M. Applications of neural networks to digital communications – a survey. *Signal Processing*, v. 80, n. 7, p. 1185–1215, 2000.

IBNKAHLA, M. et al. Neural network modeling and identification of nonlinear channels with memory: Algorithms, applications, and analytic models. *IEEE Transactions on Signals Processing*, v. 46, n. 5, p. 1208–1220, 1998.

IBNKAHLA, M.; CASTANIE, F. Vector neural networks for digital satellite communications. In: *Proceedings of the IEEE International Conference on Communications (ICC'95)*. Seattle: [s.n.], 1995. v. 3, p. 1865–1869.

JACOBS, R. A. et al. Adaptive mixtures of local experts. *Neural Computation*, v. 3, p. 79–87, 1991.

JAIN, A. K.; MAO, J.; MOHIUDDIN, K. M. Artificial Neural Networks: A Tutorial. *Computer*, v. 29, n. 3, p. 31–44, March 1996.

JOHNSON, C. R. On the interaction of adaptive filtering, identification and control. *IEEE Signal Processing Magazine*, p. 22–37, 1995.

KECHRIOTIS, G.; ZERVAS, E.; MANOLAKOS, E. S. Using recurrent neural networks for adaptive communication channel equalization. *IEEE Transactions on Neural Networks*, v. 5, n. 2, p. 267–278, 1994.

KOHONEN, T. Clustering, taxonomy, and topological maps of patterns. In: *Proceedings of the 6th International Conference on Pattern Recognition (6ICPR)*. Washington, DC: IEEE Computer Soc. Press., 1982. p. 114–128.

KOHONEN, T. et al. Combining linear equalization and self-organizing adaptation in dynamic discrete-signal detection. In: *Proceedings of the IEEE International Joint Conference on Neural Networks (IJCNN'90)*. [S.l.: s.n.], 1990. v. 1, p. 223–228.

KOHONEN, T. K. The self-organizing map. *Proceedings of the IEEE*, v. 78, n. 9, p. 1464–1480, 1990.

KOHONEN, T. K. *Self-Organizing Maps*. 2nd extended. ed. Berlin, Heidelberg: Springer-Verlag, 1997.

KOHONEN, T. K. The self-organizing map. *Neurocomputing*, v. 21, p. 1–6, 1998.

KOSKELA, T. et al. Recurrent SOM with local linear models in time series prediction. In: *Proceedings of the ESANN European Symposium on Artificial Neural Networks (ESANN'98)*. Bruges, Belgium: [s.n.], 1998. p. 167–172.

KUMAR, P.; SARATCHANDRAN, P.; SUNDARARAJAN, N. Minimal radial basis function neural networks for nonlinear channel equalization. *Proceedings of the IEE Vision Image Signal Processing*, v. 147, n. 5, p. 428–435, 2000.

LU, B.; EVANS, B. Channel equalization by feedforward neural networks. In: *Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS'99)*. [S.l.: s.n.], 1999. v. 5, p. 587–590.

MARTINETZ, T.; SCHULTEN, K. Topology representing networks. *Neural Networks*, v. 7, n. 3, p. 507–522, 1994.

MCCULLOCH, W. S.; PITTS, W. H. A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics*, v. 5, n. 1, p. 115–133, 1943.

MINSKY, M.; PAPERT, S. *Perceptrons: an Introduction to Computational Geometry.* Cambridge,Massachusets: MIT Press, 1969.

MISRA, S.; SWAMI, A.; TONG, L. Cutoff rate analysis of the Gauss-Markov fading channel with adaptive energy allocation. In: *Proceedings of the IEEE Workshop on Signal Processing Advances in Wireless Communications (SPAWC'2003).* Rome, Italy: [s.n.], 2003. p. 388–392.

MISRA, S.; SWAMI, A.; TONG, L. Optimal training over the gauss-markov fading channel: a cutoff rate analysis. In: *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP'2004).* Montreal,Canada: [s.n.], 2004. v. 3, p. 809–812.

MULGREW, B. Applying radial basis function. *IEEE Signal Processing Magazine*, v. 13, n. 2, p. 50–65, 1996.

NARENDA, K. S.; PARTHASARATHY, K. Identification and control of nonlinear systems using neural networks. *IEEE Transactions on Neural Networks*, v. 1, n. 4, p. 4–27, 1990.

OJA, M.; KASKI, S.; KOHONEN, T. Bibliography of self-organizing map (SOM) papers: 1998-2001 addendum. *Neural Computing Surveys*, v. 3, p. 1–156, 2003.

ONG, S. et al. A decision feedback recurrent neural equalizer as an infinite impulse response filter. *IEEE Transactions on Signals Processing*, v. 45, n. 11, p. 2851–2858, 1997.

PARISI, R. et al. Fast adaptive digital equalization by recurrent neural networks. *IEEE Transactions on Signal Processing*, v. 45, n. 11, p. 2731–2739, 1997.

PARK, J.; SANDBERG, I. W. Universal approximation using radial-basis-function networks. *Neural Computation*, v. 3, n. 2, p. 246–257, 1991.

PARK, J.; SANDBERG, I. W. Approximation and radial-basis-function networks. *Neural Computation*, v. 5, n. 2, p. 305–316, 1993.

PENG, M.; NIKIAS, C. L.; PROAKIS, J. G. Adaptive equalization with neural networks: New multi-layer perceptron structures and their evaluation. *IEEE*, v. 2, p. 301–304, 1992.

PINKUS, A. Approximation theory of the MLP model in neural networks. *Acta Numerica*, v. 8, p. 143–195, 1999.

POGGIO, T.; GIROSI, F. Networks for approximation and learning. *Proceedings of the IEEE*, v. 78, n. 9, p. 1481–1497, 1990.

PRINCIPE, J. C.; EULIANO, N. R.; LEFEBVRE, W. C. *Neural Adaptive Systems: Fundamentals Through Simulations.* New York, NY: John Willey & Sons, 2000.

PRINCIPE, J. C.; WANG, L.; MOTTER, M. A. Local dynamic modeling with self-organizing maps and applications to nonlinear system identification and control. *Proceedings of the IEEE*, v. 86, n. 11, p. 2240–2258, 1998.

RAIVIO, K. *Receiver Structures Based on Self-Organizing Maps.* Tese (Doutorado) — Department of Computer Science and Engineering, Helsinki University of Technology, Espoo, Finland, 1999.

RAIVIO, K. et al. Performance of two neural receiver structures in the presence of co-channel interference. In: *Proceedings of the International Conference on Neural Networks (ICNN'97).* [S.l.: s.n.], 1997. v. 4, p. 2080–2084.

RAIVIO, K.; HENRIKSSON, J.; SIMULA, O. Improving design feedback equaliser performance using neural networks. *Electronics Letters*, v. 27, n. 23, p. 2151–2153, 1991.

RAIVIO, K.; HENRIKSSON, J.; SIMULA, O. Neural detection of QAM signal with strongly nonlinear receiver. *Neurocomputing*, v. 21, n. 1–3, p. 159–171, 1998.

REBOLLO-MONEDERO, D. *Examples of non-linear estimation.* [S.l.], 2001.

REUTER, M.; ZEIDLER, J. R. Nonlinear effects in LMS adaptive equalizers. *IEEE Transactions on Signal Processing*, v. 47, n. 6, p. 1570–1579, 1999.

ROSENBLATT, F. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, v. 65, p. 386–408, 1958.

SADEGH, N. A perceptron network for functional identification and control of nonlinear systems. *IEEE Transactions on Neural Networks*, v. 4, n. 6, p. 982–988, 1993.

SCHILLING, R. J.; JR., J. J. C.; AL-AJLOUNI, A. F. Approximation of nonlinear systems with radial basis function neural networks. *IEEE Transactions on Neural Networks*, v. 12, n. 1, p. 1–15, 2001.

SIU, S.; GIBSON, G. J.; COWAN, C. F. N. Decision feedback equalization using neural network structures and performance comparison with standard architecture. *Proceedings of the IEEE*, v. 137, n. 4, p. 221–225, 1990.

SOUZA, L. G. M.; BARRETO, G. A.; MOTA, J. C. M. Using the self-organizing map to design efficient RBF models for nonlinear channel equalization. In: *Submetido a 5th Workshop on Self-Organizing Maps (WSOM'05).* [S.l.: s.n.], 2005.

SPECHT, D. F. A general regression neural network. *IEEE Transactions on Neural Networks*, v. 2, n. 6, p. 568–576, 1991.

WALTER, J.; RITTER, H.; SCHULTEN, K. Non-linear prediction with self-organizing map. In: *Proceedings of the IEEE International Joint Conference on Neural Networks (IJCNN'90).* [S.l.: s.n.], 1990. v. 1, p. 587–592.

WIDROW, B. Generalization and information storage in networks of ADALINE neurons. In: YOVITS, M.; JACOBI, G.; GOLDSTEIN, G. (Ed.). *Self-Organizing Systems.* Washington, DC: Spartan Books, 1962.

WIDROW, B.; HOFF, M. E. Adaptive switching circuits. In: *IRE WESCON Convention Record-Part 4.* [S.l.: s.n.], 1960. p. 96–104.

WIDROW, B.; KAMENETSKY, M. Statistical efficiency of adaptive algorithms. *Neural Networks*, v. 16, n. 5–6, p. 735–744, 2003.

WIDROW, B.; LEHR, M. 30 years of adaptive neural networks: Perceptron, madaline and backpropagation. In: *Proceedings of the IEEE*. [S.l.: s.n.], 1990. v. 78, p. 1415–1442.

WIDROW, B.; STEARNS, S. D. *Adaptive Signal Processing*. Upper Saddle River, NJ: Prentice-Hall, 1985.

WIDROW, B.; WINTER, R. Neural nets for adaptive filtering and adaptive pattern recognition. *IEEE Computer*, v. 21, n. 3, p. 25–39, 1988.

WIENER, N. *Extrapolation, Interpolation, and Smoothing of Stationary Time Series*. New York: John Willey and Sons, 1949.

WIENER, N.; HOPF, E. On a class of singular integral equations. In: *Proceedings of the Prussian Academic Math-Physics*. [S.l.: s.n.], 1931. p. 696.

YAIR, E.; ZEGER, K.; GERSHO, A. Competitive learning and soft competition for vector quantizer design. *IEEE Transactions on Signals Processing*, v. 40, n. 2, p. 294–309, 1992.

YANG, S.-S.; TSENG, C.-S. An orthogonal neural network for function approximation. *IEEE Transactions on Systems, Man, and Cybernetics-Part B*, v. 26, n. 5, p. 779–783, 1996.

YOU, C.; HONG, D. Adaptive equalization using the complex backpropagation algorithm. *IEEE*, p. 2136–2141, 1996.

ZAKNICH, A. Introduction to the modified probabilistic neural network for general signal processing applications. *IEEE Transactions on Signal Processing*, v. 46, n. 7, p. 1980–1990, 1998.

ZAKNICH, A. *Neural Networks for Intelligent Signal Processing*. [S.l.]: World Scientific, 2003.