UNIVERSIDADE REGIONAL DO NOROESTE DO ESTADO DO RIO GRANDE DO SUL
DeFEM – DEPARTAMENTO DE FÍSICA, ESTATISTICA E MATEMÁTICA
DeTEC – DEPARTAMENTO DE TECNOLOGIA
MESTRADO EM MODELAGEM MATEMÁTICA

Elaine Maria Gosenheimer Fiori

IDENTIFICAÇÃO DE SISTEMAS NÃO-LINEARES UTILIZANDO REDES NEURAIS ARTIFICIAIS

Dissertação de Mestrado

Orientador: Prof. Dr. Gideon Villar Leandro

Ijui, 2007

Elaine Maria Gosenheimer Fiori

IDENTIFICAÇÃO DE SISTEMAS NÃO-LINEARES UTILIZANDO REDES NEURAIS ARTIFICIAIS

Dissertação apresentada ao programa de Pós-Graduação em Modelagem Matemática da Universidade Regional do Noroeste do Estado do Rio Grande do Sul, como requisito parcial para obtenção de grau de Mestre em Modelagem Matemática.

Ijui, 2007

UNIVERSIDADE REGIONAL DO NOROESTE DO ESTADO DO RIO GRANDE DO SUL
DeFEM – DEPARTAMENTO DE FÍSICA, ESTATISTICA E MATEMÁTICA
DeTEC – DEPARTAMENTO DE TECNOLOGIA

PROGRAMA DE PÓS-GRADUAÇÃO EM MODELAGEM MATEMATICA

IDENTIFICAÇÃO DE SISTEMAS NÃO-LINEARES UTILIZANDO REDES NEURAIS ARTIFICIAIS

Banca examinadora:

Prof. Doutor Gideon Villar Leandro – DeTEC/UNIJUI – Orientador

Prof. Doutor Laurence Duarte Colvara – UNESP/SP – Examinador

Prof. Doutor Oleg Khatchatourian – DeFEM/UNIJUI – Examinador

Ijui, 2007

AGRADECIMENTOS

Agradeço em primeiro lugar a Deus por permitir que eu viajasse por tanto tempo tão longo caminho, pela força, pela fé, pela perseverança e otimismo.

Ao meu orientador Gideon pelos ensinamentos, pela paciência, atenção e amizade.

Ao meu esposo Daltro pela força, paciência, compreensão, tolerância, carinho e amor.

Aos meus pais Arno e Diva pelo incentivo, pelo apoio e esforços, não somente neste momento, mas, em todas as etapas de minha vida.

Aos meus sogros, em especial a dona Lori por cuidar de meu filho na minha ausência.

À UNIJUÍ pela bolsa de filantropia concedida.

Aos vizinhos colegas e amigos que fizeram parte desta caminhada em especial ao casal Loreni e Loreno.

Aos funcionários(as), da UNIJUÍ pela disposição no atendimento.

RESUMO

Este trabalho apresenta um estudo de implementação de uma rede neural perceptron multicamadas para identificação de sistemas não-lineares. Nesta modalidade de identificação de sistemas a rede neural é o modelo, que trabalha com dados de entrada e de saída. O algoritmo de treinamento para o ajuste dos pesos da rede corresponde ao estágio de estimação de parâmetros. A seleção da arquitetura da rede corresponde à seleção do modelo, o número de entradas, saídas e atraso do tempo estão relacionados com a ordem do modelo. Para efetivar o aprendizado da rede (atualização dos pesos) utilizou-se a técnica Back-Propagation. O algoritmo de aprendizagem Back-Propagation padrão utiliza uma taxa de aprendizagem η que define a "velocidade" da aprendizagem. Quando pequena, ela gera pequenas variações na mudança dos pesos da rede de uma iteração para a outra, aumentando o tempo e o custo computacional. Porém, se a taxa de aprendizagem for grande corre-se o risco de uma variação muito grande dos pesos da rede de uma iteração para outra, gerando uma trajetória oscilatória e conseqüentemente levando a divergência de valores. Usualmente a taxa de aprendizagem é fixa e definida através de testes (ensaios) arbitrariamente. Este inconveniente torna o processo de ajuste da taxa de aprendizagem lento, massivo e com incertezas. Com isto, torna-se de grande interesse a definição de uma taxa de aprendizagem baseada em princípios científicos e que produza bons resultados. Para atingir tais objetivos propõe-se uma metodologia que fornece a cada iteração um valor para a taxa de aprendizagem, garantindo um número menor de iterações para satisfazer o índice de desempenho especificado (erro final). Para a obtenção dos valores da taxa de aprendizagem em cada iteração são utilizados os Algoritmos Genéticos. Resultados de simulação mostram a viabilidade da metodologia proposta, disponibilizando ao usuário de redes neurais um método automático, para determinação da taxa de aprendizado, confiável e seguro.

ABSTRACT

This work presents a study of implementation of neural net perceptron multilayers to identify non linear systems. In this kind of system identification the neural net is the model that works with data of incomes and outcomes. The training algorithm to adjust the weights of the net corresponds to the level of the parameter estimate. The choice of the net architecture corresponds to the model's choice; the number of incomes and outcomes and the delay of time are related with the order of the model. To accomplish the learning of the net (update of the weights) it was used the Back-Propagation technique. The algorithm of Back-Propagation learning standard uses a learning range that defines the "speed" of the learning. When small, it generates a small variation in the change of the weights of the net from one iteration to the other, increasing the time and the computational cost. However, if the learning range is large, there is a risk of a too large variation of the net´s weight from one iteration to another, generating an oscillatory trajectory and consequently leading to a divergence of the values. Usually the learning range is fixed and it is defined through tests (assays) arbitrarily. This inconvenience makes the process of the learning adjustment range slow and with uncertainties. Because of that, the definition of a range of learning based on scientific principles and their good results become a great interest. To reach such objectives we propose a methodology that is able to supply each iteration with a value of the learning range, guaranteeing a lesser number of iterations to satisfy the index of specified performance (final error). For the attainment of the values of the learning's range in each iteration, the Genetic Algorithms are used. Results of simulation show the viability of the methodology proposed, making available to the user of neural nets an automatic method, for the determination of the range of learning, in a trustworthy and safe way.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SÍMBOLOS E ABREVIATURAS

ARMAX Autoregressivo com Média Móvel e Entradas Exógenas

ARX Autoregressivo com Entradas Exógenas

BIBO Entrada Limitada Saída Limitada

FIR Resposta ao Impulso Finita

E Função erro global ou função custo

e Erro local de treinamento de uma *MLP*

MIMO Entradas Múltiplas, Saídas Múltiplas

NARMAX Modelo Autoregressivo com Média Móvel e Entradas Exógenas

NARX Modelo Autoregressiva com Entradas Exógenas

NFIR Modelo com Resposta ao Impulso Finita

NOE Modelo com Erro na Saída

OE Erro na Saída

RNA Rede Neural Artificial

MLP Rede Neural Multicamadas (*perceptron*)

SISO Entrada única, Saída única

x Vetor de entrada *x*.

y Vetor de saída *y*.

xi i-´esima componente do vetor *x*.

$\hat{y}i$ Valor estimado de *yi*.

$f(\cdot)$, $\phi(\cdot)$ Função de ativação.

q-1 Operador atraso.

wji Peso entre o neurônio *i* da camada de entrada e o neurônio *j* da camada intermediária.

vkj Peso entre o neurônio *j* da camada intermediária e o neurônio *k* da camada de saída.

η Taxa de aprendizado.

t Tempo.

Ts Período de amostragem da planta

α Constante de momento

δ Vetor de erro da Regra Delta

θ Parâmetro de um sistema

SUMÁRIO

INTRODUÇÃO

Para trabalhar com um sistema é necessário conhecer, estudar e compreender a sua essência. Isto pode ser feito a partir de um modelo. Um modelo contém as características principais de um sistema real possibilitando ao usuário aprender, comunicar, monitorar, controlar e fazer previsões. O processo usual é selecionar, no sistema, argumentos ou parâmetros considerados essenciais e formalizá-los através de um sistema artificial, 'o modelo' (Bassanezi, 2002). Quando se transcreve o funcionamento de um sistema com linguagem matemática obtém-se um modelo matemático. O estudo dos diferentes métodos de implementar modelos matemáticos denomina-se modelagem matemática.

Existem várias formas de modelagem matemática, uma possível classificação conforme (Aguirre, 2004) é: modelagem caixa branca, modelagem caixa preta e modelagem caixa cinza. A primeira caracteriza-se pela modelagem baseada na física ou natureza do processo, o que faz com que o modelo nem sempre seja viável devido ao tempo e conhecimento do sistema para modelá-lo. A segunda requer pouca ou nenhuma informação a respeito do sistema a ser modelado, denominada também, de modelagem empírica ou identificação. E a última, envolve as duas formas de modelagem supracitadas, conta com um conhecimento parcial do sistema, isto é, informações além dos dados de entrada e de saída.

A identificação de sistemas se propõe a encontrar um modelo matemático que descreva as relações existentes entre dados de entrada e de saída de um sistema real. É uma área do conhecimento com significativas evoluções no decorrer das últimas décadas, sendo que os modelos lineares são os mais desenvolvidos devido a um maior conhecimento e facilidade de desenvolvimento, se comparados com sistemas não lineares (Billings, 1980).

Avançadas ferramentas de álgebra linear são empregadas na análise de sistemas dinâmicos, quando estes podem ser aproximados linearmente. Porém, grande parte dos sistemas de ordem prática apresenta não linearidades significativas (geração de harmônicas, múltiplos pontos de equilíbrio e caos), quando são resolvidos por aproximação linear perde-se muita informação o que acaba por reduzir sua eficiência. Sistemas não-lineares se mostram

mais complexos em seu comportamento e exigem métodos de resolução mais avançados que se adaptam as dinâmicas.

Nas últimas décadas diversas concepções de algoritmos para modelagem e identificação de sistemas dinâmicos complexos têm sido propostos na literatura, tais como: Métodos Freqüênciais, técnicas baseadas em estimativas de modelos Wiener, Hammerstein, Bilinear e Volterra, regressão não-linear, wavelets e identificação recursiva (Ljung, 1997; Haber & Unbehauen, 1990). Uma abordagem relevante entre tantas outras para representação matemática de sistemas dinâmicos com comportamento complexo ou caótico é a das redes neurais.

Redes neurais artificiais são constituídas de neurônios os quais trabalham conectados paralelamente recebendo sinais, processando-os através de funções matemáticas previamente selecionadas e enviando respostas ao sistema.

Para a implementação de uma rede neural multicamadas segue-se etapas pré-definidas: a definição de entradas e saídas da rede; a determinação da arquitetura empregada; e a decisão de qual algoritmo de aprendizagem utilizar. Os dados de entrada e saída precisam ser suficientemente grandes para representar à dinâmica do sistema. A definição da arquitetura da rede é baseada diretamente no algoritmo de aprendizagem, e consiste na escolha de quantas camadas serão empregadas e quantos neurônios irão compor cada camada e as respectivas funções de ativação.

O aprendizado de uma rede neural é um processo de adaptação dos pesos sinápticos e de níveis de “bias” através de impulsos gerados pelo ambiente no qual esta inserida. E se da de acordo com o algoritmo utilizado. Existem algumas formas de aprendizagem, dentre as quais se destacam: aprendizado supervisionado o qual conta com um “professor” que teoricamente supervisiona os resultados da rede; aprendizagem não supervisionada é uma metodologia na qual a rede aprende a ajustar seus parâmetros baseados em tendências estatísticas observada nos dados de entrada (Lima, 2000); aprendizagem híbrida é caracterizada pelo emprego das duas aprendizagens supracitadas; e o aprendizado por reforço consiste no método da tentativa e erro de modo a otimizar um índice de desempenho, denominado sinal de reforço.

Dentro do contexto dos treinamentos pode-se apresentar algoritmos que empregam formas mais específicas de treinamento. Rede Adaline emprega o algoritmo dos Mínimos

Quadrados. Rede Hebbiana emprega a lei de aprendizagem de Hebb baseado em eventos de recompensas e punições. Rede de Kohonen é baseada na aprendizagem competitiva. Rede Neural de Base Radial emprega funções de base radial não-linear, como também algoritmos de Pseudo-Inversão e OLS (Orthogonal Least Squares). A Rede Perceptrons de Multicamadas que usualmente emprega o algoritmo de Retropropagação do Erro, pois o mesmo permite o emprego de arquiteturas sofisticadas com capacidade de resolução de problemas amplos.

A opção por redes neurais para identificação de sistemas não-lineares é baseada em algumas características da planta, tais como:

- Pouco conhecimento da planta;
- A apresentação de não-linearidades significativas da planta;
- Somente dados de entrada e saída estão disponíveis.
- Disponibilidade de dados amostrados de toda a região de operação do sistema, (quando a representação for a espaços de estados).

Na identificação de sistemas a rede neural trabalha com pares de vetores u(t) e y(t) que são os dados de entrada e de saída do sistema respectivamente. O sinal de entrada é aplicado à rede e ela fornece uma saída $\hat{y}(t)$, o sinal erro é calculado e os pesos da rede são ajustados de tal forma que a saída da rede $\hat{y}(t)$ aproxime-se da saída real y(t) para todo u(t). O algoritmo de treinamento para o ajuste dos pesos da rede corresponde ao estágio de estimação de parâmetros. E a seleção da arquitetura da rede corresponde ao estágio de seleção do modelo.

Os estudos de modelos identificadores não-lineares foram intensificados a partir da segunda metade da década de 80 Leontaritis & Billings, 1985 a; Leontaritis & Billings, 1985 b; Sjöberg et al., 1995; Judisky et al., 1995.

Um período de grandes aplicações de redes neurais artificiais foi marcado por importantes autores Cybenko, 1989; Hornik et al., 1989; Mhaskar, 1992; Barron, 1993; Haikin, 2001.

Na área de identificação de sistemas não-lineares foram realizados inúmeros estudos Narendra e Parthasarathy, 1990; Hunt et al., 1992; Levin e Narendra, 1993; Levin e Narendra, 1995; Suykens et al., 1995; Levin e Narendra, 1996; Chen et al., 1997.

Um grande estudo foi desenvolvido por Levin e Narendra 1995 que propuseram um trabalho de aplicação de redes neurais artificiais a identificação de sistemas não-lineares através da observabilidade genérica não se preocupando muito com o método de ajuste de

parâmetros. Este método apresenta a vantagem de requer uma quantidade mínima de informações a respeito do sistema.

Neste trabalho é implementada uma rede neural do tipo perceptron multicamada utilizando a técnica back-propagation para a atualização dos pesos, na identificação de sistemas não-lineares.

O diferencial do trabalho se dá com relação à taxa de aprendizagem (η). A taxa de aprendizagem define a "velocidade" da aprendizagem. Quando pequena, ela gera pequenas variações na mudança dos pesos de uma iteração para a outra, aumentando o tempo e o custo computacional. Porém, quando ela se torna grande com o objetivo de acelerar a taxa de aprendizagem corre-se o risco de uma variação muito grande dos pesos gerando uma trajetória oscilatória podendo levar a divergência de valores.

Portanto, este trabalho propõe uma metodologia que fornece a cada iteração um valor para a taxa de aprendizagem, garantindo um número menor de iterações para satisfazer as especificações desejadas. O valor da taxa de aprendizagem é determinado através de Algoritmos Genéticos (AG). Os AG são algoritmos de procura baseados nas mecânicas de seleção e genética natural, onde os indivíduos mais fortes e aptos têm probabilidade maior de sobreviver e evoluir. Baseado nesta evolução a taxa de aprendizagem a cada iteração é determinada, objetivando a minimização do erro.

Esta dissertação está dividida da seguinte maneira. O capítulo 1 trata da fundamentação teórica sobre identificação de sistemas não-lineares, são apresentados vários aspectos a serem observados em problemas reais de identificação, os quais são agrupados em cinco etapas: coleta de dados, escolha da representação matemática, determinação da estrutura do modelo, estimação de parâmetros e validação do modelo (Aguirre, 2004). Também são abordados neste capítulo os modelos lineares ARX, ARMAX, e os modelos não-lineares NARX, NARMAX.

No capítulo 2 é feita uma breve revisão dos conceitos básicos de redes neurais: a fundamentação biológica, a descrição matemática de um neurônio artificial a algumas arquiteturas com seus respectivos algoritmos de aprendizagem.

O capítulo 3 é dedicado à estimação de parâmetros utilizando redes neurais, no qual aparecem as interpretações dos modelos Entrada-Saída, NARX e NARMAX por redes neurais

artificiais. Também apresenta-se uma descrição do método de ajuste do coeficiente de aprendizagem da rede neural através do Algoritmo Genético.

O capítulo 4 traz os resultados obtidos com a implementação da rede neural, para os modelos Entrada-Saída, NARX e NARMAX para as configurações com 4, 8 e 12 neurônios na camada oculta, testados com a taxa de aprendizagem fixa e variável.

O capítulo 5 apresenta as conclusões e propostas para trabalhos futuros.

1 IDENTIFICAÇÃO DE SISTEMAS NÃO-LINEARES

1.1 Introdução

O desenvolvimento de técnicas para obter equações matemáticas capazes de reproduzir de forma aproximada o comportamento dinâmico de sistemas reais é um dos temas mais fascinantes da cultura humana (Ljung, 1997). A modelagem matemática é uma área do conhecimento responsável por estudar maneiras de implementar modelos matemáticos de sistemas reais (Aguirre, 2004).

Desde os tempos mais remotos o homem tem procurado estudar e compreender fenômenos e sistemas para poder resolver problemas relacionados a eles sendo para esse fim empregadas as representações matemáticas dos sistemas (modelos matemáticos).

Os primeiros modelos matemáticos foram obtidos a partir de equações que descrevem a física do processo. Porém nem sempre era possível descrever as equações básicas do sistema. Com o crescente desenvolvimento da informática percebeu-se que seria possível desenvolver modelos matemáticos que a partir do processamento de dados coletados de um sistema seriam capazes de descrever a sua dinâmica. Procedimento este denominado de identificação de sistemas.

Apesar de a identificação de sistemas não lineares ainda ser uma arte a ser estudada e explorada, importantes estudos anteriores à década de oitenta podem ser vistos em Billings, 1980. Nesse artigo o autor cita os métodos de séries funcionais, de Volterra e técnicas de domínio de freqüência.

Na década de oitenta novos métodos para identificar sistemas não lineares surgiram, destacando-se a representação NARMAX polinomial proposta em (Leontaritis e Billings, 1985) e NARMAX racional (Billings e Chen, 1989a). Outro grande destaque em termos de representações não lineares são as redes neurais artificiais, as referências em termos de aplicações nesta área são os autores Narendra e Parthasarathi, 1990.

Após este período, surge um grande número de estudos em torno de representações e técnicas de identificação de sistemas não lineares e a possibilidade de utilizar informações a priori juntamente com dados de entrada e saída do sistema deu origem a uma nova classificação dos métodos de modelagem. Segundo (Aguirre, 2004) uma possível classificação é: modelagem caixa branca, modelagem caixa preta e modelagem caixa cinza.

A modelagem caixa branca consiste em um equacionamento dos fenômenos envolvidos, denominando-se também como modelagem fenomenológica ou conceitual geralmente determinadas pelas equações físicas ou químicas que regem o comportamento estático e dinâmico do sistema (Garcia, 1997). Esta forma de modelagem exige maior conhecimento e tempo o que faz com que o modelo nem sempre seja viável.

A modelagem caixa preta, conhecida também como modelagem empírica, não exige um profundo conhecimento do sistema, requer apenas dados de entrada e de saída, o que permite a escolha de uma estrutura conveniente facilitando a modelagem do sistema (Pottmann e Pearson, 1998).

Já a modelagem caixa cinza trabalha com as características consideradas (necessárias) as mais importantes para o modelo, diminuindo assim o número de variáveis envolvidas, facilitando a modelagem em relação à caixa branca. Em (Jorgernen e Hangos, 1995) a identificação caixa cinza é definida como "a ciência de construção de modelos que incorporam conhecimento à priori do sistema com um certo grau de incerteza na seleção da estrutura da representação".

Um modelo representa as características do sistema real em função dos objetivos para o qual é desenvolvido. Quanto mais aproximado do real, mais complexo será o processo de modelagem, pois se necessita conhecer os fenômenos que estão envolvidos no sistema e equacioná-los matematicamente.

A identificação de sistemas trata de estabelecer relações entre dados de entrada e de saída, e esta relação é traduzida através de um modelo que permite predizer resultados futuros a partir de dados inseridos. Mas etapas importantes precisam ser obedecidas para se obter uma identificação precisa: coleta de dados, escolha da representação matemática, determinação da estrutura do modelo, estimação de parâmetros e validação do modelo (Aguirre, 2004; Ljung, 1997).

Este capítulo aborda a identificação de sistemas concentrando-se nas cinco etapas citadas.

1.2 Coleta de Sinais

A coleta correta dos sinais de entrada u(k) e de saída y(k) é de fundamental importância, pois a correlação entre estes dados informa características do sistema necessárias para a modelagem. Uma maneira de certificar-se de uma escolha plausível é aplicar a função de correlação cruzada fcc equação (1.1) para determinar se há correlação significativa entre duas variáveis candidatas a compor um modelo. E quando se trata de sistemas multivariáveis aplica-se a função de autocorrelação fac equação (1.2) para verificação da existência de correlação entre as entradas. Caso a mesma ocorra, opta-se apenas por uma das entradas já que há correlação entre elas. Descartando entradas idênticas, evita-se a apresentação de mais variáveis e o modelo fica mais preciso. (Aguirre 2004).

$$r_{uy}(k) = \lim_{N\to\infty} \frac{1}{2N+1} \sum_{i=-N}^{N} u(i)y(i+k) \tag{1.1}$$

$$r_{uu}(k) = \lim_{N\to\infty} \frac{1}{2N+1} \sum_{i=-N}^{N} u(i)u(i+k) \tag{1.2}$$

1.2.1 Escolha dos sinais de entrada

Quando as funções de correlação e de autocorrelação não são aplicadas é necessário que as entradas sejam selecionadas segundo critérios semelhantes. Distinguindo-se duas maneiras: a interpretação numérica e a interpretação dinâmica.

Para excitar um determinado sistema é necessário que se tenha um conhecimento prévio acerca do que se espera, pois características não excitadas não aparecem nos dados e o que não aparece nos dados não é identificado (Aguirre, 2004). Sistemas lineares são comumente excitados com sinais binários pseudo-aleatórios, porém os mesmos não apresentam uma boa eficiência quando aplicados em sistemas não lineares, embora seja possível identificar bons modelos a partir do emprego de sinais do tipo aleatório.

Na identificação de sistemas não lineares muitas vezes é importante, quando possível, a excitação de uma "larga" faixa de amplitudes objetivando a observação de características dinâmicas e estáticas. Em contrapartida na identificação de sistemas reais a faixa de amplitude deve ser a menor possível para que a relação sinal-ruído seja aceitável (Ljung, 1997).

Neste tipo de modelos às vezes não se obtém dados em determinadas regiões, e para complementar este quesito, podem ser adicionados informações sobre o comportamento do sistema (Aguirre apud Corrêa, 2002).

Os dados podem ser utilizados de duas maneiras: quando os dados coletados são enviados diretamente para o processamento diz-se que os dados são utilizados em tempo real (on-line). Por outro lado, não havendo a possibilidade de utilizar os dados em tempo real, eles devem ser agrupados e guardados para uma futura estimação denominando o processo de off-line ou por janelas (Ljung, 1997).

1.2.2 Escolha do tempo de amostragem

Para muitas aplicações práticas utilizam-se amostras de variáveis, e quando os sistemas são contínuos faz-se necessária a discretizacão dos mesmos, e para que estes sinais representem características do sistema é preciso que o tempo de amostragem Ts (período entre duas amostras) seja curto. Segundo o teorema de Shannon o sinal que não tem componentes de freqüência acima de 1/(2Ts) pode ser determinado a partir de amostras de tal sinal separadas por Ts. Na prática a freqüência é normalmente escolhida entre 5 e 10 vezes maior do que a maior freqüência de interesse quando ela é conhecida.

Para sistemas não-lineares a determinação do período de amostragem passa por um método de autocorrelação e uma correlação não linear para detectar o tipo de iterações presentes naquele sinal. (Aguirre, 2004).

1.3 Representação matemática do modelo

Um modelo matemático de um determinado sistema deve reproduzir algumas características do sistema as quais baseiam-se nos objetivos para o qual o modelo esta sendo desenvolvido. A seguir serão apresentadas algumas considerações importantes na escolha dos modelos.

1.3.1 Representações lineares

A identificação de um sistema linear dá-se a partir da observação de seu comportamento em uma determinada faixa de operações. Sua linearidade é constatada quando se satisfaz o princípio da superposição. A superposição é observada quando um sistema ao ser excitado pela entrada $u_1(t)$ produz uma saída $y_1(t)$ e quando é excitado por uma entrada $u_2(t)$ produz uma saída $y_2(t)$. Este princípio é satisfeito se o sistema for excitado por $a*u_1(t) + b*u_2(t)$ produzir a saída $a*y_1(t) + b*y_2(t)$, sendo a e b constantes com possibilidade de serem complexas.

i) Representações lineares contínuas

Para representar matematicamente um modelo linear são empregadas funções, uma das mais usadas é a função de transferência, a qual fornece a transformada da resposta ao impulso h(t) do sistema, para condições iniciais nulas. Se a resposta ao impulso for contínua no tempo então a transformada usada é a de Laplace. Para um sinal x(t) a transformada de Laplace é definida como segue na equação (1.3). Enquanto que a transformada inversa de Laplace é definida conforme a equação (1.4). Considerando $s = \sigma + j\omega$.

$$X(s) = \int_0^{\infty} x(t) e^{-st} dt \tag{1.3}$$

$$x(t) = \frac{1}{2\pi j} \int_{\sigma - j\infty}^{\sigma + \infty} X(s) e^{st} ds \tag{1.4}$$

A função de transferência H(z) na forma discreta no tempo é representada pela transformada Z, para condições iniciais nulas. Quando é empregada uma seqüência x(k) apresenta-se a transformada Z através da equação (1.5), e a inversa da transformada Z é obtida pela equação (1.6), onde $\oint$ representa a integral fechada no sentido anti-horário, centrada na origem.

$$X(z) = \sum_{k=-\infty}^{k=\infty} x(k) z^{-k} \tag{1.5}$$

$$x(k) = \frac{1}{2\pi j} \oint X(z) z^{k-1} dz \tag{1.6}$$

Caso seja empregada a transformada de Fourier como resposta ao impulso h(t) do sistema a resposta em freqüência é apresentada como *H(jw)*, como observado na equação (1.7), e sua inversa na equação (1.8), onde $j = \sqrt{-1}$. Observa-se que *H(jw)* pode ser obtida a partir de *H(s)* substituindo-se *s* por $j\omega$.

$$X(j\omega) = \int_{-\infty}^{\infty} x(t) e^{-j\omega t} dt \tag{1.7}$$

$$x(t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} X(j\omega) e^{j\omega t} d\omega \tag{1.8}$$

A função de transferência comporta elementos que caracterizam a resposta temporal e a resposta em freqüência tais como: pólos, zeros e os resíduos. As funções de transferência como a equação (1.9) são normalmente representadas como o quociente de dois polinômios algébricos em s originados de equações diferenciais lineares e invariantes no tempo. Os valores $s = z_1, z_2, \ldots, z_q$, para os quais *H(s)* é nula são denominados de zeros da função (raízes do polinômio do numerador), enquanto os pólos são os valores de $s = p_1, p_2, \ldots, p_n$ responsáveis por zerar o polinômio do denominador (raízes do polinômio do denominador).

$$H(s) = \frac{N(s)}{D(s)} = \frac{b_0 + b_1 s + \ldots + b_q s^q}{a_0 + a_1 s + \ldots + a_n s^n} \tag{1.9}$$

O número de pólos de *H(s)* determina a ordem da função de transferência de um sistema. Em funções de transferência de sistemas reais tem-se $n \geq q$ (funções de transferência próprias) ou $n \leq q$ (funções de transferência estritamente próprias). Os pólos e zeros de uma função podem ser reais ou complexos. Quando complexos aparecem em pares conjugados, isto é, se $\alpha + j\beta$ for um pólo, então $\alpha - j\beta$ também será um pólo. O sistema é dito instável quando pelo menos um pólo da função de transferência possuir a parte real positiva, caso contrário é dito estável. Se todos os pólos possuírem a parte real negativa então o sistema é considerado assintoticamente estável.

ii) Representações lineares discretas

Os modelos de sistemas lineares invariantes e discretos no tempo podem ser definidos a partir da equação (1.10).

$$y(t) = \sum_{k=1}^{\infty} g(k)u(t-k), \qquad t = 0,1,2,\ldots \tag{1.10}$$

Onde $g(k)$ é a resposta ao impulso do sistema e $u(t)$ é o sinal de entrada (Ljung, 1997). Introduzindo-se um operador de deslocamento unitário por atraso q^{-1}, a equação (1.10) pode ser reescrita na forma reduzida como na equação (1.12),

$$y(t) = \sum_{k=1}^{\infty} g(k)(q^{-k}u(t)) \tag{1.11}$$

$$y(t) = \left[\sum_{k=1}^{\infty} g(k)q^{-k}\right]u(t) \;=\; G(q)u(t) \tag{1.12}$$

onde $G(q)$ é denominada a função de transferência do sistema. Introduzindo-se na equação (1.12) uma perturbação ou ruído aditivo, obtêm-se:

$$y(t) = G(q)u(t) + H(q)e(t), \tag{1.13}$$

com

$$H(q) = 1 + \sum_{k=1}^{\infty} h(k) q^{-k} \ , \tag{1.14}$$

Sendo que $e(t)$ representa uma seqüência de variáveis aleatórias com média zero e variância λ, $H(q)$ é considerada a função de transferência do ruído e o operador de deslocamento unitário q é definido conforme equação (1.15) (Ljung, 1997).

$$qu(t) = u(t+1) \qquad e \qquad q^{-1}u(t) = u(t-1) \tag{1.15}$$

Descrevendo-se a equação (1.13) como uma equação linear a diferenças, obtém-se uma relação entre entrada e saída representada na equação (1.16). Sendo considerado na equação (1.13) que $G(q) = B(q)/A(q)$ e $H(q) = 1$.

$$y(t) + a_1 y(t-1) + \ldots + a_n y(t-n) = b_1 u(t-1) + \ldots + b_m u(t-m) + a(q)e(t) \tag{1.16}$$

com

$$A(q) = 1 - a_1 q^{-1} - \ldots - a_n q^{-n}$$

$$B(q) = b_1 q^{-1} + \ldots + b_m q^{-m}$$

onde o termo $e(t)$ representa um ruído branco adicionado na equação a diferenças tornando o modelo conhecido como *modelo de erro na equação*. Pode-se reescrever a equação (1.13), usando-se o operador de deslocamento na forma,

$$y(t) = G(q,\theta)u(t) + H(q,\theta)e(t), \tag{1.17}$$

$$G(q,\theta) = \frac{B(q)}{A(q)} \qquad e \qquad H(q,\theta) = \frac{1}{A(q)}$$

$$\theta = [a_1 \quad a_2 \quad \ldots \quad a_n \quad b_1 \quad b_2 \quad \ldots \quad b_m]^T$$

em que θ é o vetor de parâmetros da função de transferência.

Dois modelos discretos bastante utilizados em identificação de sistemas são apresentados a seguir:

Modelo ARX

Modelo auto-regressivo com entradas externas (ARX do inglês *autoregressive with exogenous inputs*) representado pela figura (1.1) é obtida a partir do modelo geral da equação (1.17), onde $A(q)$ e $B(q)$ são polinômios arbitrários. Por outro lado, quando for considerado o polinômio $B(q)$ igual a um o modelo passa a denominar-se auto-regressivo (AR).

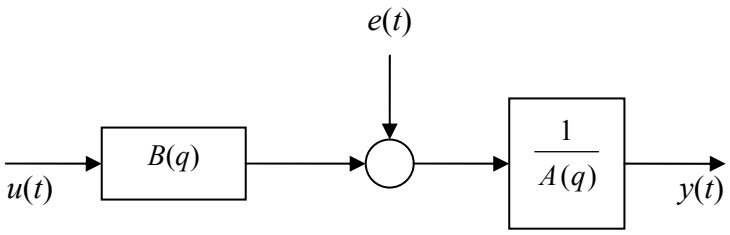

Figura 1.1 – Representação esquemática de modelo ARX

Modelo ARMAX

O modelo ARMAX, (*autoregressive moving average with exogenous inputs*) auto-regressivo com média móvel e entradas exógenas (externas), é ilustrado na figura (1.2). Pertence também a classe de modelo de erro na equação assim como o modelo ARX. A diferença entre os modelos esta na alteração do ruído. De um ruído branco com média zero (ARX), para um ruído colorido de média móvel (ARMAX). O modelo é representado pela equação (1.18) sendo considerados os polinômios $A(q)$, $B(q)$ e $C(q)$ arbitrários.

$$y(t) + a_1 y(t-1) + \ldots + a_n y(t-n) = b_1 u(t-1) + \ldots + b_m u(t-m) + e(t) + c_1 e(t-1) + \ldots + c_p e(t-p)$$

$$A(q)y(t) = B(q)u(t) + C(q)e(t) \tag{1.18}$$

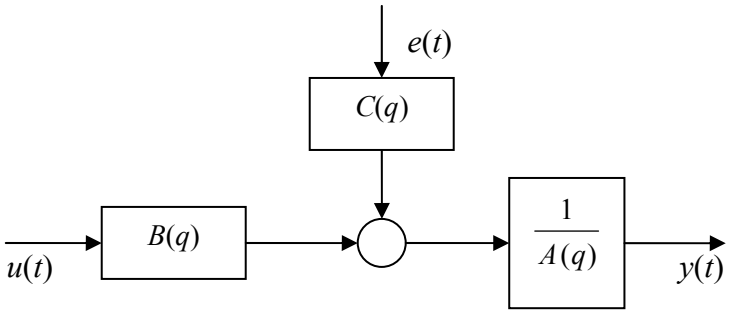

Figura 1.2 – Representação esquemática de modelo ARMAX

A partir dos modelos apresentados pode-se apresentar o modelo geral conforme equação (1.19). Na tabela 1.1 estão as diversas configurações e seus respectivos nomes (Ljung, 1997).

$$A(q)y(t) = \frac{B(q)}{F(q)}u(t) + \frac{C(q)}{D(q)}e(t) \tag{1.19}$$

Tabela 1.1 – Tipos de estruturas de modelos SISO

Polinômio usado em (1.19)	Nome da estrutura do modelo
B	FIR (*finite impulse response*)
AB	ARX
ABC	ARMAX
AC	ARMA
ABD	ARARX
ABCD	ARARMAX
BF	OE (*output error*)
BFCD	BJ (*Box-Jenkis*)

1.3.2 Representações não lineares

Vários campos da ciência utilizam-se da identificação de sistemas não lineares para descrever e analisar características de sistemas reais, podendo assim estabelecer relações entre o passado e o futuro.

Um importante passo na modelagem de sistemas é escolher uma estrutura adequada que represente o comportamento dinâmico do sistema. Algumas representações utilizadas na modelagem de sistemas não lineares são: Redes Neurais; Funções de Base Radial; Séries de Volterra; "Wavelets"; Funções Polinomiais e Racionais; Equações Diferenciais e Polinomiais (Aguirre, 1998).

A classe de modelos NARX (nonlinear autorregressive model with exogenous variables) são discretos no tempo e explicam o valor da saída y(k) em função de valores prévios dos sinais de saída e de entrada, podendo ser escrito matematicamente da seguinte maneira:

$$y(k) = F[y(k-1),...,y(k-n_y),u(k-d),...,u(k-n_u)] \tag{1.20}$$

onde n_y, n_u são os maiores atrasos em *y*, e *u* respectivamente e d representa o tempo morto.

Para que não haja uma polarização dos parâmetros são incluídos ruídos, e quando isso acontece o modelo passa a ser um modelo NARMAX (non-linear auto regressive with moving average and exogenous inputs). São modelos cuja estrutura é não linear auto-regressivo com média móvel e entrada externa.

A estrutura de um modelo NARMAX monovariável com período de amostragem normalizado é:

$$\begin{aligned} y(t) = F^l(&y(t-1),y(t-2),...,y(t-n_y),u(t-d),u(t-d-1),...,u(t-d-n_u+1),e(t-1),\\ &e(t-2),...e(t-n_e)) + e(t), \end{aligned} \tag{1.21}$$

onde t = 1,...,N, F^l é uma função não linear qualquer. y(t), u(t), e(t) são saída, entrada e ruído aditivo do sistema, cujos atrasos máximos são representados por n_y, n_u, n_e. Neste sistema o retardo ou tempo morto é representado pela letra d.

Na identificação de sistemas a função *F* normalmente não é conhecida. Para que a dinâmica do sistema seja reconstruída utiliza-se uma aproximação de *F*. Uma aproximação polinomial de grau 1 para o modelo anterior pode ser escrito da seguinte maneira:

$$y(t) = \theta_0 + \sum_{i_1}^{n} \theta_{i_1} x_{i_1}(t) + \sum_{i_1=1}^{n}\sum_{i_2=1}^{n} \theta_{i_1 i_2} x_{i_1}(t).x_{i_2}(t) + ... + \sum_{i_1=1}^{n}...\sum_{i_1=i_{l-1}}^{n} \theta_{i_1...i_1} x_{i_1}(t)...x_{i_1}(t) + e(t), \tag{1.22}$$

onde:

$$x_1(t) = y(t-1), x_2(t) = y(t-2), ..., x_{n_y+1}(t) = u(t-d), ..., x_{n_y+n_u+1}(t) = e(t-1), ..., x_n(t) = e(t-n_e).$$
$$n = n_y + n_u + n_e$$

Para ajustar a estrutura aos dados de identificação os parâmetros θ_i devem ser estimados.

A representação polinomial facilita a obtenção de informações analítica sobre a dinâmica do modelo a ser obtida; também ajusta os dados com uma maior exatidão desde que eles não apresentem variações abruptas (Aguirre, 2004).

Modelos que apresentam não linearidade fraca (lineares nos parâmetros) requerem um número maior de parâmetros do que os modelos com não linearidade forte (modelos que são não lineares nos parâmetros). O que determina o número de parâmetros de um modelo linear nos parâmetros é o tipo de regressores incluídos no modelo. Se forem empregados corretamente os regressores um modelo linear nos parâmetros, com poucos coeficientes poderá modelar a dinâmica não linear satisfatoriamente (Aguirre et all, 1998).

Segundo (Aguirre et all, 1998) uma limitação de modelos polinomiais lineares nos parâmetros é observada na modelagem de sistemas com não linearidade estática, que não possa ser adequadamente aproximada por polinômios de baixa ordem.

1.4 Determinação da estrutura do modelo

Para uma boa estimação de parâmetros faz-se necessária uma escolha correta da ordem do sistema, caso contrário, o modelo pode não representar a complexidade estrutural do sistema, bem como torná-lo mal condicionado.

Os métodos de determinação da estrutura para sistemas monovariáveis consistem na obtenção das ordens das equações a diferenças que descrevem a dinâmica do sistema. Isto corresponde à escolha do número de parâmetros a serem estimados. A utilização de modelos subparametrizados (escolha de um número de parâmetros menor que o real), pode muitas vezes omitir características consideradas fundamentais aumentando o grau de incerteza. Por outro lado modelos sobreparametrizados (escolha de um número de parâmetros maior que o real) provoca um aumento do esforço computacional e maior imprecisão na estimação de

parâmetros. A ordem do sistema baseia-se na análise do grau de dependência estatística entre os sinais de entrada, saída e perturbação.

Portanto, para determinar o número de parâmetros a ser utilizado deve-se fazer um estudo aprofundado de cada caso, pois, a complexidade ou simplicidade do modelo esta diretamente relacionada com o esforço computacional.

1.4.1 Seleção de estrutura de modelos não lineares

Modelos não lineares carregam um grande número de termos, o que dificulta em muito a definição da ordem de inclusão destes termos candidatos no modelo, mesmo para valores relativamente baixos da ordem.

Para uma correta seleção ordenada de termos tem-se a opção do cálculo da taxa de redução de erro. Baseado em um modelo NARMAX pode-se definir a seguinte taxa de redução de erro: $[ERR]_i = \dfrac{\hat{g}_i^{\,2}\langle w_i, w_i\rangle}{\langle y, y\rangle}$, onde w_i são regressores ortogonais entre si e y representa as saídas. Conforme (Aguirre, 2000), para utilizar esta equação faz-se necessária a estimação dos parâmetros $\hat{g}_i$, sendo este fato um inconveniente, já que se deseja escolher os regressores antes de estimar os parâmetros.

1.4.2 Critérios de informação

Em sistemas dinâmicos não-lineares pode-se estimar a ordem do modelo a partir de dados medidos. Os métodos mais empregados são: critérios de informação de Akaike ou AIC Akaike, 1974, Bayes ou BIC Kashyap, 1977, Schwarz Crutchfield e McNamara, 1987, entropia do modelo Mees, 1993 e o critério de Rissanen, 1978.

Atualmente o critério de informação mais empregado é o critério de Akaike. De acordo com este critério o número ótimo de termos deve minimizar a seguinte função de custo:

$$J = N\log(\mathrm{var}\{\xi(t)\}) + 2n_p \qquad (1.23)$$

onde N é o comprimento dos dados medidos, n_p é o número de termos de processo no modelo e ξ(t) é o vetor de resíduos. Esta função estabelece um compromisso entre a qualidade do ajuste aos dados de identificação e a procura por representações parcimoniosas.

Outro procedimento encontrado na literatura é iniciar a identificação com uma estrutura "superdimensionada" da rede, ou seja, com um grande número de neurônios na camada intermediária e após a etapa de treinamento, realizar a poda das redundâncias e conexões desnecessárias. Os métodos de poda mais usados em redes neurais são chamados de OBS (Optimal Braian Surgeon) e OBD (Optimal Braian Damage). Ambos os métodos utilizam uma medida da "saliência" de um peso através da estimativa da segunda derivada do erro de saída da rede com relação à aquele peso. Perturba-se o peso em estudo e mede-se em seguida a mudança no erro. A poda é realizada de modo iterativo, isto é: treina-se a rede para um nível de erro razoável, calcula-se a saliência, apagam-se os erros com saliências pequenas e treina-se a rede novamente. O método da poda pode ser aplicado a redes estática e dinâmica.

Aguirre (1994) discute a utilização de critérios de informação de sistemas não-lineares e ressalta que tal uso pressupõe que existe uma ordenação hierárquica no conjunto de termos candidatos. Em se tratando de sistemas não-lineares esta hierarquia pode ser determinada através do critério de ERR.

1.4.2.1 Detecção de estruturas de modelos polinomiais NARMAX

O aumento do grau da não linearidade l, dos atrasos máximos n_y, n_u, n_e aumenta consideravelmente o número de termos possíveis em modelos polinomiais. Para modelos monovariáveis o número de termos pode ser determinado pela seguinte expressão:

$$n_\theta = M + 1 \tag{1.24}$$

onde n_θ é o número total de termos no modelo e

$$M = \sum_{i=1}^{l} n_i, \qquad n_i = \frac{n_{i-1}(n_y + n_u + n_e + i - 1)}{i}, \; n_0 = 1$$

A seleção de termos a serem incluídos em um modelo é denominada de detecção de estrutura. A qualidade do modelo gerado esta diretamente ligada com a estrutura compatível com as não linearidades encontradas nos dados de entrada e de saída.

1.5 Estimação de parâmetros

Após a seleção da quantidade de parâmetros, faz-se necessária a correta estimação. Para tal, o modelo que descreve a relação entre a entrada e a saída do sistema é parametrizado

por um vetor θ. O problema de estimação consiste em estimar θ a partir de um conjunto de medidas $u = \{u_1, u_2, ..., u_n\}$ e $y = \{y_1, y_2, ..., y_n\}$, entrada e saída respectivamente.

Existem vários métodos de estimação de parâmetros, sendo alguns determinísticos e outros estocásticos. Pode-se citar o método dos mínimos quadrados, máxima verossimilhança e filtro de Kalman (Ljung, 1997), redes neurais artificiais entre outros.

Modelos NARMAX polinomiais são estruturas lineares nos parâmetros podendo ser estimados através do algoritmo de mínimos quadrados lineares (Billings e Voon, 1984; Chen et al., 1989; Zhu e Billings, 1996). A equação pode ser representada na forma de erro de predição conforme equação a seguir.

$$Y(t) = \sum_{i=1}^{n\theta} p_i(t)\theta_i + \xi(t,\theta) \tag{1.25}$$

Os termos p_i representam os regressores do modelo e os termos θ_i são os respectivos parâmetros. O resíduo de identificação $\xi(t,\theta)$ é definido como:

$$\xi(t,\theta) = y(t) - \hat{y}(t,\theta) \tag{1.26}$$

$$\hat{y}(t,\theta) = \sum_{i=1}^{n\theta} p_i(t)\hat{\theta}_i \tag{1.27}$$

Os erros de modelagem, o ruído aditivo e as incertezas são representados pelo vetor de resíduos $\{\xi(t,\theta) = 1,...,N\}$.

Conforme (Davis & Vinter, 1985) a estimativa obtida é dita não polarizada se os resíduos forem brancos e não apresentarem correlação com os regressores. Quando isso não é verificado, os resíduos apresentam alguma dinâmica que não foi devidamente explicada pelo modelo para que as estimativas se tornem não polarizadas e toda a dinâmica dos dados seja absorvida pelo modelo.

Segundo Luo et *al* (1994) os algoritmos de estimação de parâmetros podem ser feitos em batelada (off-line) ou recursiva (on-line). Os algoritmos (on-line) são considerados mais complexos em relação aos algoritmos (off-line).

Conforme Pröl & Karim (1994), na prática a detecção de estrutura de modelos não lineares é feita de maneira off-line enquanto que a avaliação de parâmetros pode ser feita on-line.

1.6 Validação do modelo

Um passo importante na identificação de sistemas é a verificação da validade do modelo obtido, ou então na opção de mais modelos é necessário escolher o modelo que melhor incorpora as características requeridas. Para validar um modelo é importante considerar um conjunto de critérios:

Simulação: A simulação consiste em comparar os dados obtidos pelo modelo com os dados do sistema, com o objetivo de observar se o comportamento do modelo reproduz as características do sistema. É importante que os dados utilizados para obter o modelo não sejam usados para a validação com o objetivo de verificar se o modelo tem capacidade de explicar um novo conjunto de dados do mesmo sistema, ou seja, se ele tem capacidade de generalização.

Após a simulação dos dados geralmente são observados os erros, e para que se verifiquem onde os mesmos ocorreram aplica-se o procedimento de analise de ruídos. Avaliando se as falhas ocorreram na etapa de estimação dos parâmetros, ou se a dinâmica que produziu os dados não pode ser satisfatoriamente representada pelo modelo, facilitando desta forma a correção a ser feita no modelo.

Na validação de modelos com dinâmica caótica empregam-se os invariantes dinâmicos, pois nestes tipos de modelos uma variação infinitésima nas condições iniciais, ou nos parâmetros resultará em divergências entre a solução do modelo e os dados do sistema. Uma maneira de contornar esta situação esta na reconstrução do espaço de estados, pois esta reconstrução não requer o tempo explicitamente representado (Aguirre, 2004).

Para a validação de sistemas lineares verifica-se se os resíduos são brancos e não correlacionados com a entrada. Esta verificação pode ser feita calculando-se as funções de autocorrelação dos resíduos e correlação cruzada dos resíduos com a entrada (Aguirre, Rodrigues e Jácome 1998).

Sistemas não-lineares exigem verificar se os resíduos de um determinado modelo são linearmente brancos quanto à entrada e não-linearmente branco quanto à saída, e ainda, não podem ser autocorrelacionados. Para tanto é necessário avaliar se existem correlações estatisticamente significativas nos resíduos, o que é possível através das seguintes funções de correlação. Se tais identidades podem ser verificadas então aceita-se a hipótese de que os

resíduos de modelagem são brancos. Os referidos testes são Billings e Voon, (1986); Billings e Tão, (1991):

$$\Phi_{\xi\xi}(\tau) = E\{\xi(t)\xi(t-\tau)\} = \delta(\tau)\,, \cdot \tag{1.28}$$

$$\Phi_{\xi u}(\tau) = E\{\xi(t)u(t-\tau)\} = 0 \forall \tau\,,, \tag{1.29}$$

$$\Phi_{\xi\xi u}(\tau) = E\{\xi(t)\xi(t-\tau)u(t-\tau)\} = 0 \forall \tau\,,, \tag{1.30}$$

$$\Phi_{u^2\xi}(\tau) = E\{u^2(t) - E\{u^2(t)\}\xi(t-\tau)\} = 0 \forall \tau\,,, \tag{1.31}$$

$$\Phi_{u^2\xi^2}(\tau) = E\{u^2(t) - E\{u^2(t)\}\xi^2(t-\tau)\} = 0 \forall \tau \tag{1.32}$$

$$\Phi_{(y\xi)u^2}(\tau_1) = E\{(y(t)\xi(t) - E\{y(t)\xi(t)\}(u^2(t-\tau_1) - E\{u^2(t\})\} = 0, \forall \tau_1 \tag{1.33}$$

onde E é a soma do erro médio quadrático da(s) saída(s), u(t) é a entrada do sistema, *ξ(t)* é o resíduo, $\delta(\tau)$ é a função delta de Dirac e Φ é o fator de aceleração do coeficiente de aprendizado.

Possíveis correlações não lineares existentes nos ruídos podem ser detectadas através das equações apresentadas acima. Se as correlações não forem detectadas, os ruídos são brancos. Esta avaliação das correlações é denominada de validação estatística de um modelo dinâmico, e garante apenas a não existência de correlações não modeladas nos resíduos de identificação.

Segundo Aguirre e Billings (1994), um modelo estatisticamente válido pode não reproduzir uma ou mais propriedades dinâmicas do sistema original. Por esse motivo, um procedimento de validação dinâmica deve estar sempre associado com a validação estatística. A validação dinâmica deverá verificar se o modelo identificado reproduz as principais características dinâmicas originais.

Para validar modelos são empregadas algumas propriedades dinâmicas apresentadas por Aguirre e Billing, (1994); Fiedler-Ferrara & Prado, (1994); Letellier e Gouesbet, (1995) listadas a seguir:

(i) expoentes de Lyapunov;
(ii) mapas e seções de poincaré;
(iii) dimensão de correlação
(iv) diagramas de bifurcação

(v) características topológicas de atratores reconstruídos

(vi) constantes de tempo, ganhos e características estáticas

2 REDES NEURAIS ARTIFICIAIS

2.1 Introdução

As pesquisas na tentativa de imitar o cérebro humano através de um computador apresentaram seus primeiros resultados na década de 40, com seus precursores o neurofisiologista McCulloch e o matemático Walter Pitts, desenvolvendo um modelo matemático de neurônio encontrado em organismos vivos. Este modelo foi ampliado posteriormente por Rosenblatt (1960) & Widrow & Hoff (1960) os quais implementaram uma regra de aprendizado baseada na minimização do erro quadrático o qual foi usado para formular o Adaline (Adaptative linear element). Posteriormente, Widrow e seus estudantes implementaram uma das primeiras redes neurais em camadas treináveis com múltiplos elementos adaptativos denominada Madaline (Multiple-Adaline) (Haikin, 2001).

Amari (1967) utilizou o método do gradiente estocástico para classificação adaptativa de padrões. Minsky & Papert (1969) publicaram um artigo denominado "perceptron" no qual questionam a capacidade adaptativa do neurônio artificial. Demonstraram matematicamente os limites existentes nos cálculos dos perceptrons de camada única, e afirmaram ainda, que a versão de múltiplas camadas não iria superar estas limitações, diminuindo de uma maneira brusca as pesquisas neste setor. Nesta época a máquina de Turing era considerada a mais poderosa máquina inteligente baseada em princípios que hoje denomina-se conexionista. Como Minsky & Papert eram considerados importantes e renomados pesquisadores, grande parte das pesquisas foram abandonadas e ou seus estudos omitidos (sem publicá-los).

Em meados dos anos 80 ocorreu a retomada da injeção de dinheiro para os projetos de pesquisa nesta área e com a computação mais avançada permitindo a implementação de poderosos algoritmos de aprendizagem. Neurofisiologistas e biofísicos tornaram a publicar suas descobertas. Hopfield (1982) publicou seu trabalho sobre redes neurais totalmente recorrentes utilizando a idéia de uma função de energia. Demonstrando assim o

armazenamento destas redes e Kohonen (1982) publicou um modelo de mapas auto-organizáveis surpreendendo a comunidade científica que voltou fervorosamente a pesquisar o assunto, e tomou este artigo como base para avaliações das inovações nesta área.

Em meados do ano de 1983 foi desenvolvida a máquina de Bolzmann, sendo esta a primeira aplicação bem sucedida de uma rede neural de múltiplas camadas. Contribuição valorosa foi dada por Rumelhart & McClelland (1983), Hinton & Williams (1986) que apresentaram o algoritmo de retropropagação (back-propagation) necessário para o treinamento de redes neurais múlticamadas. Werbos (1974) já havia aplicado este algoritmo, porém em outro contexto.

Broomhead & Lowe (1988) apresentaram as funções de base radial como uma alternativa aos perceptrons de múltiplas camadas. Estas funções são baseadas no método das funções de potencial que foi proposto originalmente por Bashkirov, Braverman & Muchnik (1964). Outras técnicas computacionais empregadas como algoritmos genéticos e computação evolutiva (Iyoda, 2000) e estratégias não paramétricas de evolução (Von Zuben, 1996) colaboraram para o aumento da eficiência dos algoritmos de treinamento.

O estudo da inteligência Artificial (IA) gerou dois paradigmas. O primeiro denominado simbolista defende a resolução de problemas através de etapas pré-definidas denominadas algoritmos. O segundo conexionista defende a impossibilidade de transcrever o potencial do cérebro humano através de passos finitos lógicos.

Nos modelos simbólicos os sentidos diferentes das palavras são representados independentemente, ou seja, recebem identificadores atômicos que não tem nenhuma relação superficial entre si como, por exemplo, carro e porsche (Rich & Knight, 1993). São usados elos para a descrição dos relacionamentos entre os conceitos. Quando os relacionamentos ficam difusos o sistema tem dificuldades em fazer comparações.

Os computadores são máquinas muito sofisticadas capazes de armazenar grandes quantidades de informações, realizarem atividades com alto grau de precisão e velocidade. Enquanto o cérebro humano é programado para atividades ao mesmo tempo mais simples, como o ato de andar, falar e raciocinar, mas ao mesmo tempo mais complexas como aprender novas tarefas, interpretar, compreender e habituar-se a novos locais e situações. Outra atividade desenvolvida muito bem pelo cérebro é enfrentar situações problema procurando

para a sua resolução encontrar em sua memória resolução de situações parecidas. Sendo o conexionismo uma aproximação.

A grande maioria das representações computacionais atuais é baseada nos procedimentos conexionistas como citado acima, embora sua evolução tenha sido mais lenta em relação ao simbolismo. O cérebro humano é formado por um grande número de elementos processadores chamados de neurônios que atuam em paralelo (Rich & Knight, 1993). Isto sugere empregar algoritmos em paralelo. Sendo que cada camada deve possuir a quantidade de neurônios conforme a necessidade do problema.

Neste capitulo será feita uma breve abordagem histórica de redes neurais, seguindo com a fundamentação biológica necessária para compreender o funcionamento de um neurônio. Também serão abordadas algumas redes neurais juntamente com seus algoritmos de aprendizagem.

2.2 Fundamentos biológicos

Os sistemas de inteligência artificial foram desenvolvidos a partir de um conjunto de estudos na tentativa de realizar atividades complexas até então observadas em cérebros de organismos vivos, em especial os humanos. Segundo Campos & Saito (2004), seria perigoso estabelecer um paralelismo rigoroso entre redes neurais e o comportamento do cérebro humano, pois, criar-se-iam “humanos artificiais”.

Biologicamente, possuem cérebro, apenas os organismos vivos capazes de realizar atividades motoras, explicando o fato que seria pouco provável que houvesse um deslocamento eficiente sem uma primeira análise do ambiente. Por isso, segundo estudos da robótica, é mais complicado programar máquinas que se desloquem em ambientes complexos do que fazer programas capazes de vencer os melhores jogadores de xadrez do mundo (Campos e Saito 2004).

Estima-se que o número de células nervosas (de neurônios) concentrados no cérebro humano gira em torno de cem bilhões. São capazes de receber sinais de vários outros neurônios vizinhos e processa-los de forma a chegar a uma conclusão, gerar e conduzir um sinal elétrico, transmitindo-o para um outro neurônio. A figura (2.1) mostra a estrutura de células nervosas humanas.

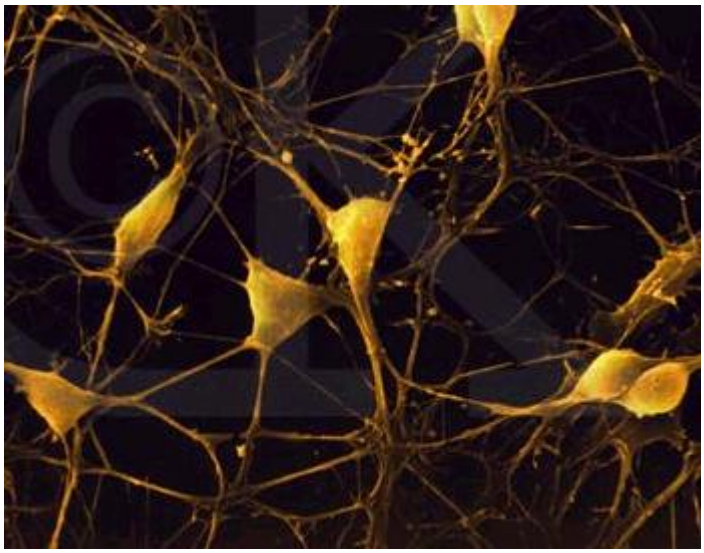

Figura 2.1 - Estrutura de células nervosas humanas

Um neurônio é constituído de três partes: corpo celular, dendritos e axônios. O primeiro realiza as atividades bioquímicas, alimentando o neurônio. Os dendritos são vários prolongamentos receptores de sinais. E o axônio é uma fibra nervosa que transporta os sinais emitidos pelo neurônio é caracterizado por uma alta resistência elétrica e uma capacitância muito grande. Os neurônios aparecem em uma grande variedade de formas e tamanhos em diferentes partes do cérebro (Haikin, 2001).

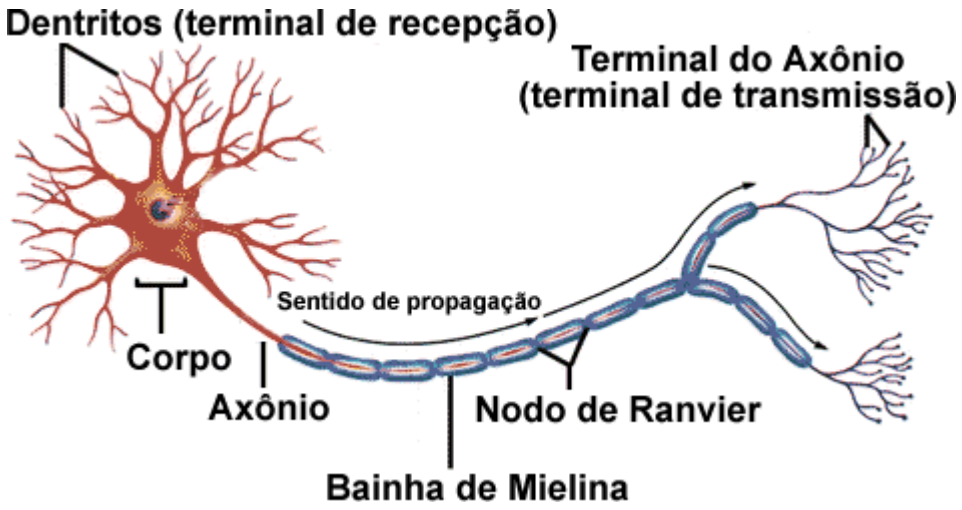

Figura 2.2 Esquema de um neurônio humano

Os sinais elétricos entre os neurônios são disparados apenas se o sinal combinado de todos os dendritos pelo neurônio é grande o bastante, ou seja, se ultrapassa certo limiar (Ferreira, 2003). Esta transmissão de sinais é chamada de sinapse, que consta da liberação de substâncias químicas chamadas de neurotransmissores (situados nas extremidades dos axônios) através de sinais elétricos. Essas substâncias espalham-se no espaço que existe entre os neurônios.

A transmissão de sinais (eventos) ocorre na ordem de milissegundos (10^{-3} s). Comparados com eventos em circuito de silício que ocorre na ordem de nanossegundos (10^{-9} s), os eventos do cérebro humano são lentos. Mas, por outro lado, a lentidão é compensada com o grande número de neurônios e suas conexões.

A neurobiologia apóia-se no principio da existência de 10^{15} (um milhão de bilhões) de conexões entre as células nervosas, sendo que a quantidade de 100.000 genes existentes no corpo humano não é suficiente para estabelecer todas as conexões. O cérebro então cria um número muito grande de conexões à medida que os neurônios procuram seus alvos que seriam predeterminados geneticamente. Porém, à medida que o cérebro é estimulado, se desenvolve e aprende ele vai eliminando as conexões desnecessárias, mantendo apenas as conexões úteis. (Campos & Saito, 2004).

Os conceitos acima especificados induziram ao desenvolvimento das redes neurais como um bloco elementar da inteligência artificial.

2.3 Neurônio Artificial

O neurônio artificial é um elemento que processa através de uma função geralmente não linear o somatório dos dados de entrada pelos pesos específicos, gerando uma saída. Conforme representação a seguir:

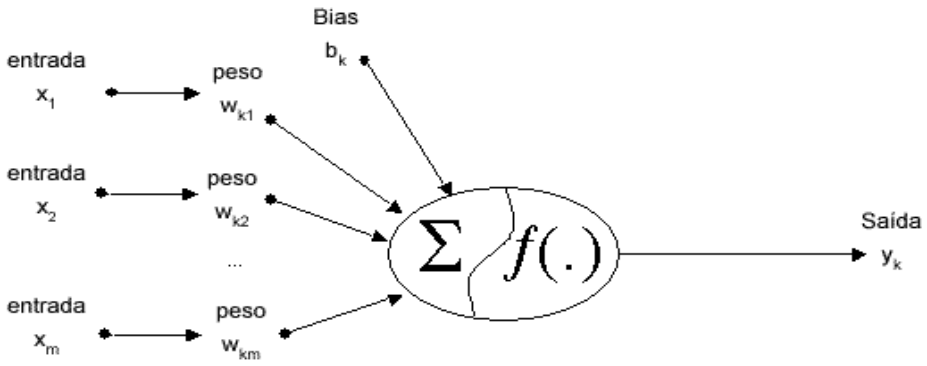

Figura 2.3 - Representação de um neurônio artificial

onde $f(.)$ é a função de ativação e o $\sum$ é a função de soma

Em termos matemáticos, um neurônio pode ser descrito da seguinte forma:

$$v_k = \sum_{j=1}^{m} w_{kj} x_j \tag{2.1}$$

$$y(k) = f(v_k + b_k) \tag{2.2}$$

onde x_1, x_2, …, x_m são os sinais de entrada; w_{k1}, w_{k2}, …, w_{km} são os pesos sinápticos do neurônio k; v_k é a saída do combinador linear das entradas pelos seus pesos; b_k é o bias; $f(.)$ é a função de ativação e y_k é o sinal de saída do neurônio.

A primeira etapa do processamento resulta da multiplicação dos dados de entrada (x_j) pelos seus respectivos pesos (w_{kj}). A ordem da multiplicação vem do emprego correto dos índices utilizados. "O índice *k* simboliza o neurônio em questão e o índice *j* se refere a entrada. Os pesos de um neurônio determinam a força dele, e podem assumir valores positivos ou negativos.

O somatório $\sum$ é responsável por somar os sinais de entrada ponderados pelos seus respectivos pesos. Ao somatório muitas vezes é acrescido um bias representado por b_k que tem efeito de aumentar ou diminuir a entrada líquida da função de ativação.

2.3.1 Funções de ativação

A função de ativação $f(.)$ tem como objetivo restringir a amplitude da saída de um neurônio e é geralmente não-linear. As funções usualmente empregadas para representar as não-linearidades são descritas a seguir.

2.3.1.1 Função limiar (Degrau)

Descreve a propriedade do tudo ou nada encontrada no modelo de Mc Culloch-Pitts onde a saída de um neurônio assume o valor 1 (um) se a soma ponderada dos sinais de entrada de um neurônio ultrapassar a um determinado limiar a (valor positivo). Caso o valor de ativação assumir valor negativo a saída assume valor zero. É necessário levar em conta que o valor de ativação é composto pelo combinador linear acrescido de um bias (Huamaní, 2003).

$$f(v) = \begin{cases} 1, & \text{se } v \geq 0 \\ 0, & \text{se } v \leq 0 \end{cases} \tag{2.3}$$

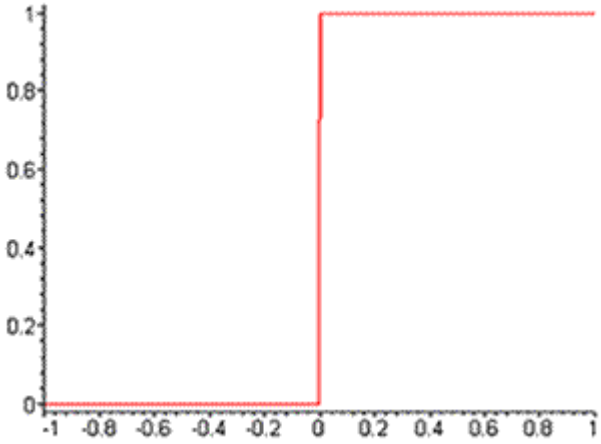

Figura 2.4 - Gráfico da Função de Limiar

2.3.1.2 Função sigmóide

Esta função permite que o valor de saída do neurônio assuma valores em um intervalo contínuo de 0 a 1. É uma função de grande importância para a construção de uma rede neural devido a sua capacidade de diferenciabilidade. Um exemplo de função sigmóide é a função logística, definida por:

$$\varphi(v) = \frac{1}{1 + \exp^{(-av)}} \tag{2.4}$$

onde v é a ativação e a é o parâmetro de inclinação da função sigmóide .

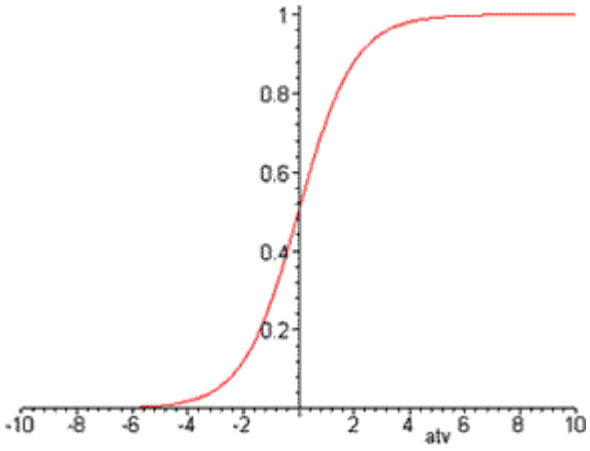

Figura 2.5 - Gráfico da Função Sigmóide

Quando aumenta-se o valor do parâmetro *a*, tendendo-o ao infinito, esta função comporta-se como uma função de limiar, como pode-se observar no gráfico a seguir:

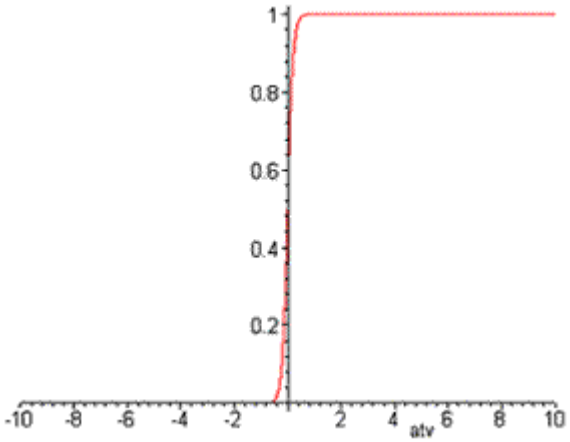

Figura 2.6 – Gráfico da Função Sigmóide - com a *tendendo ao infinito*

Substituindo-se av por z pode-se reescrever a função sigmóide gerando a seguinte equação

$$\varphi(z) = \frac{1}{1 + \exp^{(-z)}} \tag{2.5}$$

Expandindo a Equação (2.5) em uma série de potências obtém-se a equação (2.6)

$$\frac{1}{1+e^{-z}} = 0.5 - 0.25z + 0.125\frac{z^3}{3!} - 0.25\frac{z^5}{5!} + ... \tag{2.6}$$

considerando que

$$z = \sum_{l=1}^{n} w_{jl} x_l + w_{j0} \tag{2.7}$$

Substituindo a Equação (2.7) na Equação (2.6) e considerando uma rede MLP com um vetor de entrada composto por X=[1 y(k-1) y(k-2) u(k-1)]^T, com apenas um neurônio na camada escondida, e uma saída. Sua saída pode ser escrita como:

$$\hat{y}(k) = F(w_{ij} f_j(\sum_{l=1}^{n} w_{jl} x_l + w_{j0}) + w_{i0})$$

$$\hat{y}(k) = \begin{pmatrix} a_0 + a_1 y(k-1) + a_2 y(k-2) + a_3 u(k-1) + a_4 y(k-1) y(k-2) + \\ a_5 y(k-1) u(k-1) + a_6 y(k-2)^2 + a_7 y(k-2)\ u(k-1) + a_8 y(k-1) y(k-2) u(k-1) + \\ a_9 (k-1)^2 + a_{10} u(k-1)^3 + a_{11} y(k-1)^2 u(k-1) + ... \end{pmatrix} \tag{2.8}$$

Com $a_{0,1,2,...n}$ sendo considerados os pesos w da rede

2.3.1.3 Função Signum:

Esta função apresenta as mesmas características da função de limiar, porém está limitada entre 1 e -1. É representada por:

$$f(v) = \begin{Bmatrix} 1\ se\ \ v > 0 \\ 0\ se\ \ v = 0 \\ -1\ se\ \ v < 0 \end{Bmatrix} \tag{2.9}$$

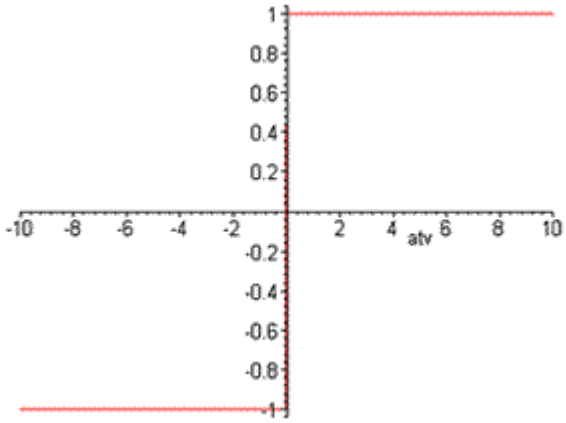

Figura 2.7 - Gráfico da Função Signun

2.3.1.4 Tangente Hiperbólica:

Como a Função Logística, também possui forma de "s", assumindo valores entre 1 e -1, sendo representada por:

$$\varphi(v) = a \frac{e^{(bv)} - e^{(-bv)}}{e^{(bv)} + e^{(-bv)}} \tag{2.10}$$

onde *a* é o parâmetro de inclinação da curva; *b* são os limites inferiores e superiores *(b = |1| no gráfico); v* é o valor de ativação.

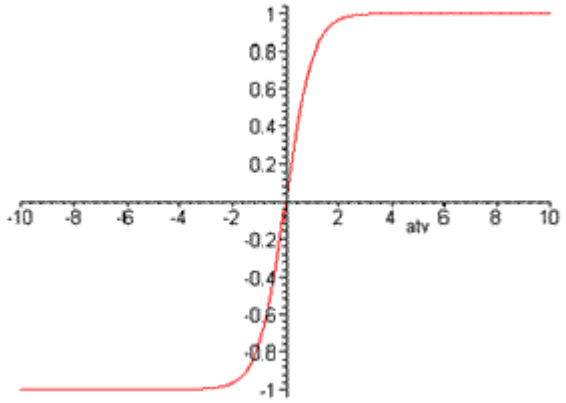

Figura 2.8 - Gráfico da Função Tangente Hiperbólica

Substituindo-se bv por z na Equação (2.10), pode-se reescrever a função tangente hiperbólica como a seguir.

$$\varphi(z) = a\,\frac{e^{(z)} - e^{(-z)}}{e^{(z)} + e^{(-z)}} \tag{2.11}$$

Expandindo a Equação (2.11) em uma série de potências obtém-se a equação (2.12).

$$\frac{e^{(z)} - e^{(-z)}}{e^{(z)} + e^{(-z)}} = \frac{z + \frac{z^3}{3!} + \frac{z^5}{5!} + \ldots}{1 + \frac{z^2}{2!} + \frac{z^4}{4!} + \ldots} \tag{2.12}$$

considerando que

$$z = \sum_{l=1}^{n} w_{jl} x_l + w_{j0} \tag{2.13}$$

Substituindo a Equação (2.12) na Equação (2.11) e considerando uma rede MLP com um vetor de entrada composto por X=[1 y(k-1) y(k-2) u(k-1)]^T, com apenas um neurônio na camada escondida, e uma saída, sua saída pode ser escrita como:

$$\hat{y}(k) = F(w_{ij} f_j(\sum_{l=1}^{n} w_{jl} x_l + w_{j0}) + w_{i0})$$

$$\hat{y}(k) = \frac{1}{D} \times \begin{pmatrix} a_0 + a_1 y(k-1) + a_2 y(k-2) + a_3 u(k-1) + a_4 y(k-1) y(k-2) + \\ a_5 y(k-1) u(k-1) + a_6 y(k-2)^2 + a_7 y(k-2) u(k-1) + \\ a_8 y(k-1) y(k-2) u(k-1) + a_9 u(k-1)^2 + a_{10} u(k-1)^3 + a_{11} y(k-1)^2 u(k-1) + \ldots \end{pmatrix} \tag{2.14}$$

Sendo

$$D = \begin{pmatrix} b_0 + b_1 y(k-1) + b_2 y(k-2) + b_3 u(k-1) + b_4 y(k-1)^2 + b_5 y(k-1) y(k-2) + \\ b_6 y(k-1) u(k-1) + b_7 y(k-2)^2 + b_8 y(k-2) u(k-1) + b_9 u(k-1)^2 \ldots \end{pmatrix} \tag{2.15}$$

onde as constantes a e b são funções dos pesos w da rede

A tabela a seguir traz a analogia entre um neurônio biológico e um neurônio artificial (Ferreira, 2003).

Tabela 2. 1 – Analogia entre um neurônio biológico e um artificial

Neurônio Biológico	Neurônio Artificial
Sinapse	Conexão de Peso
Axônio	Canal de Saída
Dendritos	Canal de Entrada
Soma (Derivada da palavra Sôma do grego que significa corpo celular)	Função de ativação

2.4 Generalidades sobre o processo de aprendizagem

A implementação de uma rede neural multicamadas segue etapas pré-definidas

- A definição de entradas e saídas da rede
- A determinação da arquitetura empregada
- Qual algoritmo de aprendizagem utilizar

O conjunto de dados de entrada e saída precisam ser suficientemente grandes para que a rede possa observar as características existentes entre as relações entrada versus saída, sendo assim representantes da dinâmica do processo.

O aprendizado de uma rede neural é um processo de adaptação dos pesos sinápticos e de níveis de bias através de impulsos gerados pelo ambiente na qual esta inserida, com o intuito de atingir objetivos relacionados com a função custo. A aprendizagem se dá de acordo com o algoritmo utilizado. A rede se torna mais instruída sobre o seu ambiente após cada iteração do processo de aprendizagem (Haikin, 2001). Existem algumas formas de aprendizagem, dentre as quais destacam-se: aprendizado supervisionado, aprendizagem não supervisionada, aprendizagem híbrida e aprendizado por reforço.

2.4.1 Aprendizado supervisionado

O aprendizado supervisionado conta com um "professor" que teoricamente supervisiona os resultados da rede. Como ele conhece o ambiente, verificará a relação entre a

entrada e a saída. Comparando a saída da rede com a saída esperada o professor trabalhará com esta diferença, de forma a tornar ótima a resposta. Nas redes compostas de múltiplas camadas, as camadas ocultas supostamente não são visíveis pelo professor, por este motivo na maioria dos casos trabalha-se apenas com o erro da saída.

2.4.2 Aprendizado não supervisionado

Nesta metodologia a rede recebe dados de entrada e reflete de forma independentemente sobre propriedades observadas na saída. Não havendo professor para supervisionar o processo de aprendizagem, a rede aprende a ajustar seus parâmetros baseados em tendências estatísticas observadas nos dados de entrada (Lima, 2000).

2.4.3 Aprendizagem híbrida

É caracterizada pelo ajuste dos pesos através da aprendizagem supervisionada e não supervisionada. Parte dos pesos é ajustada pela aprendizagem supervisionada e outra parte pela aprendizagem não supervisionada (Ferreira, 2003).

2.4.4 Aprendizado por reforço

Consiste no método da tentativa e erro de modo a otimizar um índice de desempenho denominado sinal de reforço. Neste método, não são fornecidas as saídas corretas a cada treinamento, mas sim, um valor que informa se a saída esta correta ou não. Deixando de forma um tanto quanto obscura o ajuste dos parâmetros em direção a minimização dos erros.

2.5 Arquiteturas de redes neurais artificiais

A definição da arquitetura da rede é baseada diretamente no algoritmo de aprendizagem e, consiste na escolha de quantas camadas serão empregadas e quantos neurônios irão compor cada camada. A opção por uma arquitetura muito simples pode subestimar o problema e levar a soluções “levianas”. Porém, arquiteturas mais complexas podem responder bem aos exemplos fornecidos, mas em muitos casos, não são capazes de assimilar novos exemplos. Portanto, a escolha da quantidade de camadas ocultas é complexa. O trabalho de Cybenko (1989) mostra em tese que com suficiente número de funções de base,

uma camada oculta seria suficiente para modelar a maioria dos sistemas práticos. Outros pesquisadores apresentaram posteriormente resultados semelhantes, validando tal tese, hoje aceita universalmente. É possível identificar três classes de arquiteturas de redes:

2.5.1 Redes alimentadas adiante com camada única

Nesta estrutura a camada de entrada é projetada sobre a camada de saída (neurônios processadores). Estas redes têm esta denominação, pois considera-se apenas os neurônios referentes a camada de saída. Não são contados os nós da camada de entrada porque não realizam nenhum processamento, servem apenas como fonte.

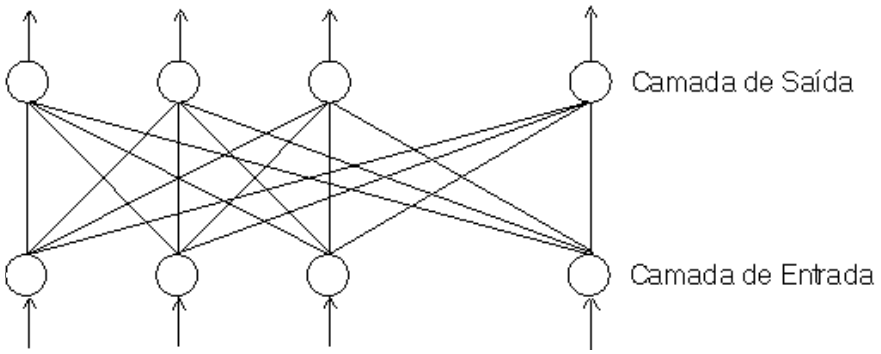

Figura 2.9 - Rede alimentada adiante ou acíclica com uma única camada de neurônios

2.5.2 Redes Alimentadas Adiante com múltiplas camadas

Esta classe de redes é caracterizada pela presença de uma ou mais camadas ocultas que por sua vez abrigam os nós computacionais denominados de neurônios ocultos (Haikin 2001). As camadas ocultas são adicionadas para tornar a rede mais potente no sentido de extrair dados de ordem estatística.

Os neurônios da camada de entrada fornecem os dados (entradas) que são projetados na primeira camada oculta processadora (segunda camada). Os dados de saída desta camada servem de entrada para a terceira camada e assim sucessivamente. A última camada fornece os dados de saída da rede. Uma rede é totalmente conectada quando todos os neurônios de uma camada estão conectados com todos os neurônios da camada seguinte. Caso contrário a rede é parcialmente conectada. A figura (2.9) ilustra uma rede acíclica totalmente conectada.

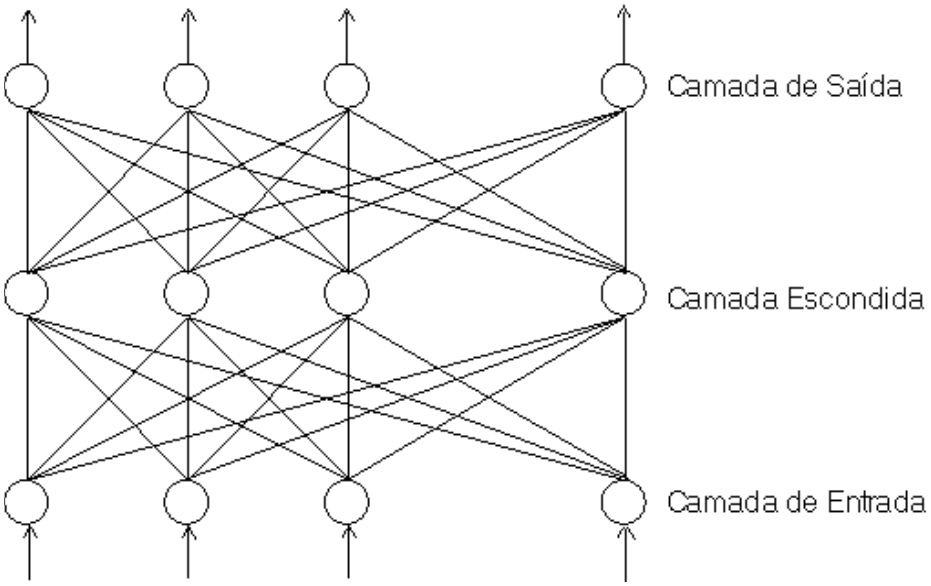

Figura 2.10 - Rede alimentada adiante com uma camada oculta e uma camada de saída

2.5.3 Redes Recorrentes

Uma rede neural recorrente é baseada no princípio da realimentação, a qual pode ser originada dos neurônios ocultos bem como dos neurônios de saída, e ainda, a saída de um neurônio pode alimentar uma ou mais entradas. Devido à recorrência, os níveis de ativação da rede formam um sistema dinâmico podendo atingir um estado estável, bem como apresentar um comportamento caótico. (Russel & Norvig 2004).

As redes neurais recorrentes podem admitir uma memória a curto prazo assemelhando-se ao cérebro humano o que as tornam interessantes; porém mais complexas em sua compreensão (Russel & Norvig 2004), como pode ser visto na figura 2.11.

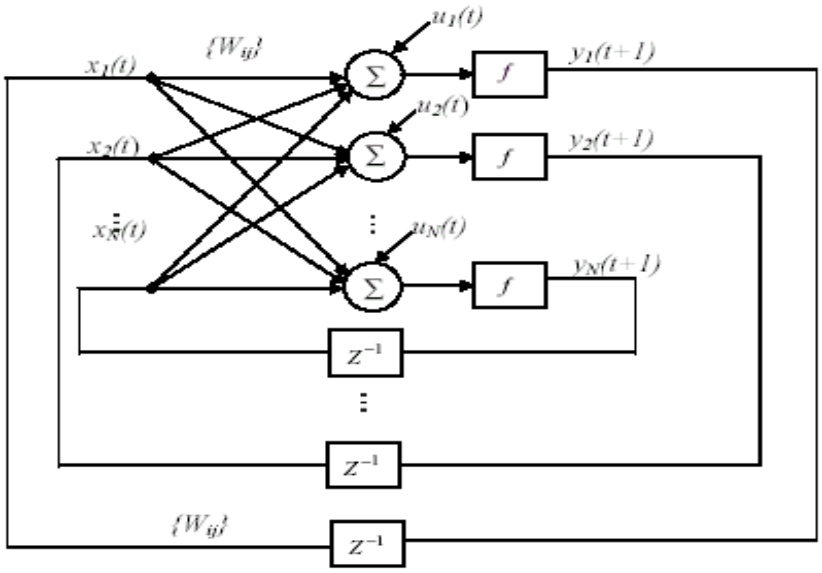

Figura 2.11 - Ilustração de uma rede neural totalmente recorrente

2.6 Exemplos de Redes Neurais

2.6.1 Rede Adaline (ADAptive Linear Element)

A rede Adaline utiliza-se de uma rede de associação linear. Porém o objetivo é obter um conjunto de pesos que minimize os erros entre as saídas obtidas y^r e as saídas desejadas y^i. Utilizando um conjunto de treinamento, definido por (x_i, y_i), os pesos w, a função custo é definida como:

$$f(w) = \min \sum_{i=1}^{N} (y_i - y_i{}^r)^2 \tag{2.16}$$

onde N é o tamanho do conjunto de dados

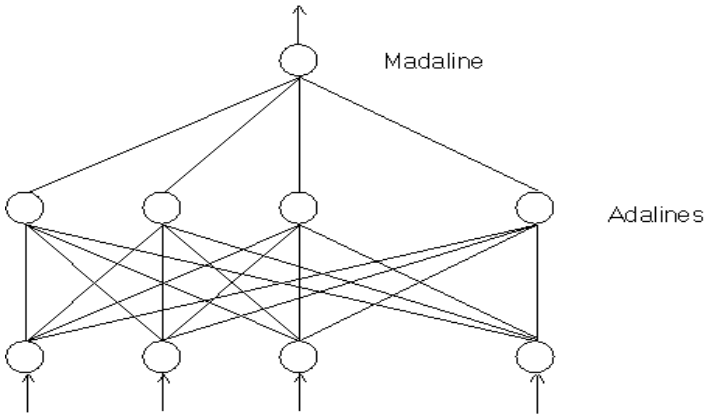

Figura 2.12 - Ilustração de uma rede Adaline

O algoritmo empregado para o ajuste de pesos (w) é o de mínimos quadrados objetivando caminhar na direção oposta do gradiente (Widrow e Stearns, 1985, apud Mario, 2004), como apresentado na equação (2.17)

$$\nabla f(w) = \nabla_w (\sum (y_i - {y_i}^r)^2$$

$$\nabla f(w) = \sum (\nabla ((y_i - {y_i}^r)^2))$$

$$\nabla f(w) = \sum (2(y_i - {y^r}_i) - \nabla ({y^r}_i))$$

Sendo considerado ${y^r}_i = wx$ então $\nabla ({y^r}_i) = x_i$, para um peso especifico $w(i)$.

$$\nabla f(w) = \sum (2(y_i - {y^r}_i)(-x_i))$$

$$\nabla f(w) = -2E((y_i - {y^r}_i)(-x_i)) \qquad (2.17)$$

Widrow em sua dedução substituiu então a esperança $E((y_i - {y^r}_i)(-x_i))$, pelo próprio valor pontual $2((y_i - {y^r}_i)(-x_i))$, expressando da seguinte maneira a lei de aprendizagem dos mínimos quadrados:

$$w_i^{novo} = w_i^{antigo} + \alpha(erro_i)(x_i) \qquad (2.18)$$

O parâmetro α controla a velocidade da aprendizagem e varia entre 0.001 e 10, quando muito pequeno produz uma convergência muito lenta. Quando muito grande pode produzir divergência.

2.6.2 Rede com lei de aprendizagem de Hebb

Hebb estudou o funcionamento do cérebro humano, em especial, os fenômenos cognitivos ligados à memória e o aprendizado e propõe uma teoria para a aprendizagem a nível celular. Considera que quando um neurônio emite um pulso devido a uma informação vindo de uma entrada, esta entrada aumenta a sua capacidade de produzir novos impulsos no futuro (como se ela aumentasse o seu peso) (Campos e Saito 2004). O aumento da conexão ocorre no nível da sinapse. Esta lei de aprendizagem advém da função de eventos através de recompensas e punições. Pode ser usada em uma rede de associação linear. Conforme figura (2.12), e representada pela equação (2.19).

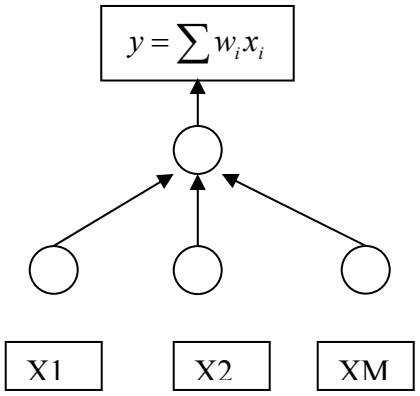

Figura 2.13 - Representação de um esquema de rede Hebbiana

$$w_i^{novo} = w_i^{antigo} + y_i x_i \tag{2.19}$$

Onde (y_i, x_i) são exemplos de entrada e saída, utilizados para a aprendizagem da rede. Os pesos (w_i) possuem seu valor inicial zero. Ao final do treinamento os pesos terão valor igual a $w_i = \sum (y_J x_j^T)$. Supondo que os valores de X_I são ortonormais ($\|X_I\| = 1$, e ortogonais $(X_I X_J = 1)$ se I = J ou 0 se I $\neq$ J), portanto toda vez que um vetor X_K for apresentado a esta rede a sua saída será Y_K. Tem-se então a fórmula (2.20) (Campos e Saito 2004)

$$(Y = WX_K = \sum (Y_J x^T{}_J) X_K = Y_K) \tag{2.20}$$

2.6.3 Rede de Kohonen "(Aprendizado não supervisionado)"

A rede de Kohonen é composta de vários elementos que atuam em paralelo. Cada unidade processadora tem ao seu alcance um conjunto de pesos (w_I) e uma função que calcula a distância entre os pesos e a entrada. A unidade que tiver os pesos mais próximos da entrada vence a competição e tem a sua saída ativada, enquanto que as saídas dos outros neurônios ficam desativadas (Kováscs, 1997).

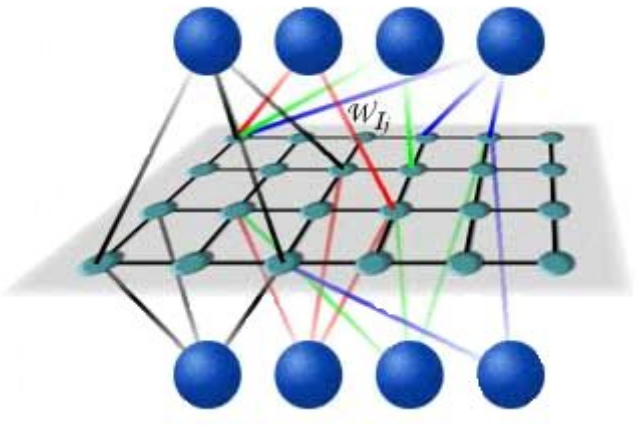

Figura 2.14 - Exemplo de arquitetura da rede de kohonen

A lei de aprendizagem da rede Kohonen é baseada na aprendizagem competitiva, onde os neurônios competem entre si, para responder a um determinado estímulo. Seguindo a seguinte lei para a atualização dos pesos.

$$w_{antigo} = w_{antigo} + \alpha(x - w_{antigo})z \tag{2.21}$$

O parâmetro α determina a velocidade de aprendizagem e é valorado entre 0 e 1, podendo variar durante uma simulação. O elemento X representa as entradas e Z correspondem às saídas.

2.6.4 Rede Neural de Base Radial

Uma rede neural com função de ativação de base radial (RBF, *radial-basis function*) é uma rede caracterizada por uma camada de entrada (que conectam a rede com seu ambiente), uma camada de saída e uma intermediária (Huamaní, 2003). Sendo que a saída passa por uma função linear, enquanto o processamento da camada intermediária dá-se através de uma função de base radial (não linear) sendo a mais usada a Gaussiana na forma:

$$z_j = \exp\left(\frac{\sum_{i=1}^{n}(x_i - w_{ji})^2}{2\sigma^2 j}\right) \quad j=1,2,...,M \quad \text{onde} \tag{2.22}$$

z_j = saídas de i-ésimo nó da camada intermediária

x = vetor de entrada

w_{ji} = os pesos da camada de entrada, que também fazem o papel de centros das funções de ativação.

M = nº de neurônios da camada intermediária

n = nº de entradas da rede

σj = são os parâmetros de dispersão para o i-ésimo nó

j = número de nós da camada intermediária.

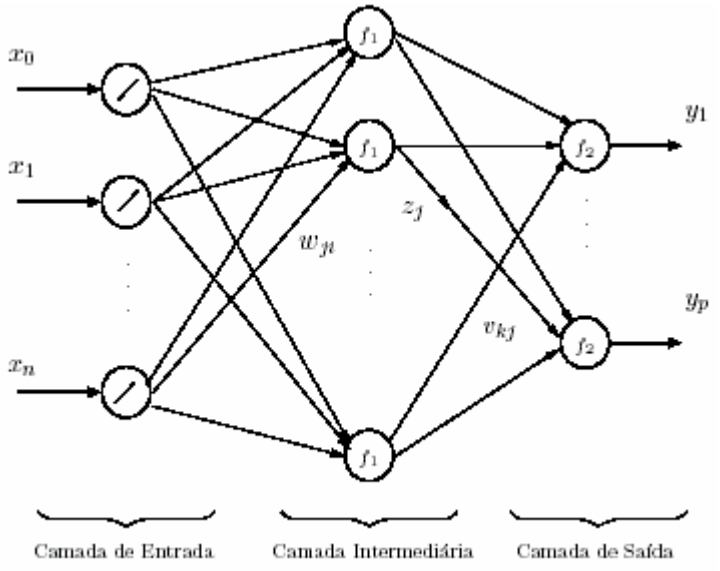

Figura 2.15 – Ilustração de uma rede neural de base radial

A saída desta rede é definida pela seguinte equação

$$y_k = \sum_{j=1}^{M} z_j * \vartheta_{kj} \tag{2.23}$$

Sendo ϑ_{kj} os pesos que conectam a camada intermediária com a camada de saída.

A denominação da função surge devido ao fato de as curvas gaussianas serem radialmente simétricas, ou seja, cada nó produz uma saída idêntica para entradas localizadas a

uma mesma distância radial fixa no centro C (Campos & Saito, 2004). Devido ao fato de utilizar curvas para promover o ajuste dos pesos, aprender significa encontrar uma região que forneça o melhor ajuste para os dados de treinamento (Haykin, 2001). Esta superfície é usada então para interpolar os dados de teste. As primeiras aplicações das funções de base radial foram na solução de problema de interpolação multivariada real.

Para o treinamento de redes neurais de base radial é necessário determinar o centro, a forma e extensão das funções, geralmente baseado em treinamento não-supervisionado, ou computação evolutiva. No aprendizado dos pesos da camada de saída são empregados algoritmos como pseudo-inversão e *OLS (Orthogonal Least Squares)*.

2.6.5 Rede Perceptrons de Multicamadas

A perceptron multicamada é uma rede neural estática, representa um comportamento direto não realimentado com um determinado número de camadas intermediárias. O vetor x = $[x_1,x_2,x_3,...,x_{n0}]$ contém n_0 entradas e o vetor y = $[y_1,y_2,y_3,...,y_{ns}]$ contém ns saídas da rede. As camadas intermediárias são representadas por ni. Cada camada, nó ou unidade é conectada com a seguinte por um peso denotado por *w*. Uma conexão da unidade j para a unidade i serve para propagar a ativação x_j desde j até i representado por w_{ji}. A saída é baseada em uma soma interna $v_j = \sum w_{ji} * x_j$ esta soma é lançada em uma função de ativação y(i) = g(*v*(i)). Baseado na topologia da rede é incluído ou não o fator constante "bias" a x(i) com seu respectivo peso, podendo aparecer também na camada intermediária. A figura a seguir ilustra uma MLP.

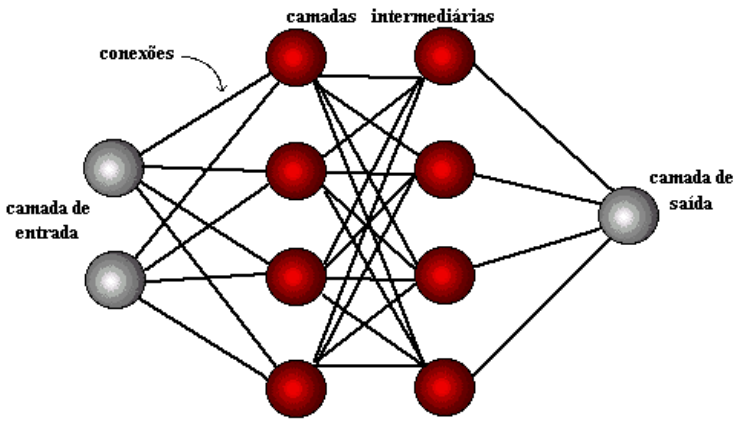

Figura 2.16 - Rede perceptron multicamada

Treinamento

O algoritmo de aprendizagem por retropropagação do erro é um dos métodos mais utilizados atualmente, pois permite o emprego de arquiteturas sofisticadas com capacidade de resolução de problemas amplos empregados em sistemas reais. Para o seu treinamento utiliza-se um conjunto aproximado de 80 à 90% do total de dados reservando o restante para testes (Campos & Saito, 2004). O conjunto de treinamento permite observar a aprendizagem da rede, e o conjunto de testes permite observar a capacidade de generalização da mesma.

Neste algoritmo cada camada tem uma função específica: a camada de entrada recebe os valores e repassa-os (linear ou não-linearmente) para a primeira camada oculta a qual os computa juntamente com os respectivos pesos através de uma função. As camadas ocultas detectam características internas e permitem que a rede crie a sua própria representação. A camada de saída recebe os estímulos da camada intermediária e constrói o padrão que será a resposta.

A retropropagação é um método que utiliza à derivada parcial do critério de erro em relação aos pesos para o ajuste dos mesmos, iterativamente na direção oposta à indicada pelo gradiente, visando atingir um erro mínimo. Durante o processamento do algoritmo back-propagation (retropagagação) a rede recebe um padrão de dados de entrada os quais são propagados através da rede, camada por camada, obtendo uma resposta na camada de saída a qual é comparada com a saída desejada. Se esta não estiver correta calcula-se o erro que é retropropagado até a camada de entrada. Neste processo, os pesos das conexões internas são alterados, para que a saída da rede aproxime-se da resposta real (Bharath & Drosen, 1994).

O algoritmo de retropropagação exige uma função de ativação continua e diferenciável (em qualquer ponto), sendo em sua grande maioria empregada a função sigmoidal, em função da derivação da regra de atualização dos pesos (Rich & Knight, 1993).

$$y_j = \frac{1}{1 + e^{(-v_j)}} \tag{2.24}$$

onde

v_j é a soma ponderada de todas as entradas (sinápticas incluindo o bias) do neurônio j multiplicadas pelos seus respectivos pesos

y_j é a saída do neurônio

Etapa de retropogação

O sinal de erro é calculado na saída do neurônio j, na iteração *n* (pertencente à última camada) após o último exemplo de treinamento apresentado.

$$e_j(n) = d_j(n) - y_j(n) \tag{2.25}$$

Para o conhecimento da energia total do erro é necessário multiplicar o quadrado dos erros por 0.5 de todos os neurônios da camada de saída. Representado pela equação (2.26), onde c engloba todos os neurônios da camada de saída.

$$E(n) = \frac{1}{2}\sum_{j \in c} e_j^{\,2}(n) \tag{2.26}$$

A energia média do erro quadrático de um conjunto de N exemplos, é obtida pela equação (2.27).

$$E_{med} = \frac{1}{N}\sum_{n=1}^{N} E(n) \tag{2.27}$$

A equação (2.27) traduz a "qualidade" da aprendizagem. O objetivo da aprendizagem é a minimização desta equação. O passo seguinte do algoritmo é a multiplicação dos pesos pelas suas respectivas entradas.

$$v_j(n) = \sum_{i=0}^{m} w_{ji}(n) * x_i \tag{2.28}$$

onde

m é o número total de entradas excluindo a contagem do bias, aplicadas ao neurônio j.

x_0 entrada fixa *+1* que multiplicado pelo peso w_{0j} representa o bias aplicado ao neurônio j.

y_j representa a saída, após a aplicação da função de ativação $\varphi_j(.)$, apresentado na equação (2.29).

$$y_j(n) = \varphi_j(v_j(n)) \tag{2.29}$$

Os procedimentos supracitados são necessários para realizar o passo para frente (propagação). Para a correção dos pesos (através da retropropagação do erro) faz-se necessária a derivação da soma instantânea dos erros quadráticos em relação aos pesos. Este gradiente é escrito conforme a regra da cadeia do cálculo como segue:

$$\frac{\partial E(n)}{\partial w_{ji}(n)} = \frac{\partial E(n)}{\partial e_j} \frac{\partial e_j}{\partial y_j(n)} \frac{\partial y_j(n)}{\partial v_j(n)} \frac{\partial v_j(n)}{\partial w_{ji}(n)} \tag{2.30}$$

A derivada parcial que indica a direção de busca para os pesos w_{ji} é definida como segue

$$\frac{\partial E(n)}{\partial e_j} = e_j(n) \tag{2.31}$$

Derivando o erro em relação a saída obtém-se:

$$\frac{\partial e_j(n)}{\partial y_j(n)} = -1 \tag{2.32}$$

O próximo passo é derivar a saída em relação ao somatório de cada neurônio j (das entradas multiplicadas pelos seus respectivos pesos). O que significa derivar a função de ativação em relação ao somatório de cada neurônio j, demonstrado a seguir.

$$\frac{\partial y_j(n)}{\partial v_j(n)} = \varphi'_j(v_j(n)) \tag{2.33}$$

Para completar a seqüência das derivações é necessário derivar o somatório de cada neurônio j em relação aos pesos, conforme a equação (2.34).

$$\frac{\partial v_j(n)}{\partial w_{ji}(n)} = y_i(n) \tag{2.34}$$

Tem-se então a regra delta definida como:

$$\Delta w_{ji}(n) = -\eta \frac{\partial E(n)}{\partial W_{ji}(n)} \tag{2.35}$$

O sinal negativo aponta uma direção para a mudança dos pesos (descida do gradiente). O parâmetro η representa a taxa de aprendizagem.

Após a definição da regra delta pode-se apresentar o gradiente local que indica para as modificações necessárias nos pesos sinápticos conforme a equação (2.36).

$$\delta_j(n) = e_j(n)\varphi'_j(v_j(n)) \tag{2.36}$$

Podendo reescrever a regra delta da seguinte forma:

$$\Delta w_{ji}(n) = \eta\delta_j(n)y_i(n)$$

$$\Delta w_{ji}(n) = \eta e_j(n)\varphi'_j(v_j(n)y_i(n) \tag{2.37}$$

Sendo que $y_i(n)$ é o sinal de entrada do neurônio j.

O passo seguinte do algoritmo requer conhecimento da posição do neurônio j. Quando j é um neurônio de saída, é um caso simples de tratar. Cada neurônio de saída da rede é suprido com uma resposta desejada particular, fazendo com que o cálculo do sinal de erro associado seja direto. Quando j é um neurônio oculto, o caso já não é tão simples. Embora a camada oculta seja de certa forma inacessível ela tem participação nos erros cometidos pela camada de saída assim ela também tem responsabilidades e precisa se adequar para que ocorra a redução do erro (Haykin 2001). Portanto duas situações são citadas.

- <u>O neurônio j é um neurônio de saída.</u>

Devido à facilidade de associação do erro é necessária apenas a aplicação da regra delta deduzida na equação (2.37).

- <u>O neurônio j é um neurônio oculto.</u>

O sinal de erro do neurônio oculto deve ser determinado recursivamente, em termos dos sinais de erro de todos os neurônios aos quais o neurônio oculto esta diretamente conectado (Haikin 2001). Para os procedimentos seguintes adota-se o índice j como sinalizador da camada oculta e k para a saída. Sendo agora o neurônio j pertencente à camada intermediária necessita-se redefinir o gradiente como segue:

$$\delta_j(n) = -\frac{\partial E(n)}{\partial y_j(n)}\frac{\partial y_j(n)}{\partial v_j(n)} \tag{2.38}$$

$$\delta_j(n) = -\frac{\partial E(n)}{\partial y_j(n)}\varphi'(v_j(n)) \tag{2.39}$$

A derivada da soma instantânea do erro em relação a uma saída interna é descrita pela equação (2.41).

$$\frac{\partial E(n)}{\partial y_j(n)} = \sum_k e_k \frac{\partial e_k}{\partial y_j(n)} \tag{2.40}$$

$$\frac{\partial e_k}{\partial y_j(n)} = \frac{\partial e_k(n)}{\partial v_k(n)} \frac{\partial v_k(n)}{\partial y_j(n)} \tag{2.41}$$

Por y_j indicar a saída de uma camada oculta tem-se a seguinte relação do erro

$$e_k(n) = d_x(n) - y_k(n) \tag{2.42}$$

$$= d_k(n) - \varphi_k(v_k(n)) \tag{2.43}$$

Tem-se então:

$$\frac{\partial e_k(n)}{\partial v_k(n)} = -\varphi_k'(v_k(n)) \tag{2.44}$$

Usando a equação ($v_k(n) = \sum_{j=0}^{m} w_{kj}(n) * y_j$) e, derivando em relação à y_j tem-se

$$\frac{\partial v_k(n)}{\partial y_j(n)} = w_{kj}(n) \tag{2.45}$$

Com as derivadas calculadas acima é possível reescrever a equação (2.36) da seguinte maneira

$$\frac{\partial E(n)}{\partial y_j(n)} = \sum_k e_k \varphi_k'(v_k(n)) w_{kj}(n) \tag{2.46}$$

$$= -\sum_k \delta_k(n)_k w_{kj}(n) \tag{2.47}$$

A correção dos pesos sinápticos obedece a uma regra geral como determinada abaixo

$$\Delta w_{ji}(n) = \eta \delta_j(n) y_i(n) \tag{2.48}$$

Da regra supracitada é fundamental observar a que camada pertence o gradiente. E $y_i(n)$ indica o sinal de entrada do neurônio j.

Assim quando o neurônio j pertence à camada oculta e seguindo a regra geral chega-se a seguinte correção de pesos

$$\Delta w_{ji}(n) = \eta \varphi'_j(v_j(n)) \sum_k e_k \varphi'_k(v_k(n)) w_{kj}(n) y_i(n) \tag{2.49}$$

A taxa de aprendizagem η define a "velocidade" da aprendizagem. Quando pequena ela gera pequenas variações na mudança dos pesos de uma iteração para a outra, originando uma trajetória suave no espaço dos pesos. Porém, se o objetivo for acelerar a taxa de aprendizagem corre-se o risco de uma variação muito grande dos pesos gerando uma trajetória oscilatória.

Para evitar as oscilações e não cair no problema dos mínimos locais (conforme figura 2.17), o ideal é acrescentar a regra de correção de pesos um fator denominado de momentum (α). Este fator procura forçar as mudanças dos pesos para que eles mantenham sempre a mesma direção (permitindo que pule os mínimos locais). Formalizando a equação (2.50).

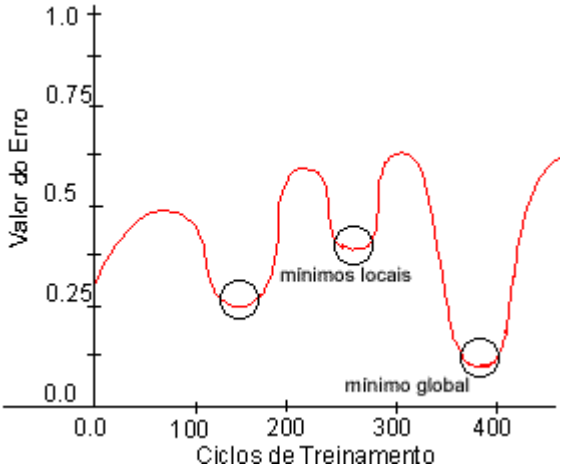

Figura 2.17 – Gráfico representando os mínimos locais e mínimo global

$$\Delta w_{ji}(n) = \alpha \Delta w_{ji}(n-1) \eta \varphi'_j(v_j(n)) \sum_k e_k \varphi'_k(v_k(n)) w_{kj}(n) y_i(n) \tag{2.50}$$

O aprendizado ocorre após muitas apresentações de um determinado conjunto de treinamento. Um conjunto de treinamento é denominado de época. Para atingir o objetivo de minimizar o erro médio quadrático são realizadas iterações de épocas.

Para o modo de aprendizado seqüencial é apresentado um conjunto de exemplos e em seguida é feita a atualização dos pesos até serem apresentados todos os exemplos de uma dada época. Porém a aprendizagem por lote adota a apresentação de todos os exemplos de uma determinada época, e somente então é feito o ajuste dos pesos sinápticos.

Para perceber se a rede atingiu os objetivos (se convergiu) é necessário estabelecer um critério de parada para a atualização dos pesos, sendo este critério baseado em considerações particulares observando as propriedades de mínimos locais ou globais.

2.7 Aplicações de Redes Neurais.

O amplo emprego de redes neurais se deve em grande parte a sua capacidade de generalização (capacidade de produzir saídas adequadas para entradas que não estavam no conjunto de exemplos apresentados), possibilitando a resolução de problemas altamente complexos. Contudo, possui também uma capacidade adaptativa elevada. Uma rede neural treinada para um determinado conjunto de exemplos pode ser facilmente retreinada para pequenas modificações. Quando uma rede neural opera em um ambiente onde as mudanças ocorrem no decorrer do tempo ela pode ser projetada para modificar seus pesos em tempo real (Haikin 2001).

As redes neurais artificiais podem ser treinadas para encontrar soluções, interpretar e classificar dados, reconhecer padrões, aproximar funções, prever eventos futuros, como também o processamento de sinais e aplicações de controle. As aplicações encontram-se em todas as áreas do conhecimento destacando-se: Processamento de imagens, setor militar, robótica, biologia e medicina, telecomunicações etc...

3 ESTIMAÇÃO DE PARÂMETROS UTILIZANDO REDES NEURAIS ARTIFICIAIS

3.1 Introdução

Para facilitar a estimação de parâmetros em sistemas não lineares é conveniente que o sistema seja caracterizado por equações a diferenças não-lineares de dimensão finita. (Lima, 2000). Dependendo das informações a priori disponíveis diferentes modelos não-lineares podem ser escolhidos se convenientemente parametrizados. As redes neurais artificiais são empregadas com sucesso nas representações de tais modelos. A opção por redes neurais para identificação é baseada em algumas características da planta como:

- Pouco conhecimento da planta;
- A apresentação de não-linearidades significativas da planta;
- Somente dados de entrada e saída estão disponíveis.

O emprego de redes neurais exige a disponibilidade de dados amostrados de toda a região de operação do sistema, no espaço de estados.

Nesta modalidade de identificação de sistemas a rede neural é o modelo, que trabalha com pares de vetores $u(t)$ e $y(t)$ que são os dados de entrada e de saída respectivamente. O sinal de erro é calculado e os pesos da rede são ajustados de tal forma que a saída da rede $\hat{y}(t)$ aproxime-se da saída real $y(t)$ para todo $u(t)$. O algoritmo de treinamento para o ajuste dos pesos da rede corresponde ao estágio de estimação de parâmetros. E a seleção da arquitetura da rede corresponde ao estágio de seleção do modelo

Neste capítulo é feita uma revisão bibliográfica das técnicas de identificação de sistemas utilizando redes neurais artificiais empregadas neste trabalho, bem como a descrição da metodologia de definição da taxa de aprendizagem adaptativa da rede neural para uma identificação mais precisa.

3.2 Estimação clássica X Estimação utilizando redes neurais.

As técnicas de identificação de sistemas utilizando redes neurais vêm conquistando espaço por serem menos rigorosas em relação às técnicas de identificação clássica. Na identificação clássica o usuário deve especificar a natureza do relacionamento entre as entradas e saídas, sendo necessários estudos aprofundados do sistema demandando tempo e dinheiro. Na identificação utilizando redes neurais o usuário deve somente especificar a topologia da rede, que é suficiente para descrever o mapeamento entrada-saída.

Comparada com a identificação clássica de sistemas dinâmicos, redes neurais apresentam a primeira vantagem que é a não necessidade da especificação da estrutura do modelo, a qual pode ser muito difícil. Isso fica claro quando a estrutura do sistema real não pertence ao conjunto de modelos que podem ser gerados pelo método escolhido para identificação (Lima, 2000).

Os modelos tradicionais ARX (modelo auto-regressivo com entradas externas) e ARMAX (modelo auto-regressivo com média móvel e entradas externas) exigem experimentos controlados sobre o intervalo de operação do processo. Portanto quando são empregados estes modelos lineares para identificar sistemas não-lineares o conjunto de dados deve ser separado em subconjuntos sob um processo aproximadamente linear, sendo que estes requisitos não são exigidos quando se fala em redes neurais. Porém, existe uma desvantagem a ser mencionada. Redes neurais necessitam da especificação dos dados de entrada sobre toda região de interesse: não conhecendo a estrutura do sistema real, os resultados dos experimentos, não há a possibilidade de extrapolar os dados para fora da região onde o experimento foi conduzido. (Lima, 2000).

Portanto, identificação de sistemas utilizando redes neurais necessita de pouca interferência do usuário, mas por outro lado requer um número consideravelmente maior de pares de entrada e saída em relação às técnicas de identificação de sistemas clássica.

3.3 Estimação de parâmetros utilizando redes neurais para o modelo Entrada - Saída

A abordagem de entrada-saída para redes neurais foi introduzida e ilustrada por NARENDRA & PARTASARATHY (1990), NERRAND *et al*. (1994). Desde então,

procedimentos de projeto para estimação de modelos usando dados de entrada-saída e procedimentos de síntese para seu controle têm sido realizados com sucesso. Para sistemas lineares controláveis e observáveis de ordem *n*, segue que o estado inicial do sistema pode ser computado a partir de *n* valores de entrada e saída e, portanto, qualquer sistema com uma entrada e uma saída (*SISO – single input single output*) pode ser descrito pela seguinte equação:

$$y(k+1) = \sum_{i=0}^{n-1} \alpha_i y(k-i) + \sum_{j=0}^{n-1} \beta_j u(k-j) \tag{3.1}$$

onde α_i *e* $\beta_j (i, j = 0,1,... n-1)$ são parâmetros constantes. Sistemas lineares *MIMO*, onde $y(k)$ e $u(k)$ são de dimensão *m* e *p*, respectivamente, podem ser descritos por:

$$y(k+1) = \sum_{i=0}^{n-1} A_i y(k-i) + \sum_{j=0}^{n-1} B_j u(k-j) \tag{3.2}$$

onde A e B são matrizes (*m* x *m*) e (*m* x *p*), respectivamente. As equações (3.1) e (3.2) representam o conhecido modelo *ARMA* para sistemas lineares *SISO* e *MIMO* invariantes no tempo, respectivamente. Portanto, para sistemas lineares invariantes no tempo dados pela representação por espaço de estados 3.4 existe uma representação entrada-saída dada pela equação (3.2).

A relação entre a descrição de entrada-saída e a representação de estados é, entretanto, consideravelmente mais complexa para sistemas não-lineares. Equações de entrada-saída são de importância fundamental na teoria de sistemas dinâmicos, porque elas descrevem o comportamento do sistema sob o ponto de vista de um observador externo. Por outro lado, muitos dos desenvolvimentos recentes da teoria de sistemas dinâmicos usam representação por espaço de estados, já que elas permitem a aplicação de técnicas de equações diferenciais e teoria de otimização. Assim, uma questão básica é decidir quando um dado operador de entrada-saída admite representação na forma de espaço de estados.

Uma questão que surge na relação entre equação de estados e equações de entrada-saída pode ser formulada como segue: dado um sistema *S* representado pela equação (3.3), pode-se produzir uma equação de entrada-saída para o sistema? Como mencionado anteriormente, em sistemas lineares a existência de equações de entrada-saída equivalentes está relacionada à habilidade de determinar os estados do sistema através de um número finito de medidas de entrada-saída.

$$x(k+1) = f[x(k), u(k)] \qquad (3.3)$$
$$y(k) = h[x(k)]$$

onde $u(k), x(k), y(k)$ representam a entrada, o estado e a saída para o instante k de tempo. Se o sistema descrito pela equação (3.3) é suposto linear e invariante no tempo, as equações que governam seu comportamento podem ser expressas por:

$$x(k+1) = Ax(k) + Bu(k) \qquad (3.4)$$
$$Y(k) = Cx(k)$$

Para sistemas não lineares, NARENDRA (1992) propõe procedimentos similares aos discutidos para sistemas lineares. Considerando o intervalo [*k, k*+*m*-1], a saída do sistema dinâmico dado pela equação (3.3), pode ser expressa como:

$$y(k) = h[x(k)]$$
$$y(k+1) = h[f[x(k), u(k)]] \qquad (3.5)$$
$$y(k+m-1) = h[f[f[...f[f[x(k), u(k)], u(k+1)]..., u(k+m-3)], u(k+m-2)]]$$

Se a equação (3.5) puder ser solucionada para *x*(*k*) em termos de valores de saída e entrada, segue-se que uma equação recursiva de entrada-saída também pode ser derivada para o sistema não-linear. Ou seja, se o sistema for observável para um tempo finito, então existe uma equação recursiva de entrada-saída. Suposições rigorosas sobre as funções *f* e *h* têm sido feitas no caso não-linear para garantir a existência de uma solução. Além disso, para existência de tais soluções, o estado inicial do sistema deve estar restrito a uma dada região do espaço de estados. Para sistemas não-lineares, uma noção de observabilidade hierárquica é possível. Estas incluem observabilidade, de simples experimentos, observabilidade genérica de simples experimentos, observabilidade forte e observabilidade genérica global. Observabilidade forte supõe a existência de um inteiro *K* tal que qualquer seqüência de entrada-saída de comprimento maior ou igual a *K* irá determinar unicamente o estado inicial da equação (3.5), enquanto observabilidade genérica assegura o mesmo para quase todas as seqüências de comprimento suficiente, uma vez que se trabalha com plantas nas quais as funções *f* e *h* são supostas desconhecidas.

3.4 Estimação de parâmetros utilizando redes neurais para o modelo NARX

Para a predição de séries temporais não-lineares pode ser empregada à equação NAR, utilizando um número finito de parâmetros, e, não sendo especificada nenhuma entrada externa, sua representação é dada pela equação (3.6) a seguir.

$$y(k+1) = f[y(k), y(k-1),..., y(k-n-1)] \tag{3.6}$$

Onde *f* é uma função não-linear podendo ser aproximada por uma rede neural.

Quando o número de atrasos a ser usado é muito pequeno é recomendado o emprego de uma rede neural RBF. Porém quando existe a necessidade de um número grande de atrasos, uma rede neural multicamada deve ser utilizada.

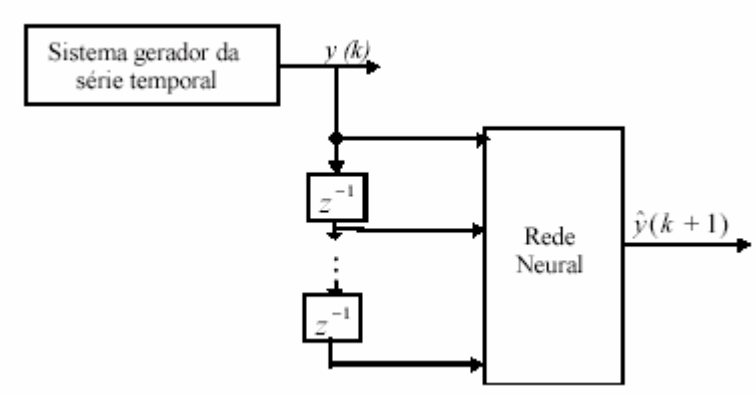

Figura 3.1 Modelo neural NAR para predição de séries temporais

Na identificação de sistemas dinâmicos, a este modelo de séries temporais pode ser acrescido um ruído e uma entrada externa *u* em instantes de tempo anteriores, podendo ser descrito pelo modelo NARX conforme equação (3.7).

$$y(k+1) = f[y(k), y(k-n+1), u(k),...,u(k-m+1)] + e(k+1) \tag{3.7}$$

onde *e*(k) é um ruído branco. A representação teórica deste sistema pode ser descrita pela equação (3.8). E representado pela figura (3.2)

$$\hat{y}(k+1) = f[y(k), y(k-n+1), u(k),...,u(k-m+1)] \tag{3.8}$$

Neste caso o erro de predição $e(k+1) = \hat{y}(k+1) - y(k+1)$ resulta no ruído $w(k+1)$, desta forma pode-se representar o sistema através de uma rede neural pela equação (3.9).

$$\hat{y}(k+1) = \varphi[y(k), y(k-n+1), u(k), ..., u(k-m+1), \theta] \tag{3.9}$$

sendo φ uma função não-linear identificada por uma rede neural multicamadas com vetor de parâmetros θ.

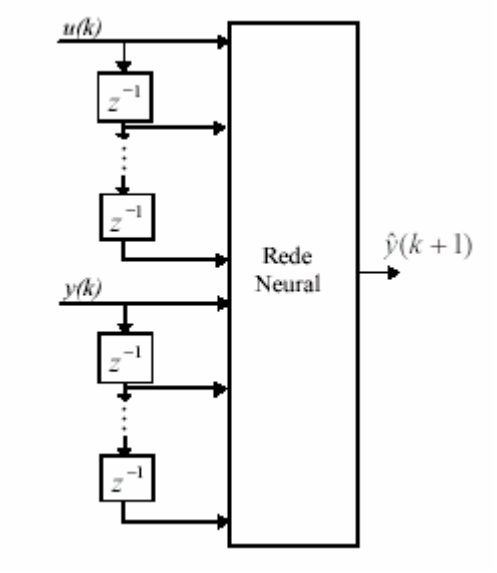

Figura 3.2 Modelo neural NARX para identificação de sistemas dinâmicos

3.5 Estimação de parâmetros utilizando redes neurais para o modelo NARMAX

Quando é considerado um modelo auto-regressivo com média móvel ele é representado pelo modelo ARMA. No momento em que são acrescidos valores passados da entrada externa tem-se o modelo ARMAX. O modelo NARMAX é uma extensão do modelo ARMAX para o caso não linear (Leontaritis & Bilings, 1985). E pode ser representado pela equação (3.10).

$$y(k+1) = h[y(k), ..., y(k-n+1), u(k), ...u(k-m+1), w(k), ..., w(k-p+1)] \tag{3.10}$$

onde os valores de $w(k), ..., w(k-p+1)$ representam os ruídos do sistema e para facilitar os cálculos eles podem ser descritos pela equação do erro (3.11).

$$e(k+1) = \hat{y}(k+1) - y(k+1) \tag{3.11}$$

Desta forma pode-se reescrever o modelo através da equação (3.12).

$$y(k+1) = h[y(k),..., y(k-n+1), u(k),...u(k-m+1), e(k),..., e(k-p+1)] \tag{3.12}$$

Para representar este modelo implementado por uma rede neural pode-se utilizar a equação (3.13).

$$\hat{y}(k+1) = \varphi[y(k),..., y(k-n+1), u(k),...u(k-m+1), e(k),..., e(k-p+1); \theta] \tag{3.13}$$

onde φ é uma função não-linear implementada pela rede. Se o modelo suposto é correto e se φ se aproxima de *h* com uma precisão arbitrária então a equação que representa o sistema é ótima e o erro é mínimo. (Lima, 2000).

Segundo (Sjoberg, 1995) o modelo NARMAX tem a vantagem de permitir uma maior flexibilidade no processo de modelagem do ruído. A presença de entradas vinculadas diretamente ao ruído evita a necessidade de se utilizar um grande número de elementos da entrada (*u(k), y(k)*) para incorporar a descrição do ruído, como ocorre no modelo NARX. Porém, pode-se acrescentar ainda que, devido à realimentação da informação da saída o ajuste de parâmetros é mais complexo.

3.6 Caracterização das plantas:

Para representar sistemas em aplicação e controle (Narendra & Parthasarathy 1990) definiram quatro classes de modelos. Estas classes de modelos podem ser descritas pelas seguintes equações a diferenças não-lineares:

Modelo I: $$y(k+1) = \sum_{i=0}^{n-1} \alpha_i y(k-i) + f[u(k), u(k-1),...u(k-m+1)] \tag{3.14}$$

Modelo II: $$y(k+1) = f[y(k), y(k-1),...y(k-n+1)] + \sum_{i=0}^{n-1} \beta_i u(k-i) \tag{3.15}$$

Modelo III: $$y(k+1) = f[y(k), y(k-1),..y(k-n+1)] + g[u(k), u(k-1),..., u(k-m+1)] \tag{3.16}$$

Modelo IV: $y(k+1) = f[y(k), y(k-1), ...y(k-n+1); u(k), u(k-1), ..., u(k-m+1)]$ (3.17)

Onde *u(k), y(k)* representa o par entrada e saída da planta SISO no instante de tempo *k.* As letras *m* e *n* representam o número de atrasos em *u* e *y* respectivamente. As funções f e g são funções diferenciáveis em relação aos seus argumentos. A seguir cada modelo será abordado individualmente.

Modelo I : A saída da planta não-linear desconhecida é suposta depender linearmente de seus valores passados e não linearmente dos valores passados da entrada (figura 3.3).

Modelo II : Neste modelo a saída *y(k)* depende linearmente dos valores atuais e passados da entrada *u(k),* mas não linearmente em relação aos valores passados da própria saída y(k) (figura 3.4).

Modelo III : Neste caso a planta depende não linearmente dos valores passados da entrada e da saída. As funções não-lineares aplicadas aos dados de saída e de entrada são diferentes e perfazem o modelo com a sua soma (figura 3.5).

Modelo IV : Este modelo apresenta uma generalização dos modelos anteriormente vistos. A saída depende de uma função não-linear dos valores passados de entrada e de saída. (figura 3.6).

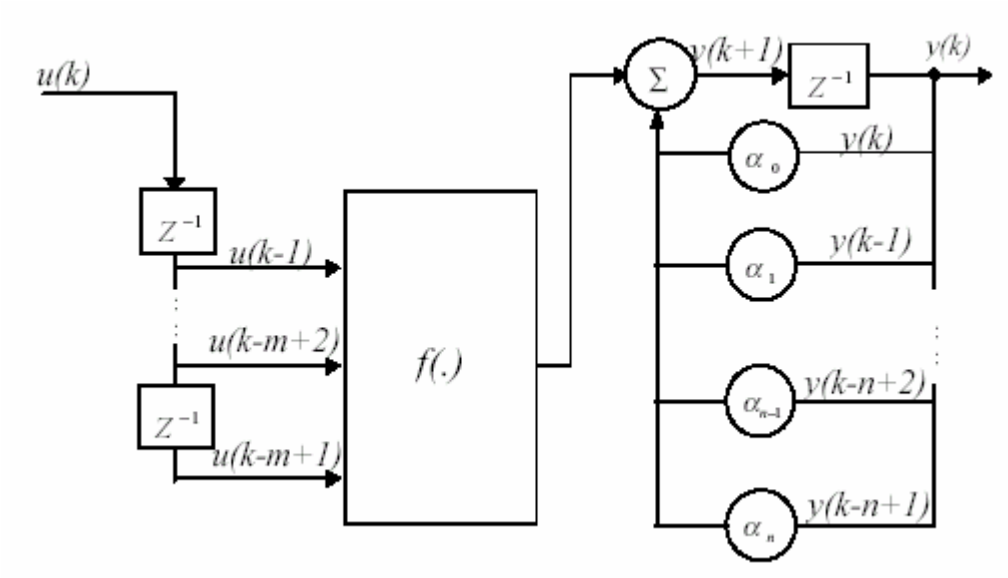

Figura 3.3 – Estrutura do Modelo I

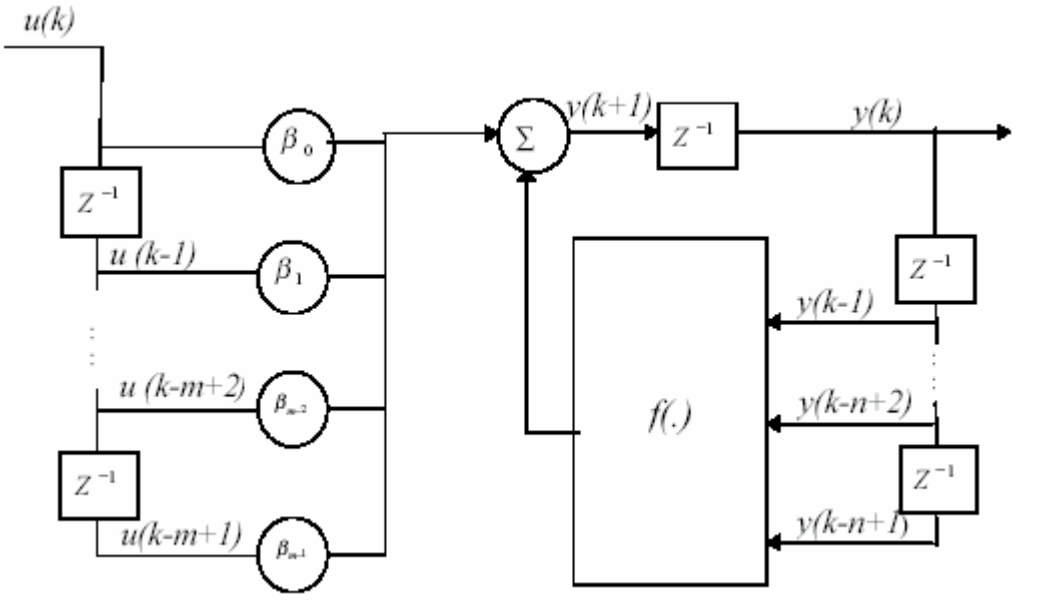

Figura 3.4 – Estrutura do Modelo II

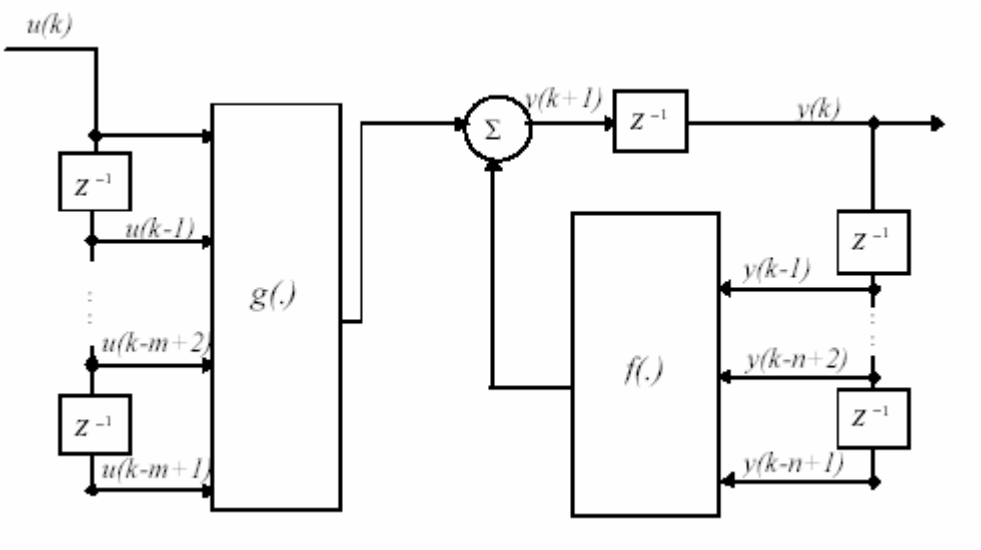

Figura 3.5 – Estrutura do Modelo III

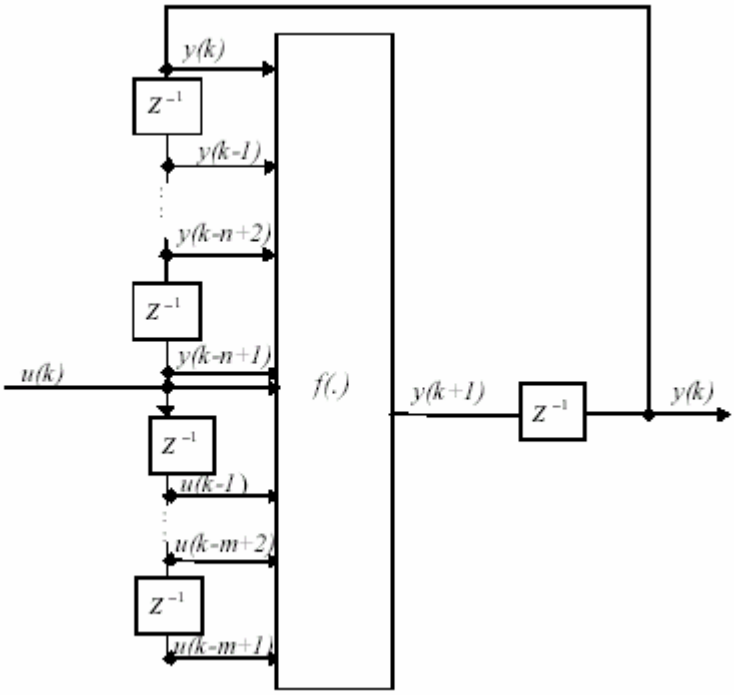

Figura 3.6 Estrutura do Modelo IV

3.7 Processos de identificação utilizando redes neurais

Para uma identificação concisa a rede neural deve conter um número suficiente de camadas e de neurônios por camada de forma a representar a dinâmica existente entre os dados de entrada e de saída. Portanto as funções não-lineares nas equações a diferenças podem

ser substituídas por redes neurais com parâmetros (pesos) a determinar. Para identificar uma planta, um modelo de identificação é escolhido baseado na informação prévia sobre a classe a qual ele pertence.

Dentro da estrutura do modelo identificado existem duas classes a serem citadas; Modelo Paralelo e Modelo Série Paralelo, como descrito a seguir.

Modelo Paralelo: Considerando N_1 e N_2 pesos de uma rede neural, aplicados a estrutura do modelo III, tem-se a substituição de N_1 e N_2 por g e f. Se $y(k+1)$ e $\hat{y}(k+1)$ correspondem a saída da planta e a saída do modelo no instante $(k+1)$, então o modelo é descrito pela equação (3.18) a seguir:

$$\hat{y}(k+1) = N_2[\hat{y}(k),\hat{y}(k-1),...,\hat{y}(k-n+1)] + N_1[u(k),u(k-1),...,u(k-m+1)] \quad (3.18)$$

Neste caso os parâmetros N_1 e N_2 devem ser ajustados usando a retropropagação dinâmica.

Modelo Série-Paralelo: A diferença entre o modelo anterior (paralelo) e série-paralelo é a substituição dos valores de saída empregados no modelo de $\hat{y}(k+1)$ por $y(k+1)$ gerando a seguinte saída do modelo:

$$\hat{y}(k+1) = N_2[y(k),y(k-1),...,y(k-n+1)] + N_1[u(k),u(k-1),...,u(k-m+1)] \quad (3.19)$$

Observa-se que a malha de realimentação não contém um elemento não linear então a atualização dos pesos pode ser feita através da retropropagação estática.

Após feita a seleção da estrutura faz-se necessária a determinação dos números de entradas, números de saídas e atraso de tempo associado com o processo (determinar a ordem do modelo). Existem dois métodos utilizados para determinar a ordem do modelo (Bomberger e Seborg, 1998):

Método dos números de Lipschitz. Foi desenvolvido para modelos determinísticos, porém pode ser também aplicado a sistemas com baixo nível de ruídos, baseado na propriedade de continuidade das funções que representam os modelos entrada-saída, sendo que não depende do uso de qualquer método de aproximação ou estrutura de modelo particular.

Método da falsa vizinhança mais próxima (false nearest neigbohbors). Método desenvolvido para a determinação da dimensão mínima associada ao número de observações deslocadas no tempo necessárias para modelar o comportamento dinâmico do sistema, para sistemas caóticos.

Para se determinar a ordem do modelo da melhor maneira é necessário ter um número suficiente de Insight físico acerca do sistema a ser modelado.

3.8 Determinação da taxa de aprendizagem adaptativa através de Algoritmos Genéticos

O algoritmo de treinamento utiliza uma taxa de aprendizagem η que define a "velocidade" da aprendizagem. Quando pequena, ela gera pequenas variações na mudança dos pesos de uma iteração para a outra, aumentando o tempo e o custo computacional. Porém, se o objetivo for acelerar a taxa de aprendizagem corre-se o risco de uma variação muito grande dos pesos gerando uma trajetória oscilatória levando a divergência de valores.

Sendo, portanto, de grande interesse a definição de uma taxa de aprendizagem baseada em princípios científicos e que produza bons resultados. Propôs-se então o emprego de algoritmos genéticos para a sua determinação a cada época de treinamento. Os passos padrões de um algoritmo genético são apresentados resumidamente a seguir.

3.8.1 Algoritmos Genéticos

Algoritmos Genéticos (AG) são algoritmos de procura baseados nas mecânicas de seleção e genética natural (GOLDBERG, 1989), onde os indivíduos mais fortes e aptos têm probabilidade maior de sobreviver e evoluir. Sua lógica de funcionamento está baseada nas leis da evolução natural propostas por Charles Darwin, e utiliza uma nomenclatura derivada da genética, na qual os seres vivos têm no núcleo de suas células moléculas de DNA, onde estão registradas suas características em *genes*, organizados em cadeias, chamadas cromossomos.

Em vários problemas científicos há a necessidade de se encontrar uma solução ótima ou um conjunto de soluções que melhor representam esse problema, normalmente escrito em forma de função. Todos os valores possíveis e suas combinações são chamados de *espaço de busca* e representa o domínio do problema. Essa busca pela melhor solução é conhecida como

problema de otimização onde, normalmente, visa-se maximizar uma função em busca do maior valor possível. Entretanto, podem ocorrer funções não deriváveis, descontínuas ou *multimodais* que impedem o uso de métodos de otimização tradicional, baseados em cálculo numérico. Nos problemas dessa natureza é que se percebe a aplicação dos AG, sendo um método probabilístico, mas tendo regras que permitem o algoritmo evoluir para regiões de prováveis soluções ótimas e possui certa facilidade de fugir dos máximos (ou mínimos) locais.

Os AG são algoritmos computacionais que a partir de uma amostra da população procura evoluir conforme o desempenho de seus indivíduos, privilegiando os mais aptos. A estrutura básica de um AG simples é mostrada na figura 3.7.

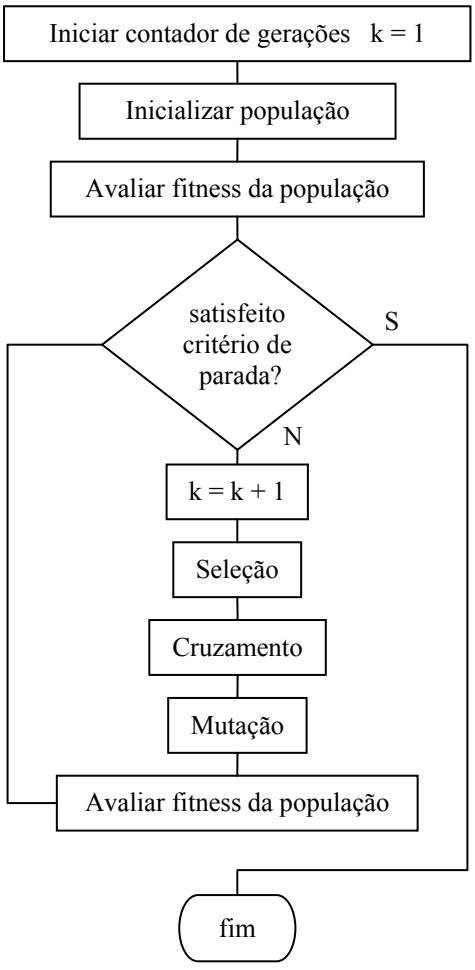

Figura 3.7 – Fluxograma de um Algoritmo Genético simples

Nesse algoritmo, a família inicial de cromossomos é gerada aleatoriamente e em seguida submetida à função de avaliação para determinar a aptidão ou *fitness* de cada cromossomo. Depois é verificado se foi atingido o critério de parada, que normalmente está relacionado a precisão da resposta e ao número máximo de gerações. Caso não seja satisfeito esse critério, entra-se no laço onde ocorrem o incremento das gerações e a aplicação dos operadores genéticos de reprodução, cruzamento e mutação para, em seguida, os cromossomos serem novamente submetidos à função de avaliação.

3.8.2 Representação dos Cromossomos

O ponto de partida nos AG é a representação de cada possível solução *x* no espaço de busca (universo de soluções) como uma seqüência de símbolos *s* pertencentes a um alfabeto finito $U = \{ s_1 , s_2 , ... , s_m \}$. A representação mais adequada vai depender de cada problema. A mais simples é a binária, onde os elementos de *U* são compostos por 0 e 1. A cada seqüência de *s* corresponde a um cromossomo, e cada elemento de *s* equivale a um *gene*. Sendo que cada *gene* pode assumir qualquer valor de *U*, então cada elemento de *U* que forma o *gene* é chamado de alelo, conforme figura 3.8. A posição de um *gene* dentro do cromossomo é chamada de *lócus*. Normalmente, associa-se cada indivíduo a um único cromossomo, o que permite usar os termos *indivíduo* e *cromossomo* indistintamente. Na figura 3.9 estão representados os mesmos cromossomos, porém com representação real ou ponto flutuante.

cromossomos	{	1	0	1	1		1	1	0	0		0	1	1	0
		0	1	0	1		1	0	0	0		1	0	1	1
							⏟						↑		
							gene						alelo		

Figura 3.8 – Representação binária de uma família de cromossomos

cromossomos	{	11		12		6
		5		8		11
				⏟		
				gene		

Figura 3.9 – Representação real de uma família de cromossomos

3.8.3 Inicialização da População

Normalmente a população inicial é gerada aleatoriamente, procurando cobrir todo o espaço de busca do problema. Para a representação binária é usada uma técnica de gerar aleatoriamente a metade da população, enquanto que a outra metade surge da inversão dos bits da primeira metade. Outra possibilidade consiste em, conhecendo-se uma ou mais características da população, gerar os indivíduos de acordo com alguma regra inicial.

3.8.4 Função de Avaliação

A função de avaliação, função objetivo ou função custo é quem determina o grau de aptidão (fitness) de um cromossomo (possível solução). Nos AG todos os cromossomos, em cada geração, são submetidos à função objetivo para determinar o valor de *fitness* de cada indivíduo que definirá a probabilidade de seleção e reprodução para a geração seguinte. Ela define uma medida de quão adaptado está o indivíduo ao ambiente, ou seja, quanto maior o valor da função de *fitness*, maiores são as chances de reproduzir e passar seu material genético às gerações futuras. A escolha da função de avaliação adequada é um dos fatores determinante para o sucesso na convergência do AG.

3.8.5 Operadores Genéticos

São responsáveis pelas mudanças que ocorrem na população, ao passar de uma geração para outra, mudando algumas de suas características. Os operadores genéticos são necessários para diversificar a população de cromossomos e manter as características de adaptação adquirida pelas gerações anteriores. A evolução dos AG é baseada na aplicação dos operadores genéticos sobre a população, com o objetivo de selecionar os mais aptos, cruzar os cromossomos entre si e provocar mutação nos descendentes. Os operadores genéticos são: seleção, reprodução, cruzamento ou recombinação e mutação.

3.8.5.1 Seleção

Nesta etapa os indivíduos mais aptos são selecionados para reprodução, gerando cópias (filhos) que serão submetidos aos operadores de cruzamento e mutação, formando uma nova população para a geração seguinte. O processo de seleção se inicia com o cálculo do grau de aptidão de cada cromossomo quando submetidos, um a um, à função de avaliação (função

custo ou *fitness*). Essa aptidão F_i está associada à medida do desempenho de cada indivíduo ao ser avaliado pela função objetivo.

Os cromossomos que possuírem maior valor de *fitness* terão maior probabilidade de serem selecionados para a geração seguinte, podendo, o mesmo cromossomo, ser selecionado mais que uma vez. Dependendo do método de seleção escolhido poderá haver maior ou menor pressão seletiva, provocando um impacto direto na diversidade da população. Se a pressão seletiva for muito alta, somente os melhores indivíduos serão selecionados. Conseqüentemente, a diversidade da população diminui e a possibilidade da convergência prematura do AG para um mínimo local aumenta bastante. Por outro lado, com pressão seletiva baixa torna a procura ineficaz, gerando diversidade da população e tornando a busca aleatória (MICHALEWICZ, 1999).

Há vários métodos de seleção. Pode-se citar o da roleta, método baseado no *Ranking*, seleção por torneio e modelos elitistas, além de outras variações desses métodos (GOLDBERG, 1989; MICHALEWICZ, 1999). É natural pensar que em qualquer método de seleção escolhido, há uma probabilidade maior de sobrevivência dos indivíduos mais adaptados.

3.8.5.2 Cruzamento ou crossover

É um dos operadores genéticos responsável pela troca de informações entre os cromossomos ocorrendo através de um processo sexuado, ou seja, envolve mais de um indivíduo. Esses indivíduos, chamados pais, recombinam suas características permitindo que as próximas gerações (filhos) herdem suas informações genéticas.

Normalmente, atribui-se uma probabilidade de cruzamento p_c que define os indivíduos da população a serem cruzados. O valor dessa probabilidade de cruzamento depende do problema em estudo, mas é comum se trabalhar com valores elevados. Em (GOLDBERG, 1989) é mostrado um exemplo com $p_c = 0{,}6$, ou seja, há uma probabilidade de 60% da população sofrer cruzamento. No entanto, é possível trabalhar com probabilidades variando numa larga escala de valores.

O processo de cruzamento é feito em duas etapas. Na primeira, escolhem-se, dentro da população, quais indivíduos devem passar pelo processo de cruzamento. Essa seleção é feita gerando-se um número aleatório $r(i)$ uniformemente distribuído entre [0 1] para cada

cromossomo *i*, e em seguida verificar se $r(i) \leq p_c$, então o cromossomo *i* deve cruzar. A segunda etapa trata do cruzamento propriamente dito, onde os cromossomos selecionados, na etapa anterior, são enfileirados e tomados dois a dois para passar pelo processo de cruzamento. Há várias formas de se fazer o cruzamento genético entre dois cromossomos as mais empregadas são: Cruzamento simples, Cruzamento Uniforme, Cruzamento Aritmético, Cruzamento aritmético com os extremos, Cruzamento Heurístico.

3.8.5.3 Mutação

O operador de mutação pode ser aplicado, indiferentemente, antes ou depois do operador de cruzamento, e opera sobre um único cromossomo de cada vez. A mutação consiste em modificar aleatoriamente um ou mais gene do cromossomo pai, e tem como objetivo restaurar a diversidade genética eventualmente perdida durante o processo evolutivo (GOLDBERG, 1989). É através da probabilidade de mutação p_m que se define a freqüência da ocorrência de mutação em cada indivíduo. Normalmente, utiliza-se probabilidade baixa de mutação para evitar a possível destruição de indivíduos mais adaptados e tornar o AG sem direção, isto é, tornar a busca da solução ótima completamente aleatória.

3.8.6 Critério de Parada

Os Algoritmos Genéticos são métodos iterativos de busca estocástica e, como tal, necessitam de um critério de parada para informar ao programa o momento de parar. Esse critério pode ser o número máximo de iterações ou gerações da população, ou ainda, verificar a evolução do melhor indivíduo após determinado número de gerações. Além desses, são possíveis outras variações e combinações entre os vários critérios de parada.

4 RESULTADOS

4.1 Introdução

Neste capítulo é apresentada a metodologia usada para a identificação de sistemas, bem como a estrutura, parâmetros e funcionamento das Redes Neurais Artificiais aplicadas à estimação de parâmetros. Os resultados são apresentados em gráficos e tabelas para validar as simulações. Para o desenvolvimento da rede neural foi utilizado o software MATLAB (marca registrada de MathWorks, Inc). A escolha desse software, dentre outros aplicativos, foi determinada pela facilidade no manuseio de vetores e matrizes e uma boa qualidade da saída gráfica.

4.2 Rede Neural Escolhida

A rede neural utilizada neste trabalho é uma rede Perceptron Multicamadas com uma camada de entrada, uma camada de saída e uma camada oculta, ilustrada na figura 4.1.

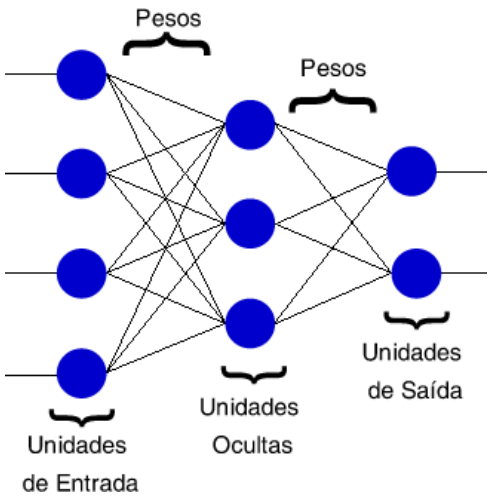

Figura 4.1 – Configuração da rede neural utilizada

A opção por uma camada oculta é baseada no trabalho de (Cybenko 1989) o qual demonstra que é possível obter bons resultados em identificação de sistemas não lineares com apenas uma camada oculta.

O modelo escolhido para identificação é caracterizado pelo modelo IV apresentado por (Narendra & Parthasarathy 1990). Representado pela equação 4.1 cuja saída *y(k)* depende de uma função não-linear dos valores passados de entrada e de saída (figura 4.2)

$$\textit{Modelo IV}: \quad y(k+1) = f[y(k), y(k-1), \ldots y(k-n+1); u(k), u(k-1), \ldots, u(k-m+1)] \qquad (4.1)$$

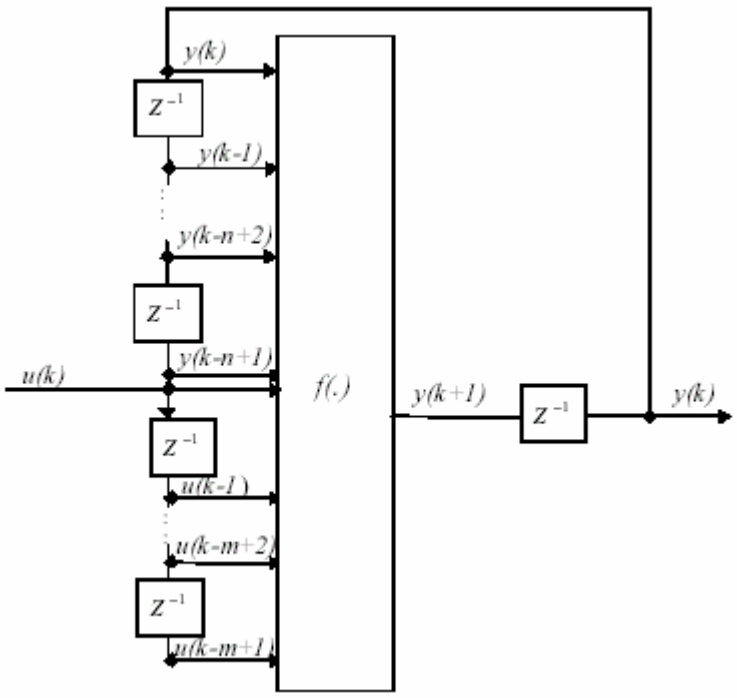

Figura 4.2 - Modelo IV de identificação de sistemas

Para cada sistema estudado foram utilizadas 3 arquiteturas de redes neurais. Para a camada de entrada foi determinado o número de neurônios conforme o sistema em questão. Para o modelo de Entrada-Saída (entrada única) há um neurônio de entrada (u(k)) mais o bias. Para o modelo NARX são 4 neurônios de entrada (u(k-1), u(k-2), y(k-1) e y(k-2)) mais o bias. Para o modelo NARMAX são 6 neurônios de entrada (u(k-1), u(k-2), y(k-1) , y(k-2), e(k-1) e e(k-2)) mais o bias. Portanto, existem nove configurações, três tipos de entrada, três tipos de camada oculta com 4, 8 e 12 neurônios e uma saída.

A camada de entrada cabe o oficio de enviar os dados de entrada para a rede, os pesos são determinados de forma aleatória a partir da informação da quantidade de neurônios da camada oculta e o sistema a ser utilizado. Para a camada oculta são feitos testes com 4, 8 e 12 neurônios processadores, que por sua vez enviam os dados para a camada de saída, a qual é constituída de um neurônio processador.

O conjunto de dados utilizados para o treinamento é de 50% do total de dados, e os outros 50% foram reservados para a validação do processo.

O algoritmo utilizado na aprendizagem da rede neural (adaptação de pesos sinápticos) é o back-propagation apresentado no capítulo 2.

A função de ativação utilizada é a tangente hiperbólica a qual é uma função não-linear, possuí a propriedade adicional de se poder definir suas derivadas em função delas mesmas, significando que se pode reutilizar o valor já calculado para a ativação ao se realizar a retropropagação do erro, sem necessidade de novos cálculos complexos. É uma função contínua e tem intervalo de existência entre –1 e 1, sendo assintótica nos dois ramos. Isto é, embora ela tenda a –1 e a 1, tais valores não são alcançados nunca.

O Algoritmo Genético empregado, para a obtenção da taxa de aprendizagem, é decimal com seleção Geométrica Normalizada com parâmetro de pressão seletiva para o ranking geométrico de 3%. Mutação não Uniforme no domínio de busca, com 1% de mutação inicial e 4% de mutação final.

O espaço de busca do algoritmo genético para a definição da taxa de aprendizagem foi determinado entre 0 e 1, com um número máximo de 10 gerações.

Para a taxa de aprendizagem fixa foi determinado o parâmetro 0.1, pois através de varias simulações foi o valor que apresentou os melhores resultados levando em consideração o erro médio quadrático do treinamento da rede e o tempo de processamento.

4.3 Dados do sistema

Para a análise da rede neural foram utilizados 3 sistemas com não linearidades diferentes. O primeiro sistema analisado é linear, de segunda ordem, com a entrada saturada, os dados de entrada e saída foram obtidos utilizando a ferramenta *Simulink* do software MATLAB. O segundo sistema analisado também é linear, de terceira ordem, porém, a sua

saída que é saturada, novamente os dados foram obtidos com a ferramenta *Simulink*. O terceiro sistema é um conversor tipo buck, cujos dados foram obtidos do livro Introdução à Identificação de Sistemas Técnicas Lineares e Não Lineares Aplicadas a Sistemas Reais (Aguirre, 2004).

4.3.1 Sistema 1

O sistema utilizado neste teste é representado na figura 4.3.

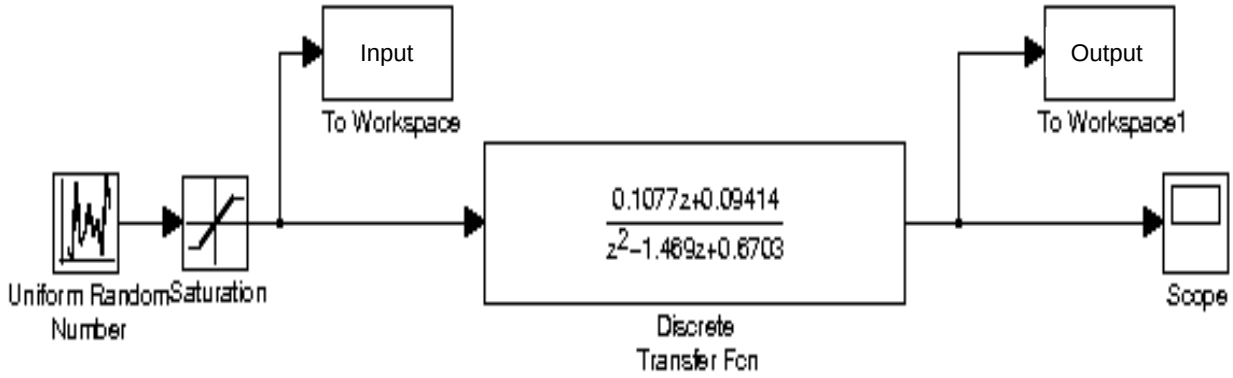

Figura 4.3 - Esquema para gerar dados com saturação na entrada.

O sinal de entrada é aleatório com amplitude que oscila entre -1 e 1, e média zero, ele passa por um saturador com limite superior de 0.75 e inferior de -0.75, gerando o sinal de entrada do sistema. Este sinal é aplicado na função de transferência de segundo grau, reapresentada na equação (4.2), discreta no tempo. A resposta do sistema é o sinal de saída.

$$H(z) = \frac{0{,}1077z + 0{,}09414}{z^2 - 1{,}469z + 0{,}6703} \tag{4.2}$$

O sinal de entrada aplicado ao sistema e o sinal de sua saída são apresentados na figura 4.4. E são utilizados para o procedimento de treinamento e validação para todos os testes do sistema 1.

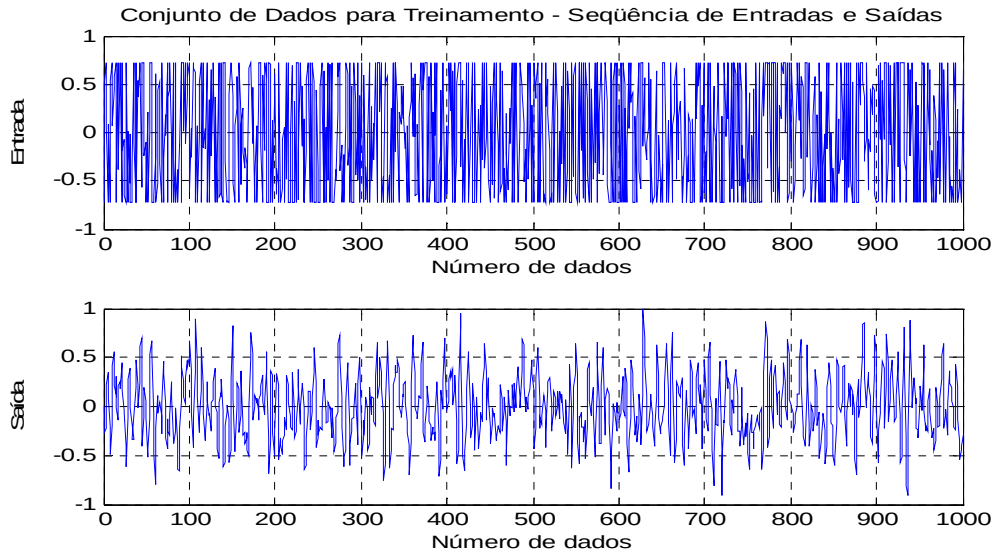

Figura 4.4 - Gráfico representativo das entradas e saídas do sistema com entrada saturada.

Apresenta-se a seguir os gráficos de saída, erro médio quadrático no treinamento e erro médio quadrático na validação do sistema 1, comparativos entre a taxa de aprendizagem fixa (RNTF) e adaptativa (RNTA), como também a evolução da taxa de aprendizagem. Iniciando-se os testes com 4 neurônios na camada oculta.

O gráfico a seguir apresenta a saída

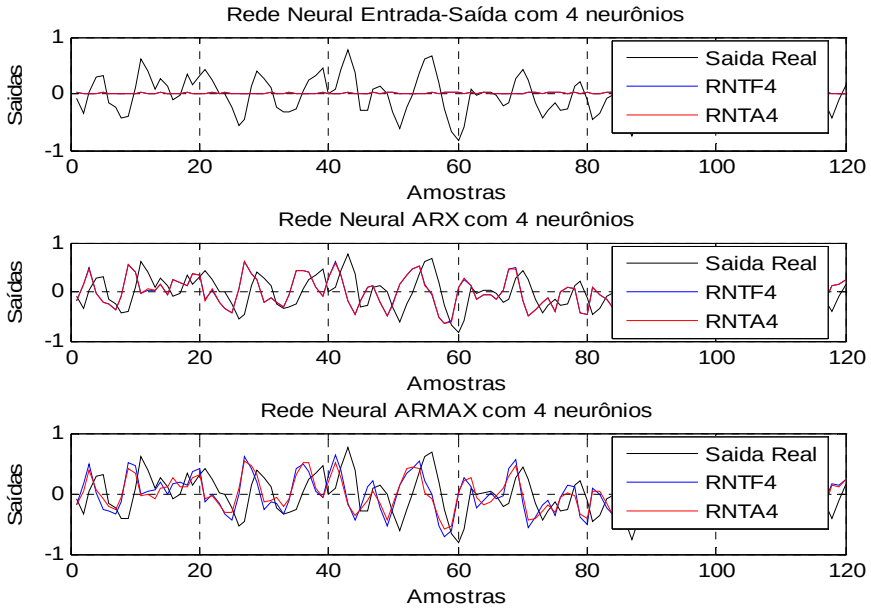

Figura 4.5 - Gráfico com a saída validada pela rede com 4 neurônios na camada oculta para o modelo Entrada-Saída, NARX, NARMAX (taxa de aprendizagem fixa e adaptativa)

O gráfico a seguir apresenta o erro no treinamento

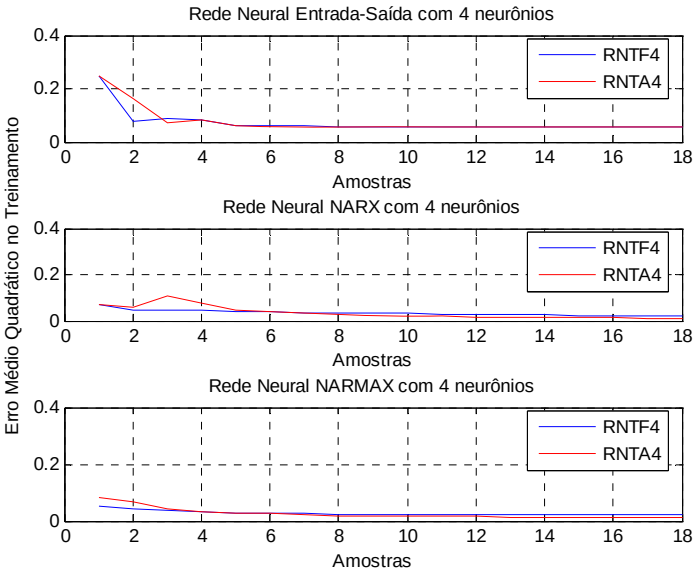

Figura 4.6 - Gráfico do erro médio quadrático no treinamento da rede com 4 neurônios na camada oculta para o modelo entrada-saída, NARX, NARMAX (taxa de aprendizagem fixa e adaptativa)

O gráfico a seguir apresenta o erro na validação

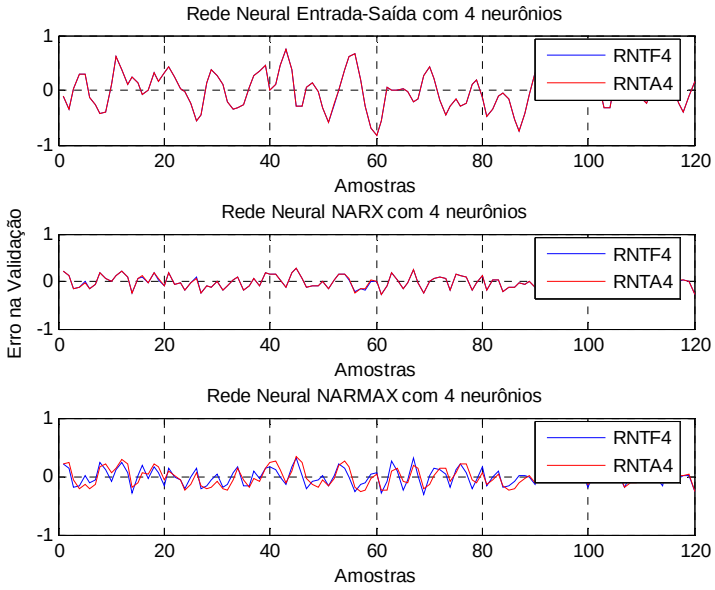

Figura 4.7 - Gráfico do erro médio quadrático da rede na validação com 4 neurônios na camada *oculta para o modelo entrada-saída,NARX, NARMAX (taxa de aprendizagem fixa e adaptativa)*

O gráfico a seguir apresenta a evolução da taxa de aprendizagem

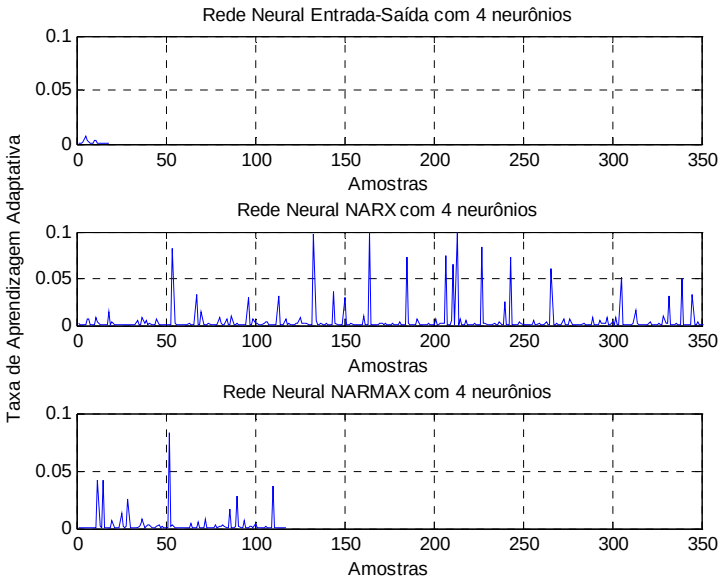

Figura 4.8 - Gráfico da evolução da taxa de aprendizagem com 4 neurônios na camada oculta para o modelo entrada-saída, NARX, NARMAX

As tabelas a seguir apresentam o número de épocas, o tempo de processamento da rede na fase de treinamento, o erro quadrático médio de treinamento e de validação para a arquitetura com 4 neurônios na camada oculta, para os modelos Entrada- Saída, NARX, NARMAX com a taxa de aprendizagem fixa e adaptativa.

Tabela 4.1 – Resumo dos Resultados obtidos pela rede para taxa de aprendizagem fixa com 4 neurônios na camada oculta.

Estrutura	Épocas	Tempo de Processamento	Erro Quadrático Médio - Treinamento	Erro Quadrático Médio – Validação
Entrada-Saída 4 neurônios camada oculta	7.9000000e+001	1.0000000e+000	5.8988043e-002	5.5275504e-002
NARX – 4 neurônios camada oculta.	1.0000000e+003	7.0000000e+000	9.3263165e-003	8.8135661e-003
NARMAX – 4 neurônios camada oculta	9.0200000e+002	7.0000000e+000	9.2710365e-003	1.0290072e-002

Tabela 4.2 – Resumo dos Resultados obtidos pela rede para taxa de aprendizagem variável com 4 neurônios na camada oculta.

Estrutura	Épocas	Tempo de Processamento	Erro Quadrático Médio - Treinamento	Erro Quadrático Médio – Validação

Entrada -Saída 4 neurônios camada oculta	1.8000000e+001	2.0000000e+000	7.4818304e-002	5.5284368e-002
NARX – 4 neurônios camada oculta.	5.0800000e+002	5.3000000e+001	9.6522405e-003	8.8203457e-003
NARMAX – 4 neurônios camada oculta	2.4800000e+002	2.7000000e+001	9.9411049e-003	1.4052090e-002

As figuras 4.5 a 4.8 mostram que os resultados obtidos com a taxa de aprendizagem fixa e adaptativa possuem comportamento semelhante. Graficamente os erros médios quadráticos da taxa adaptativa tendem a ser menores no final (figura 4.6) em relação à taxa fixa. Das tabelas 4.1 e 4.2 tem-se que a menor média se dá com o modelo NARX. Observa-se ainda, que a rede neural com taxa de aprendizagem fixa utiliza tempos menores de processamento no treinamento, em contrapartida a rede neural com taxa de aprendizagem adaptativa converge sempre com o menor número de épocas.

O sistema 1 agora será abordado para uma nova configuração na camada oculta, sendo que esta passa a ter 8 neurônios, e o objetivo é tentar obter uma melhor representação matemática do sistema, ou seja, menor erro médio quadrático no treinamento e na validação. Apresenta-se a seguir os gráficos de saída, erro médio quadrático no treinamento e erro médio quadrático na validação do sistema 1, comparativos entre a taxa de aprendizagem fixa (RNTF) e adaptativa (RNTA), como também a evolução da taxa de aprendizagem. Com 8 neurônios na camada oculta.

O gráfico a seguir apresenta a saída da rede

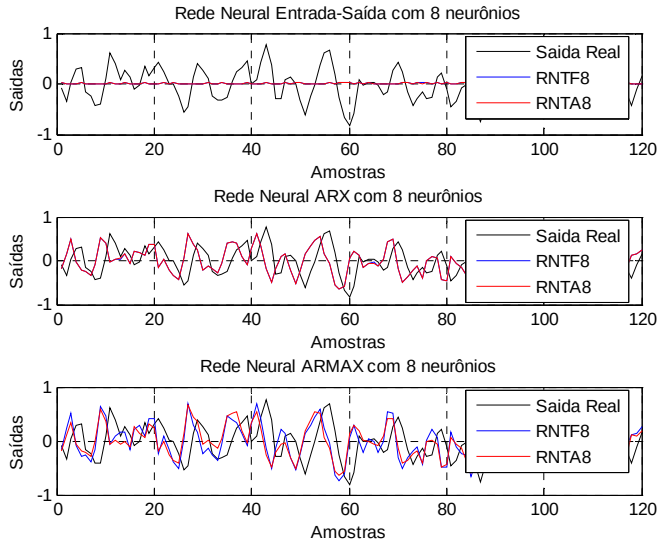

Figura 4.9- Gráfico com a saída validada pela rede com 8 neurônios na camada oculta para o modelo Entrada-Saída, NARX, NARMAX (taxa de aprendizagem fixa e adaptativa)

O gráfico a seguir apresenta o erro no treinamento

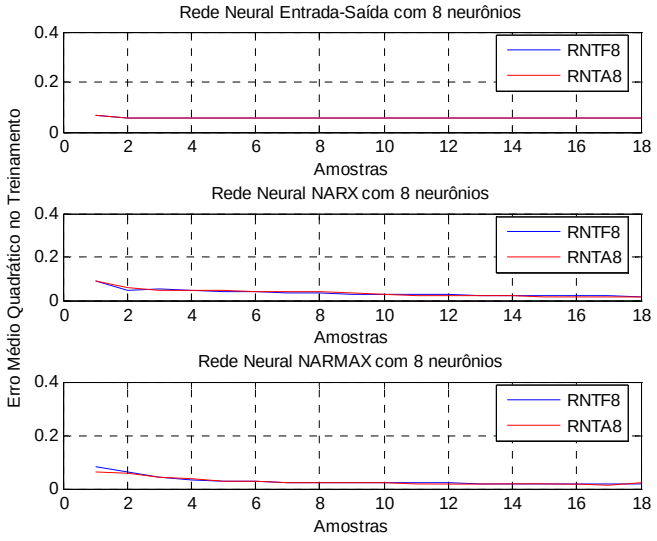

Figura 4.10 - Gráfico do erro médio quadrático no treinamento da rede com 8 neurônios na camada oculta para o modelo entrada-saída, NARX, NARMAX (taxa de aprendizagem fixa e adaptativa)

O gráfico a seguir apresenta o erro na validação

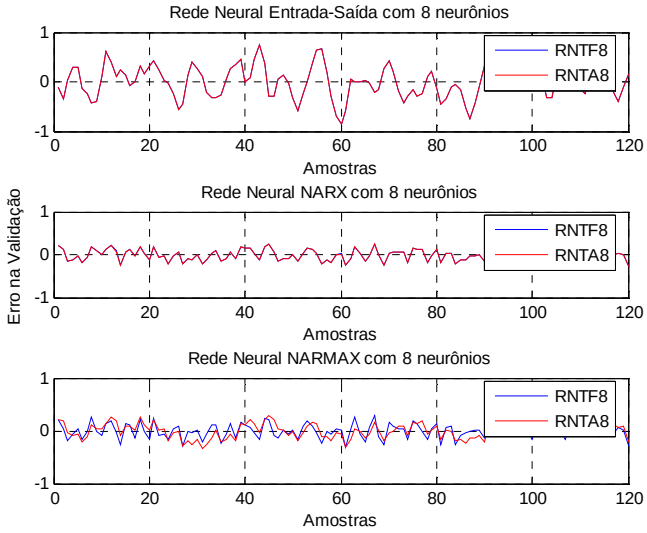

Figura 4.11 - Gráfico do erro médio quadrático na validação da rede com 8 neurônios na camada oculta para o modelo entrada-saída, NARX, NARMAX (taxa de aprendizagem fixa e adaptativa)

O gráfico a seguir apresenta a evolução da taxa de aprendizagem

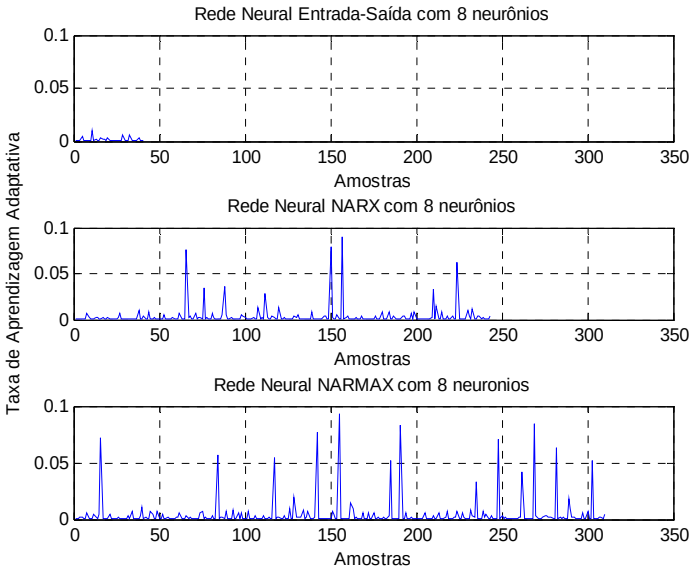

Figura 4.12 - Gráfico da evolução da taxa de aprendizagem rede com 8 neurônios na camada oculta para o modelo entrada-saída, NARX, NARMAX

As tabelas a seguir apresentam o número de épocas, o tempo de processamento da rede na fase de treinamento, o erro quadrático médio de treinamento e de validação para a

arquitetura com 8 neurônios na camada oculta, para os modelos Entrada- Saída, NARX, NARMAX com a taxa de aprendizagem fixa e adaptativa.

Tabela 4.3 – Resumo dos Resultados obtidos pela rede para taxa de aprendizagem fixa com 8 neurônios na camada oculta.

Estrutura	Épocas	Tempo de Processamento	Erro Quadrático Médio – Treinamento	Erro Quadrático Médio - Validação
Entrada-Saída - 8 neurônios camada oculta	1.9000000e+001	1.0000000e+000	5.5983617e-002	5.5270890e-002
NARX – 8 neurônios camada oculta.	7.2100000e+002	9.0000000e+000	9.5044141e-003	8.8166616e-003
NARMAX – 8 neurônios camada oculta	9.3000000e+002	1.0000000e+001	9.1083462e-003	9.7131220e-003

Tabela 4.4 – Resumo dos Resultados obtidos pela rede para taxa de aprendizagem adaptativa com 8 neurônios na camada oculta.

Estrutura	Épocas	Tempo de Processamento	Erro Quadrático Médio – Treinamento	Erro Quadrático Médio - Validação
Entrada -Saída 8 neurônios camada oculta	4.1000000e+001	8.0000000e+000	5.5548785e-002	5.5272841e-002
NARX – 8 neurônios camada oculta.	2.4300000e+002	5.4000000e+001	1.0845039e-002	8.8131533e-003
NARMAX – 8 neurônios camada oculta	3.1000000e+002	6.2000000e+001	9.9851164e-003	1.1999701e-002

As figuras 4.9 a 4.12 mostram que os resultados obtidos com a taxa de aprendizagem fixa e adaptativa possuem comportamento semelhante. Graficamente os erros médios quadráticos da taxa adaptativa e fixa tendem a ser iguais no final (figura 4.10). Das tabelas 4.3 e 4.4 tem-se que a menor média se dá com o modelo NARX. Observa-se ainda, que a rede

neural com taxa de aprendizagem fixa utiliza tempos menores de processamento no treinamento, em contrapartida a rede neural com taxa de aprendizagem adaptativa converge com o menor número de épocas, com exceção do modelo Entrada Saída.

Uma nova e última configuração é aplicada ao sistema 1. A arquitetura com 12 neurônios na camada oculta tem a função de tentar reduzir ainda mais erro médio quadrático no treinamento e na validação, observado nas configurações anteriores.

Apresenta-se a seguir os gráficos de saída, erro médio quadrático no treinamento e erro médio quadrático na validação do sistema 1, comparativos entre a taxa de aprendizagem fixa (RNTF) e adaptativa (RNTA), como também a evolução da taxa de aprendizagem. Com 12 neurônios na camada oculta.

O gráfico a seguir apresenta a saída da rede com 12 neurônios na camada oculta

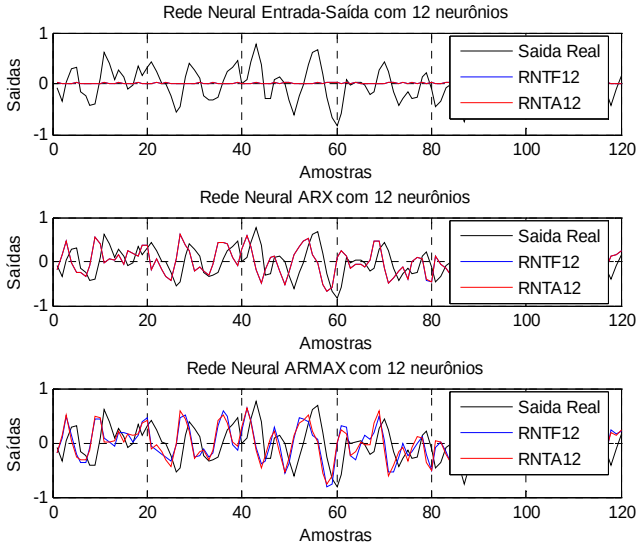

Figura 4.13 - Gráfico com a saída validada pela rede com 12 neurônios na camada oculta para o modelo entrada-saída, NARX, NARMAX (taxa de aprendizagem fixa e adaptativa)

O gráfico a seguir apresenta a o erro médio quadrático no treinamento da rede com 12 neurônios na camada oculta

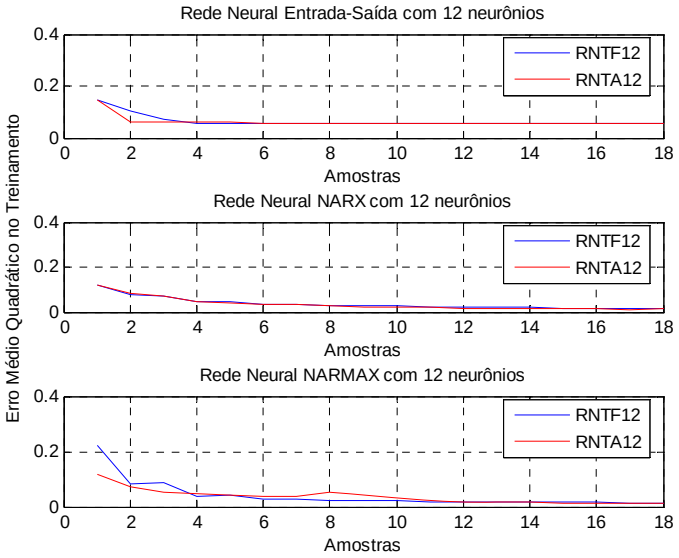

Figura 4.14 - Gráfico do erro médio quadrático no treinamento da rede com 12 neurônios na camada oculta para o modelo entrada-saída, NARX, NARMAX (taxa de aprendizagem fixa e adaptativa)

O gráfico a seguir apresenta a o erro médio quadrático na validação da rede com 12 neurônios na camada oculta

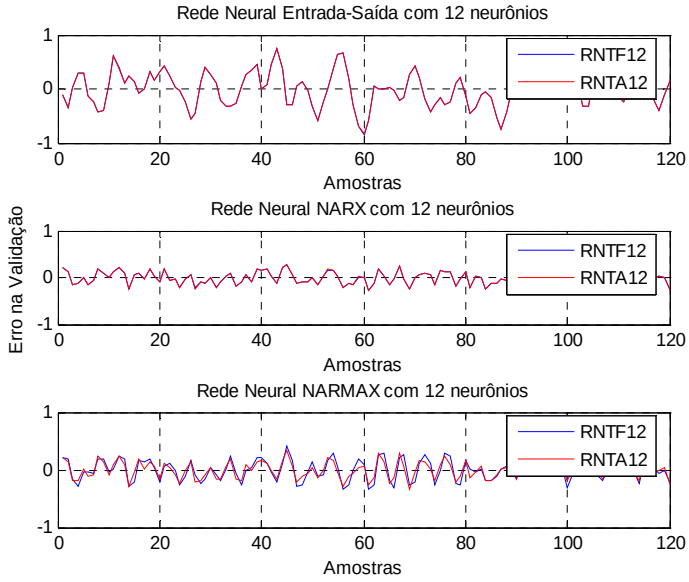

Figura 4.15 - Gráfico do erro médio quadrático na validação da rede com 12 neurônios na camada oculta para o modelo entrada-saída, NARX, NARMAX (taxa de aprendizagem fixa e adaptativa)

O gráfico a seguir apresenta a evolução da taxa de aprendizagem

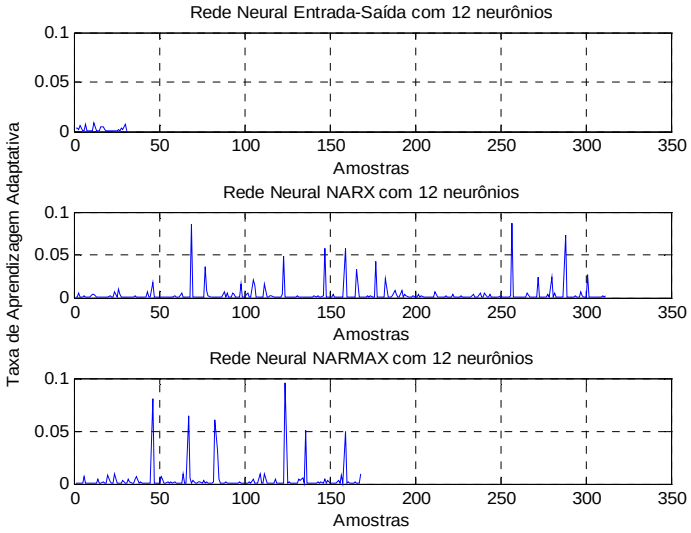

Figura 4.16 - Gráfico da evolução da taxa de aprendizagem para o modelo entrada-saída, NARX, NARMAX com 12 neurônios na camada oculta.

As tabelas a seguir apresentam o número de épocas, o tempo de processamento da rede na fase de treinamento, o erro quadrático médio de treinamento e de validação para a arquitetura com 12 neurônios na camada oculta, para os modelos Entrada- Saída, NARX, NARMAX com a taxa de aprendizagem fixa e adaptativa.

Tabela 4.5 – Resumo dos Resultados obtidos pela rede para taxa de aprendizagem fixa com12 neurônios na camada oculta.

Estrutura	Épocas	Tempo de Processamento	Erro Quadrático Médio – Treinamento	Erro Quadrático Médio - Validação
Entrada -Saída 12 neurônios camada oculta	2.1000000e+001	1.0000000e+000	6.2628763e-002	5.5272983e-002
NARX – 12 neurônios camada oculta.	1.0000000e+003	1.5000000e+001	9.2581986e-003	8.8485365e-003
NARMAX – 12 neurônios camada oculta	5.4200000e+002	9.0000000e+000	9.7692842e-003	1.5731847e-002

Tabela 4.6 – Resumo dos Resultados obtidos pela rede para taxa de aprendizagem Adaptativa com 12 neurônios na camada oculta.

Estrutura	Épocas	Tempo de Processamento	Erro Quadrático Médio – Treinamento	Erro Quadrático Médio - Validação
Entrada-Saída 12 neurônios camada oculta	3.1000000e+001	9.0000000e+000	5.8938115e-002	5.5272110e-002
NARX – 12 neurônios camada oculta.	3.1200000e+002	9.9000000e+001	1.0243682e-002	8.8415410e-003
NARMAX – 12 neurônios camada oculta	1.6800000e+002	4.7000000e+001	1.1654474e-002	1.1286076e-002

As figuras 4.13 a 4.16 mostram que os resultados obtidos com a taxa de aprendizagem fixa e adaptativa possuem comportamento semelhante. Graficamente os erros médios quadráticos da taxa adaptativa e fixa tendem a ser iguais no final (figura 4.14). Das tabelas 4.5 e 4.6 tem-se que a menor média se dá com o modelo NARX. Observa-se ainda, que o a rede neural com taxa de aprendizagem fixa utiliza tempos menores de processamento no treinamento, em contrapartida a rede neural com taxa de aprendizagem adaptativa converge com o menor número de épocas, com exceção do modelo Entrada Saída.

Dos resultados apresentados, observa-se que as diferentes configurações da rede para a representação do sistema 1, encontram soluções semelhantes. Desta forma, a configuração com 4 neurônios é a mais indicada, pois ela gera um modelo com menor complexidade e com um custo computacional menor.

Outra observação a ser feita é que os resultados encontrados pela rede neural com taxa fixa e adaptativa levam á resultados semelhantes nas três configurações, demonstrando que a metodologia para encontrar o valor da taxa de aprendizado está correta.

Observou-se ainda que em todas as configurações apresentadas o tempo de processamento é menor para a taxa de aprendizagem fixa, porém o número de iterações é consideravelmente menor para a taxa de aprendizagem adaptativa exceto para o modelo Entrada-Saída. Este aumento de tempo de processamento e diminuição do número de iterações

ocorre em função do emprego do programa que define a taxa de aprendizagem adaptativa através de algoritmos genéticos.

4.3.2 Sistema 2

O sistema utilizado neste teste é representado na figura 4.17

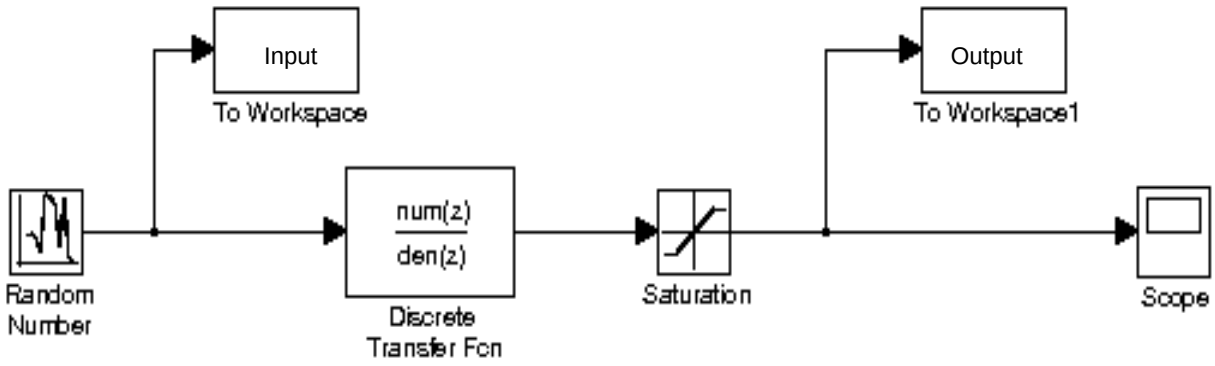

Figura 4.17 - Esquema para gerar dados com saturação na saída

O sinal de entrada é aleatório com amplitude que oscila na faixa de 1 e -1, com média zero. Este sinal é aplicado na função de transferência de terceiro grau, apresentada na equação (4.3), discreta no tempo. A resposta do sistema então passa por um saturador o qual varia de –0.12 a 0.12, este é o sinal de saída adotado.

$$H(z) = \frac{0{,}2659z^2 - 0{,}1155z + 0{,}0235}{z^3 - 0{,}6813z^2 - 0.0049z - 0{,}1477} \tag{4.3}$$

O sinal de entrada aplicado ao sistema e o sinal de sua saída são apresentados na figura 4.18. E são utilizados para o procedimento de treinamento e validação para todos os testes do sistema 2.

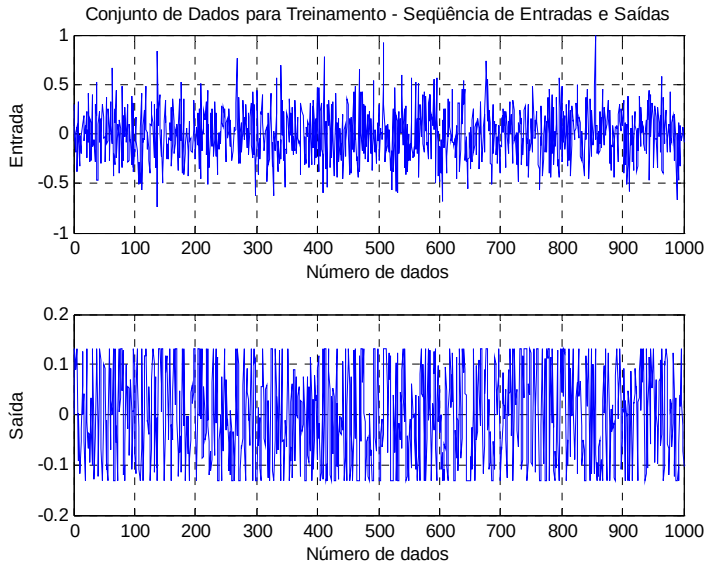

Figura 4.18 - Gráfico representativo das entradas e saídas do sistema

Apresenta-se a seguir os gráficos de saída, erro médio quadrático no treinamento e erro médio quadrático na validação do sistema 2, comparativos entre a taxa de aprendizagem fixa (RNTF) e adaptativa (RNTA), como também a evolução da taxa de aprendizagem. Com 4 neurônios na camada oculta.

O gráfico a seguir apresenta a saída da rede com 4 neurônios na camada oculta

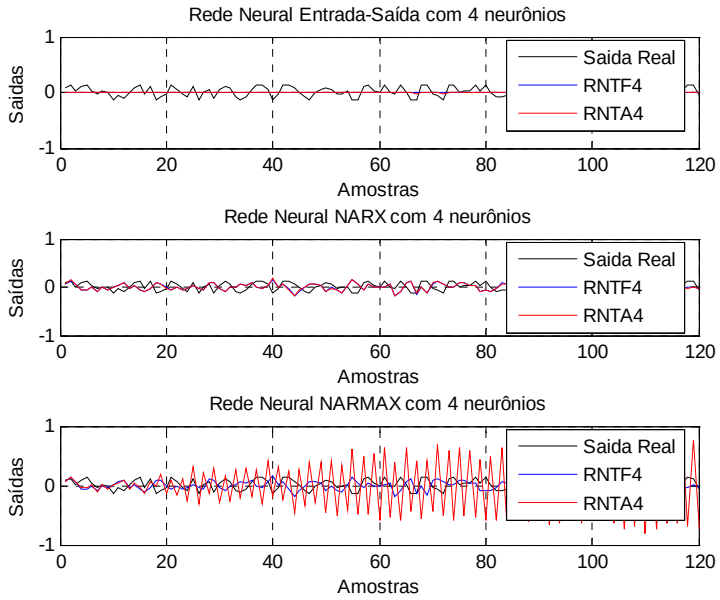

Figura 4.19 - Gráfico com a saída validada pela rede com 4 neurônios na camada oculta para o modelo entrada-saída, NARX, NARMAX (taxa de aprendizagem fixa e adaptativa)

O gráfico a seguir apresenta o erro médio quadrático no treinamento da rede com 4 neurônios na camada oculta

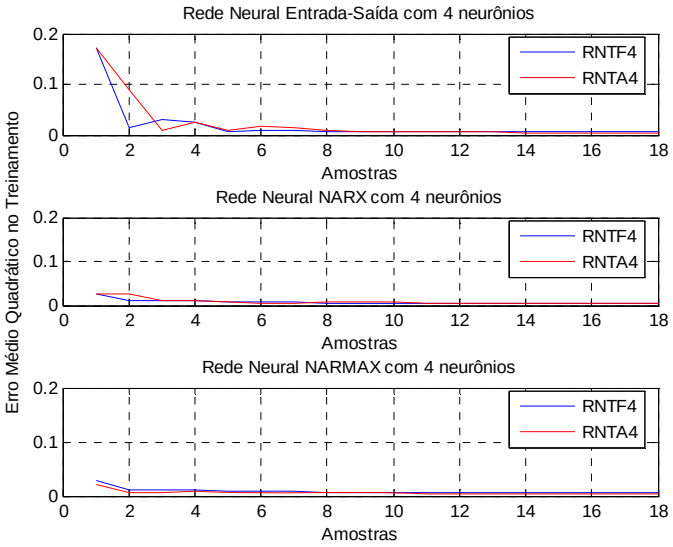

Figura 4.20 - Gráfico do erro médio quadrático da rede com 4 neurônios na camada oculta para o modelo entrada-saída, NARX, NARMAX (taxa de aprendizagem fixa e adaptativa)

O gráfico a seguir apresenta o erro médio quadrático na validação da rede com 4 neurônios na camada oculta

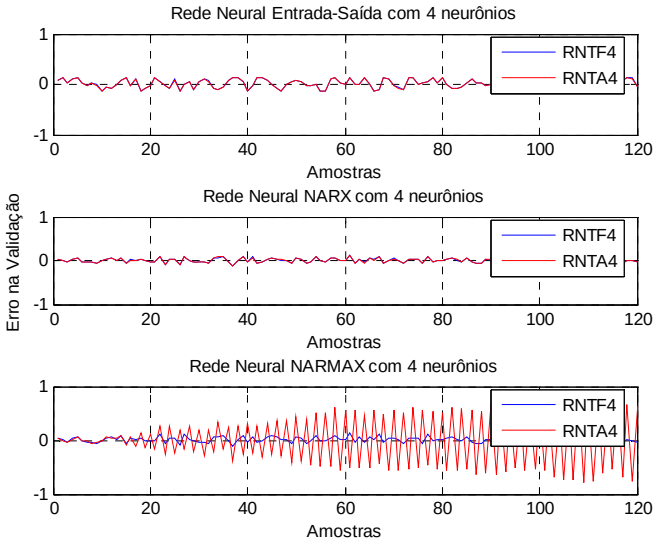

Figura 4.21 - Gráfico do erro médio quadrático na validação da rede com 4 neurônios na camada oculta para o modelo entrada-saída, NARX, NARMAX (taxa de aprendizagem fixa e adaptativa)

O gráfico a seguir apresenta a evolução da taxa de aprendizagem da rede com 4 neurônios na camada oculta

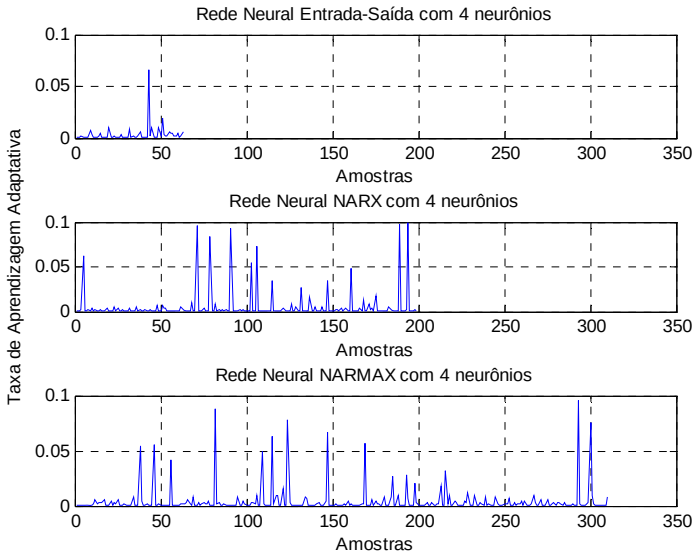

Figura 4.22 - Gráfico da evolução da taxa de aprendizagem com 4 neurônios na camada oculta para o modelo entrada-saída, NARX, NARMAX

As tabelas a seguir apresentam o número de épocas, o tempo de processamento da rede na fase de treinamento, o erro quadrático médio de treinamento e de validação para a arquitetura com 4 neurônios na camada oculta, para os modelos Entrada- Saída, NARX, NARMAX com a taxa de aprendizagem fixa e adaptativa.

Tabela 4.7 – Resumo dos Resultados obtidos pela rede para taxa de aprendizagem fixa com 4 neurônios na camada oculta.

Estrutura	Épocas	Tempo de Processamento	Erro Quadrático Médio - Treinamento	Erro Quadrático Médio - Validação
Entrada -Saída 4 neurônios camada oculta	2.7300000e+002	2.0000000e+000	5.2570494e-003	4.2591837e-003
NARX – 4 neurônios camada oculta.	9.9800000e+002	7.0000000e+000	2.1553705e-003	1.8257728e-003
NARMAX – 4 neurônios camada oculta	1.0000000e+003	8.0000000e+000	2.5757059e-003	1.8181172e-003

Tabela 4.8 – Resumo dos Resultados obtidos pela rede para taxa de aprendizagem adaptativa com 4 neurônios na camada oculta.

Estrutura	Épocas	Tempo de Processamento	Erro Quadrático Médio - Treinamento	Erro Quadrático Médio - Validação
Entrada -Saída 4 neurônios camada oculta	6.3000000e+001	7.0000000e+000	9.5099328e-003	4.2608171e-003
NARX – 4 neurônios camada oculta.	1.9900000e+002	2.4000000e+001	2.6680711e-003	1.8318552e-003
NARMAX – 4 neurônios camada oculta	2.2600000e+002	3.0000000e+001	2.5058240e-003	9.3310144e-003

As figuras 4.19 a 4.22 mostram que os resultados obtidos com a taxa de aprendizagem fixa e adaptativa possuem comportamento semelhante. Graficamente os erros médios quadráticos da taxa adaptativa tendem a ser menores no final (figura 4.20) em relação à taxa fixa. Das tabelas 4.7 e 4.8 tem-se que a menor média se dá com o modelo NARX. Observa-se ainda, que a rede neural com taxa de aprendizagem fixa utiliza tempos menores de processamento no treinamento, em contrapartida a rede neural com taxa de aprendizagem adaptativa converge sempre com o menor número de épocas.

O sistema 2 agora será abordado para uma nova configuração na camada oculta, esta passa a ter 8 neurônios, o objetivo é tentar obter uma melhor representação matemática do sistema, ou seja, menor erro médio quadrático no treinamento e na validação.
Apresenta-se a seguir os gráficos de saída, erro médio quadrático no treinamento e erro médio quadrático na validação do sistema 2, comparativos entre a taxa de aprendizagem fixa (RNTF) e adaptativa (RNTA), como também a evolução da taxa de aprendizagem. Com 8 neurônios na camada oculta.

O gráfico a seguir apresenta a saída da rede 8 neurônios na camada oculta.

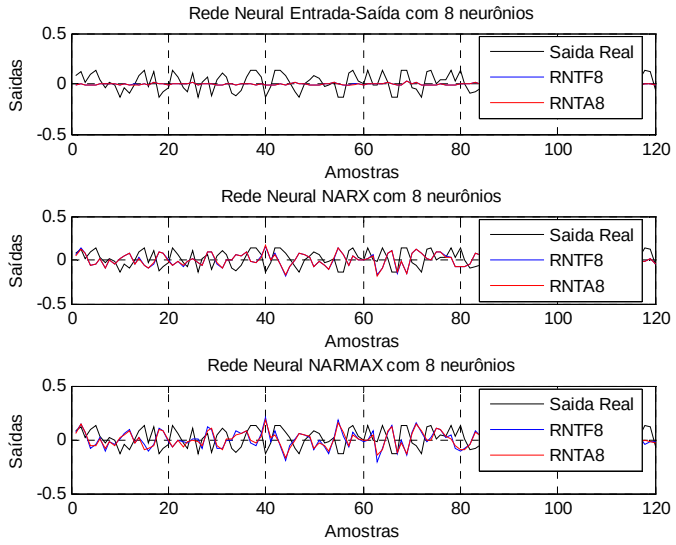

Figura 4.23 - Gráfico com a saída validada pela rede com 8 neurônios na camada oculta para o modelo entrada-saída, NARX, NARMAX (taxa de aprendizagem fixa e adaptativa).

O gráfico a seguir apresenta o erro médio quadrático no treinamento da rede com 8 neurônios na camada oculta

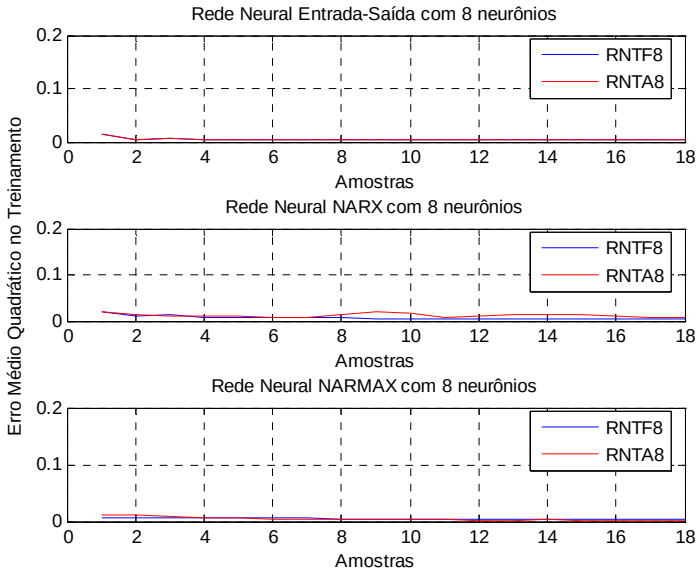

Figura 4.24 - Gráfico do erro médio quadrático no treinamento da rede com 8 neurônios na camada oculta para o modelo entrada-saída, NARX, NARMAX (taxa de aprendizagem fixa e adaptativa)

O gráfico a seguir apresenta o erro médio quadrático na validação da rede com 8 neurônios na camada oculta

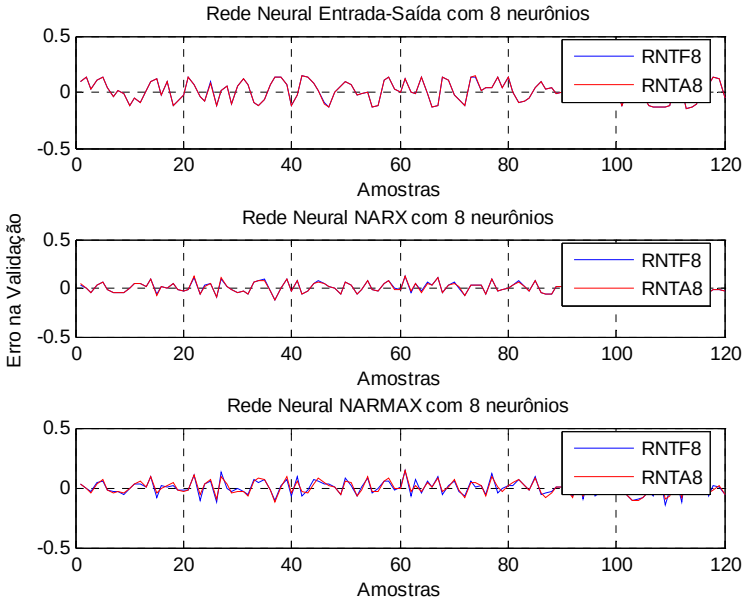

Figura 4.25 - Gráfico do erro médio quadrático na validação da rede com 8 neurônios na camada oculta para o modelo entrada-saída, NARX, NARMAX (taxa de aprendizagem fixa e adaptativa)

O gráfico a seguir apresenta a evolução da taxa de aprendizagem da rede com 8 neurônios na camada oculta

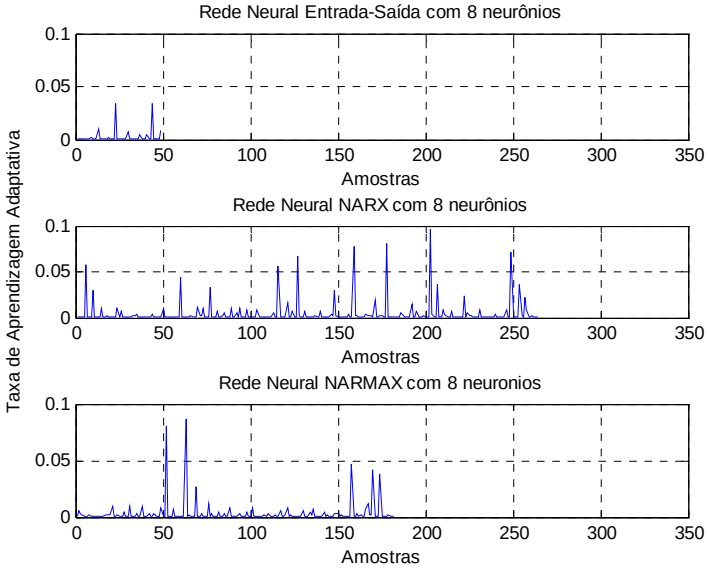

Figura 4.26 - Gráfico da evolução da taxa de aprendizagem com 8 neurônios na camada oculta para o modelo entrada-saída, NARX, NARMAX

As tabelas a seguir apresentam o número de épocas, o tempo de processamento da rede na fase de treinamento, o erro quadrático médio de treinamento e de validação para a

arquitetura com 8 neurônios na camada oculta, para os modelos Entrada- Saída, NARX, NARMAX com a taxa de aprendizagem fixa e adaptativa.

Tabela 4.9 – Resumo dos Resultados obtidos pela rede para taxa de aprendizagem fixa com 8 neurônios na camada oculta.

Estrutura	Épocas	Tempo de Processamento	Erro Quadrático Médio - Treinamento	Erro Quadrático Médio - Validação
Entrada -Saída - 8 neurônios camada oculta	4.9000000e+001	1.0000000e+000	4.5326087e-003	4.2567505e-003
NARX – 8 neurônios camada oculta.	6.5200000e+002	6.0000000e+000	2.1345943e-003	1.8163444e-003
NARMAX – 8 neurônios camada oculta	1.0000000e+003	1.0000000e+001	2.2292283e-003	2.0848608e-003

Tabela 4.10 – Resumo dos Resultados obtidos pela rede para taxa de aprendizagem adaptativa com 8 neurônios na camada oculta.

Estrutura	Épocas	Tempo de Processamento	Erro Quadrático Médio - Treinamento	Erro Quadrático Médio - Validação
Entrada -Saída 8 neurônios camada oculta	4.9000000e+001	1.1000000e+001	4.4971832e-003	4.2588573e-003
NARX – 8 neurônios camada oculta.	2.6400000e+002	5.8000000e+001	2.9048225e-003	1.8247183e-003
NARMAX – 8 neurônios camada oculta	1.2600000e+002	2.5000000e+001	2.0369677e-003	1.9220821e-003

As figuras 4.23 a 4.26 mostram que os resultados obtidos com a taxa de aprendizagem fixa e adaptativa possuem comportamento semelhante. Graficamente os erros médios quadráticos da taxa adaptativa tendem a ser menores no final (figura 4.24) em relação à taxa fixa, com exceção do modelo NARX. Das tabelas 4.9 e 4.10 tem-se que a menor média se dá

com o modelo NARX. Observa-se ainda, que o a rede neural com taxa de aprendizagem fixa utiliza tempos menores de processamento no treinamento, em contrapartida a rede neural com taxa de aprendizagem adaptativa converge sempre com o menor número de épocas.

Uma nova e última configuração é aplicada ao sistema 2. A arquitetura com 12 neurônios na camada oculta tem a função de tentar reduzir ainda mais erro médio quadrático no treinamento e na validação, observado nas configurações anteriores.

Apresenta-se a seguir os gráficos de saída, erro médio quadrático no treinamento e erro médio quadrático na validação do sistema 2, comparativos entre a taxa de aprendizagem fixa (RNTF) e adaptativa (RNTA), como também a evolução da taxa de aprendizagem. Com 12 neurônios na camada oculta.

O gráfico a seguir apresenta a saída da rede neural com 12 neurônios na camada oculta

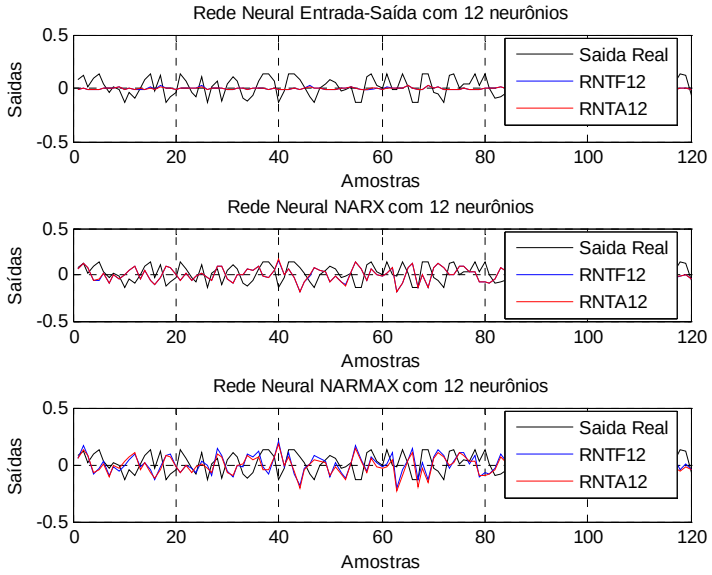

Figura 4.27 - Gráfico com a saída validada pela rede com 12 neurônios na camada oculta para o modelo entrada-saída, NARX, NARMAX (taxa de aprendizagem fixa e adaptativa)

O gráfico a seguir apresenta o erro médio quadrático no treinamento da rede neural com 12 neurônios na camada oculta

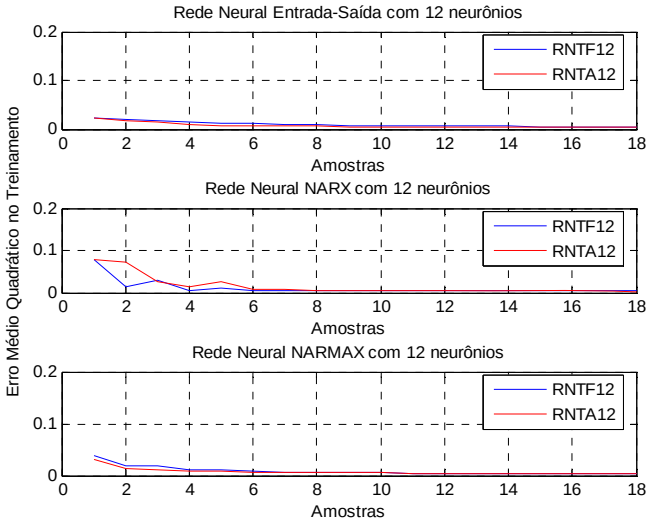

Figura 4.28 - Gráfico do erro médio quadrático no treinamento da rede com 12 neurônios na camada oculta para o modelo entrada-saída, NARX, NARMAX (taxa de aprendizagem fixa e adaptativa)

O gráfico a seguir apresenta o erro médio quadrático na validação da rede neural com 12 neurônios na camada oculta

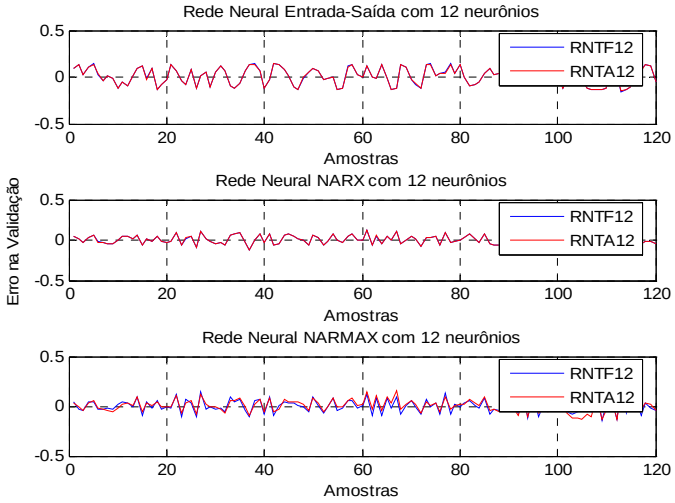

Figura 4.29 - Gráfico do erro médio quadrático na validação da rede com 12 neurônios na camada oculta para o modelo entrada-saída, NARX, NARMAX (taxa de aprendizagem fixa e adaptativa)

Apresenta-se a seguir o gráfico com a evolução da taxa de aprendizagem para a rede com 12 neurônios na camada oculta

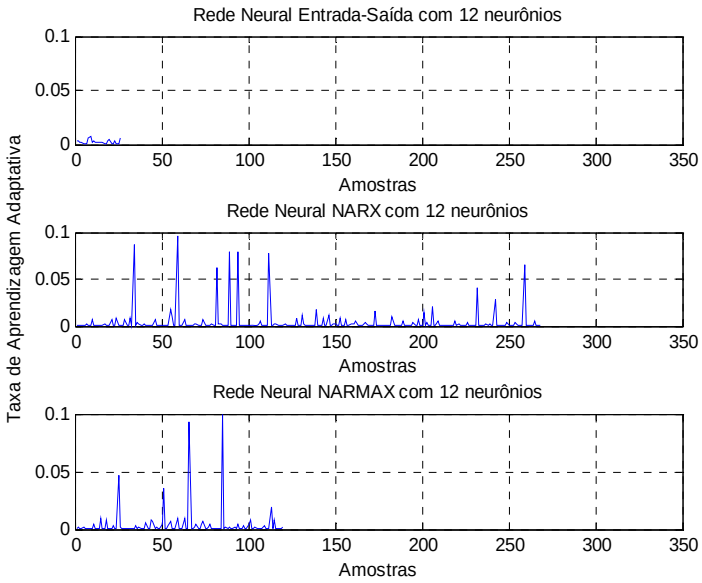

Figura 4.30 - Gráfico da evolução da taxa de aprendizagem com 12 neurônios na camada oculta para o modelo entrada-saída, NARX, NARMAX

As tabelas a seguir apresentam o número de épocas, o tempo de processamento da rede na fase de treinamento, o erro quadrático médio de treinamento e de validação para a arquitetura com 12 neurônios na camada oculta, para os modelos Entrada- Saída, NARX, NARMAX com a taxa de aprendizagem fixa e adaptativa.

Tabela 4.11 – Resumo dos Resultados obtidos pela rede para taxa de aprendizagem fixa com 12 neurônios na camada oculta.

Estrutura	Épocas	Tempo de Processamento	Erro Quadrático Médio - Treinamento	Erro Quadrático Médio - Validação
Entrada-Saída 12 neurônios camada oculta	6.0000000e+001	1.0000000e+000	5.9004882e-003	4.2641980e-003
NARX – 12 neurônios camada oculta.	1.0000000e+003	1.5000000e+001	2.2611296e-003	1.8132160e-003
NARMAX – 12 neurônios camada oculta	3.5200000e+002	5.0000000e+000	2.2781452e-003	2.3127482e-003

Tabela 4.12 – Resumo dos Resultados obtidos pela rede para taxa de aprendizagem adaptativa com 12 neurônios na camada oculta.

Estrutura	Épocas	Tempo de Processamento	Erro Quadrático Médio - Treinamento	Erro Quadrático Médio - Validação
Entrada -Saída 12 neurônios camada oculta	2.6000000e+001	7.0000000e+000	6.3821904e-003	4.2584205e-003
NARX – 12 neurônios camada oculta.	2.6800000e+002	9.0000000e+001	2.9346691e-003	1.8159506e-003
NARMAX – 12 neurônios camada oculta	2.4000000e+002	7.6000000e+001	2.4464203e-003	2.1912470e-003

As figuras 4.27 a 4.30 mostram que os resultados obtidos com a taxa de aprendizagem fixa e adaptativa possuem comportamento semelhante. Graficamente os erros médios quadráticos da taxa adaptativa tendem a ser menores no final (figura 4.28) em relação à taxa fixa. Das tabelas 4.11 e 4.12 tem-se que a menor média se dá com o modelo NARX. Observa-se ainda, que a rede neural com taxa de aprendizagem fixa utiliza tempos menores de processamento no treinamento, em contrapartida a rede neural com taxa de aprendizagem adaptativa converge sempre com o menor número de épocas.

Numa análise geral, observa-se que as diferentes configurações da rede para a representação do sistema 2 encontram soluções semelhantes. Desta forma, a configuração com 4 neurônios é a mais indicada, pois ela gera um modelo com menor complexidade e com um custo computacional menor. Observa-se também que os resultados encontrados pela rede neural com taxa fixa e adaptativa levam á resultados semelhantes nas três configurações, demonstrando que a metodologia para encontrar o valor da taxa de aprendizado está correta.

4.3.2 Sistema 3

O sistema utilizado neste teste é representado na figura 4.31.

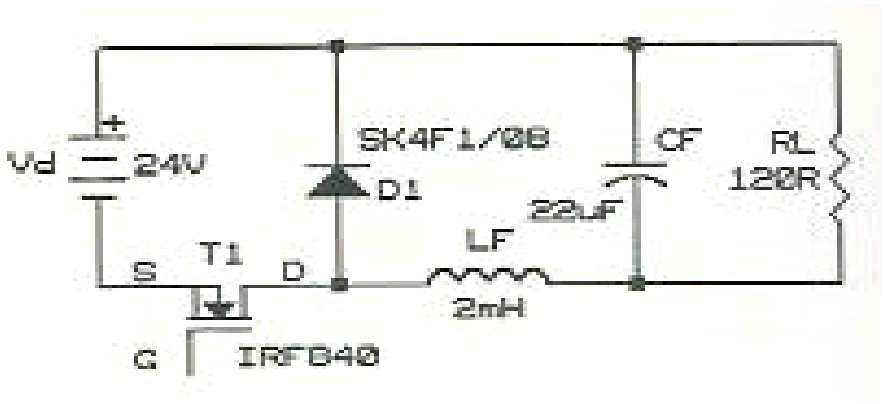

Figura 4.31 – Conversor Buck (Aguirre 2004, pág. 542)

A figura 4.32 apresenta o sinal de entrada e saída do conversor do tipo buck (Aguirre 2004, pág. 82), neste trabalho os valores estão normalizados (todos foram divididos pelo maior valor). Eles são utilizados para o procedimento de treinamento e validação para todos os testes do sistema 3.

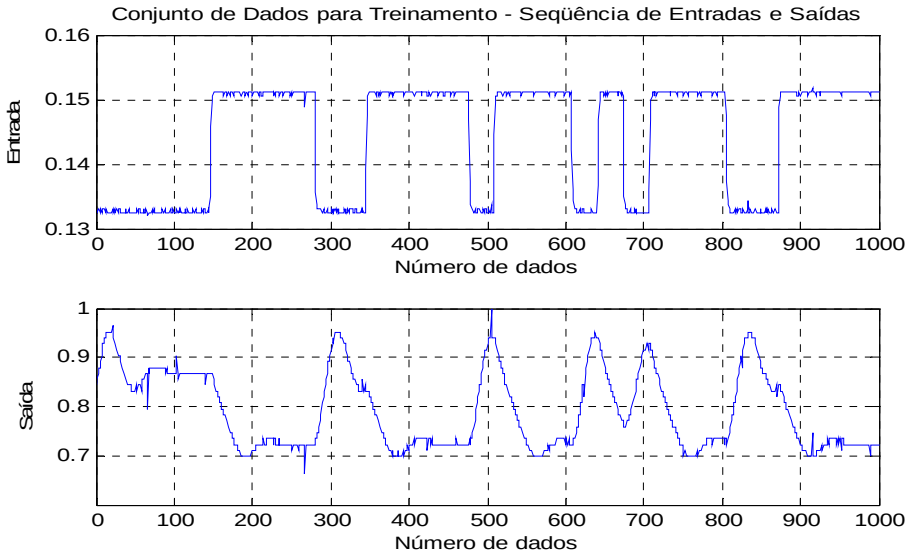

Figura 4.32 - Gráfico representativo das entradas e saídas normalizadas do conversor buck.

Apresenta-se a seguir os gráficos de saída, erro médio quadrático no treinamento e erro médio quadrático na validação do sistema 3, comparativos entre a taxa de aprendizagem fixa (RNTF) e adaptativa (RNTA), como também a evolução da taxa de aprendizagem. Com 4 neurônios na camada oculta.

O gráfico a seguir apresenta a saída da rede neural com 4 neurônios na camada oculta

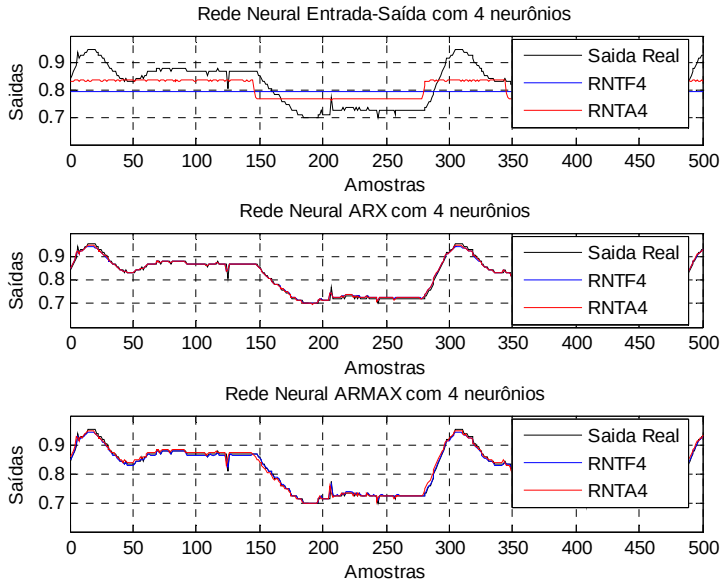

Figura 4.33 - Gráfico com a saída validada pela rede com 4 neurônios na camada oculta para o modelo entrada-saída, NARX, NARMAX (taxa de aprendizagem fixa e adaptativa)

O gráfico a seguir apresenta o erro médio quadrático no treinamento da rede neural com 4 neurônios na camada oculta

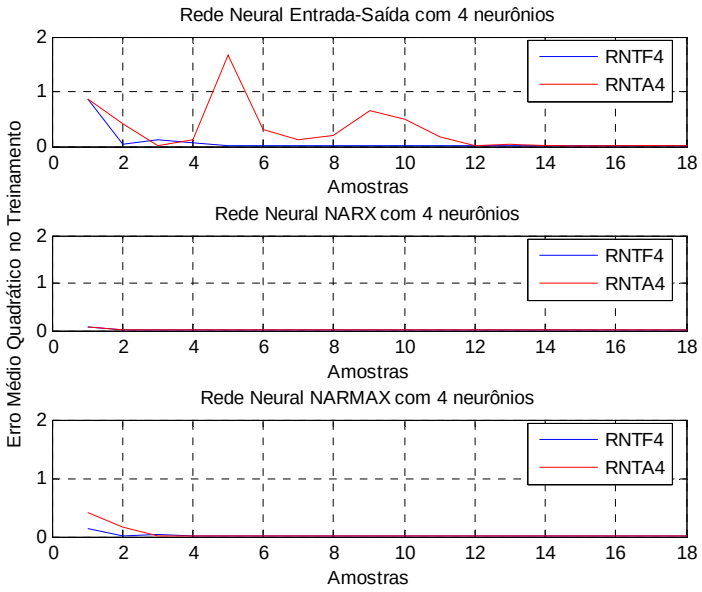

Figura 4.34 - Gráfico do erro médio quadrático no treinamento da rede com 4 neurônios na camada oculta para o modelo entrada-saída, NARX, NARMAX (taxa de aprendizagem fixa e adaptativa)

O gráfico a seguir apresenta o erro médio quadrático na validação da rede neural com 4 neurônios na camada oculta

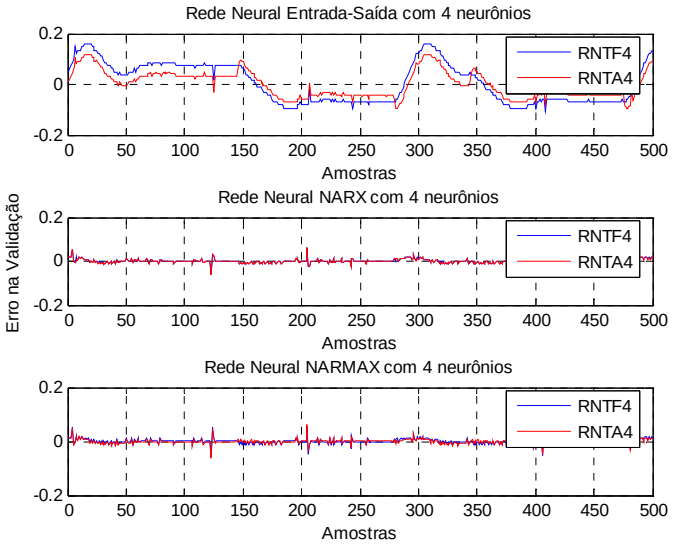

Figura 4.35 - Gráfico do erro médio quadrático na validação da rede com 4 neurônios na camada oculta para o modelo entrada-saída, NARX, NARMAX (taxa de aprendizagem fixa e adaptativa)

O gráfico a seguir apresenta a evolução da taxa de aprendizagem da rede neural com 4 neurônios na camada oculta

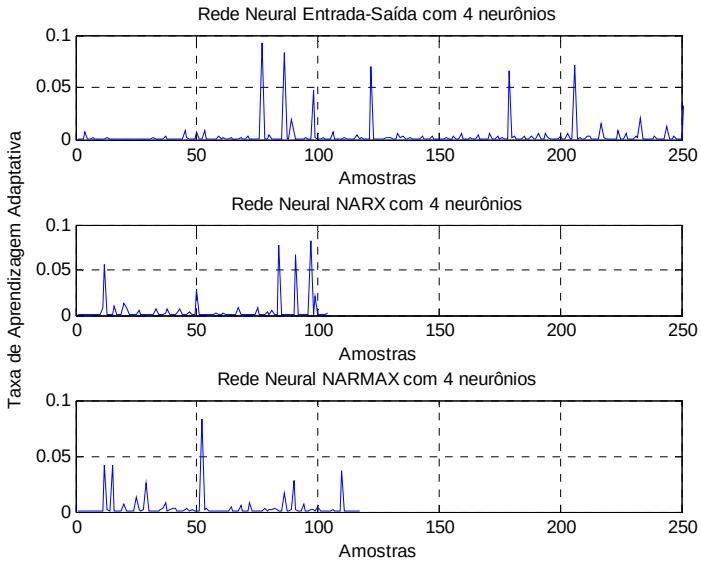

Figura 4.36 - Gráfico da evolução da taxa de aprendizagem com 4 neurônios na camada oculta para o modelo entrada-saída, NARX, NARMAX

As tabelas a seguir apresentam o número de épocas, o tempo de processamento da rede na fase de treinamento, o erro quadrático médio de treinamento e de validação para a arquitetura com 4 neurônios na camada oculta, para os modelos Entrada- Saída, NARX, NARMAX com a taxa de aprendizagem fixa e adaptativa.

Tabela 4.13 – Resumo dos Resultados obtidos pela rede para taxa de aprendizagem fixa com 4 neurônios na camada oculta.

Estrutura	Épocas	Tempo de Processamento	Erro Quadrático Médio - Treinamento	Erro Quadrático Médio - Validação
Entrada-Saída 4 neurônios camada oculta	1.0000000e+003	8.0000000e+000	4.27196620331 e-003	3.10735343403 e-003
NARX – 4 neurônios camada oculta.	5.1800000e+002	5.0000000e+000	6.675372096093345e-004	5.53854348179310 7e-005
NARMAX – 4 neurônios camada oculta	4.5900000e+002	4.0000000e+000	8.766121879037484e-004	7.02913017763877 6e-005

Tabela 4.14 – Resumo dos Resultados obtidos pela rede para taxa de aprendizagem adaptativa com 4 neurônios na camada oculta.

Estrutura	Épocas	Tempo de Processamento	Erro Quadrático Médio - Treinamento	Erro Quadrático Médio - Validação
Entrada -Saída 4 neurônios camada oculta	1.0000000e+003	1.4400000e+002	7.61495369915 e-003	1.77242231098 e-003
NARX – 4 neurônios camada oculta.	1.0400000e+002	1.5000000e+001	1.53903081995 e-003	5.41057584829 7074e-005
NARMAX – 4 neurônios camada oculta	7.4800000e+002	1.1600000e+002	4.4426169e-003	3.1779328e-003

As figuras 4.33 a 4.36 mostram que os resultados obtidos com a taxa de aprendizagem fixa e adaptativa possuem comportamento semelhante. Graficamente os erros médios quadráticos da taxa adaptativa e fixa tendem a ser iguais no final (figura 4.34). Das tabelas 4.13 e 4.14 tem-se que a menor média se dá com o modelo ARX. Observa-se ainda, que a rede neural com taxa de aprendizagem fixa utiliza tempos menores de processamento no treinamento, em contrapartida a rede neural com taxa de aprendizagem adaptativa converge com o menor número de épocas com exceção do modelo ARMAX.

O sistema 3 agora será abordado para uma nova configuração na camada oculta, esta passa a ter 8 neurônios, o objetivo é tentar obter uma melhor representação matemática do sistema, ou seja, menor erro médio quadrático no treinamento e na validação.

Apresenta-se a seguir os gráficos de saída, erro médio quadrático no treinamento e erro médio quadrático na validação do sistema 3, comparativos entre a taxa de aprendizagem fixa (RNTF) e adaptativa (RNTA), como também a evolução da taxa de aprendizagem. Com 8 neurônios na camada oculta.

O gráfico a seguir apresenta a saída da rede neural com 8 neurônios na camada oculta.

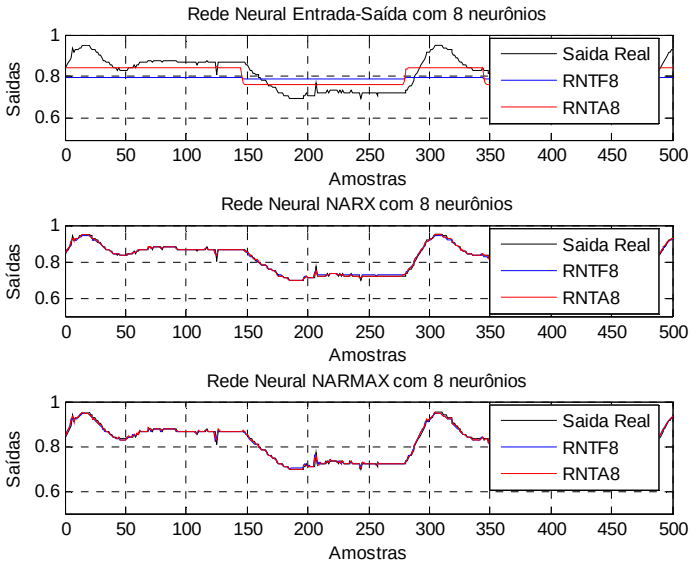

Figura 4.37 - Gráfico com a saída validada pela rede com 8 neurônios na camada oculta para o modelo entrada-saída, NARX, NARMAX (taxa de aprendizagem fixa e adaptativa)

O gráfico a seguir apresenta o erro médio quadrático no treinamento da rede neural com 8 neurônios na camada oculta

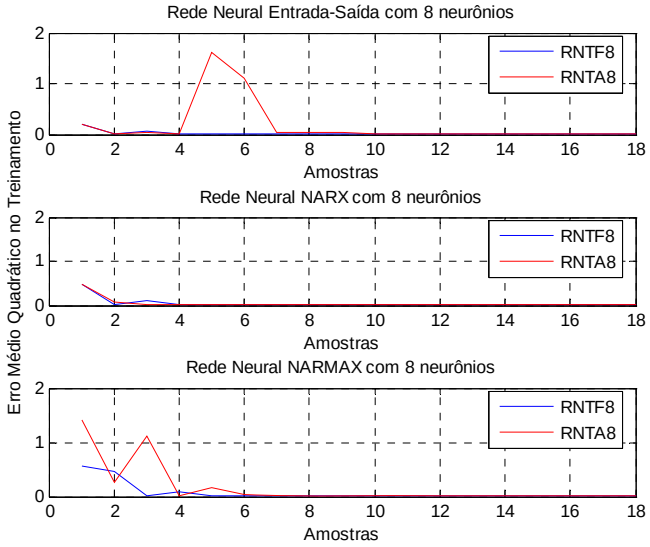

Figura 4.38 - Gráfico do erro médio quadrático no treinamento da rede com 8 neurônios na camada oculta para o modelo entrada-saída, NARX, NARMAX (taxa de aprendizagem fixa e adaptativa)

O gráfico a seguir apresenta o erro médio quadrático na validação da rede neural com 8 neurônios na camada oculta

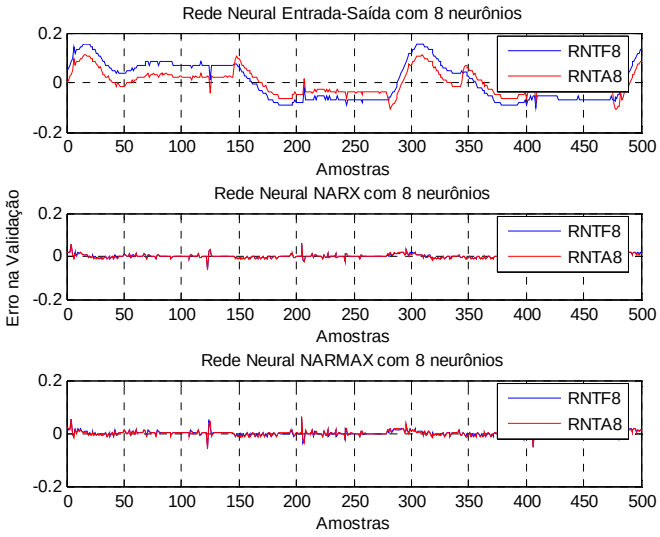

Figura 4.39 - Gráfico do erro médio quadrático na validação da rede com 8 neurônios na camada oculta para o modelo entrada-saída, NARX, NARMAX (taxa de aprendizagem fixa e adaptativa)

O gráfico a seguir apresenta a evolução da taxa de aprendizagem para a rede neural com 8 neurônios na camada oculta.

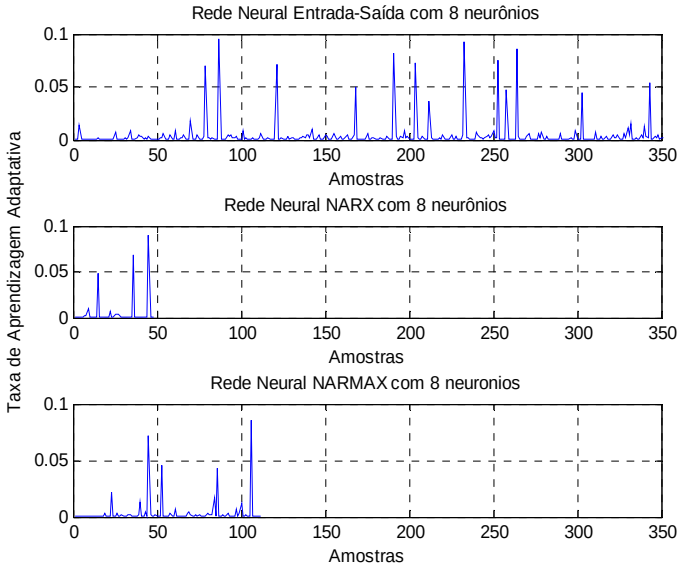

Figura 4.40 - Gráfico da evolução da taxa de aprendizagem com 8 neurônios na camada oculta para o modelo entrada-saída, NARX, NARMAX

As tabelas a seguir apresentam o número de épocas, o tempo de processamento da rede na fase de treinamento, o erro quadrático médio de treinamento e de validação para a arquitetura com 8 neurônios na camada oculta, para os modelos Entrada- Saída, NARX, NARMAX com a taxa de aprendizagem fixa e adaptativa.

Tabela 4.15 – Resumo dos Resultados obtidos pela rede para taxa de aprendizagem fixa com 8 neurônios na camada oculta.

Estrutura	Épocas	Tempo de Processamento	Erro Quadrático Médio - Treinamento	Erro Quadrático Médio – Validação
Entrada - Saída 8 neurônios camada oculta	1.0000000e+003	1.4000000e+001	3.3582805e-003	2.9578235e-003
NARX – 8 neurônios camada oculta.	2.9700000e+002	4.0000000e+000	2.4922745e-003	5.3740076e-005
NARMAX – 8 neurônios camada oculta	4.5700000e+002	5.0000000e+000	3.0290737e-003	5.7192507e-005

Tabela 4.16 – Resumo dos Resultados obtidos pela rede para taxa de aprendizagem adaptativa com 8 neurônios na camada oculta.

Estrutura	Épocas	Tempo de Processamento	Erro Quadrático Médio - Treinamento	Erro Quadrático Médio – Validação
Entrada-Saída 8 neurônios camada oculta	1.0000000e+003	2.2400000e+002	5.4408727e-003	1.6016642e-003
NARX – 8 neurônios camada oculta.	4.8000000e+001	1.2000000e+001	1.2723292e-002	5.5684611e-005
NARMAX – 8 neurônios camada oculta	1.1200000e+002	2.7000000e+001	2.7746903e-002	6.1415574e-005

As figuras 4.37 a 4.40 mostram que os resultados obtidos com a taxa de aprendizagem fixa e adaptativa possuem comportamento semelhante. Graficamente os erros médios quadráticos da

taxa adaptativa e fixa tendem a ser iguais no final (figura 4.38). Das tabelas 4.15 e 4.16 tem-se que a menor média se dá com o modelo NARX. Observa-se ainda, que a rede neural com taxa de aprendizagem fixa utiliza tempos menores de processamento no treinamento, em contrapartida a rede neural com taxa de aprendizagem adaptativa converge sempre com o menor número de épocas.

Uma nova e última configuração é aplicada ao sistema 3. A arquitetura com 12 neurônios na camada oculta tem a função de tentar reduzir ainda mais erro médio quadrático no treinamento e na validação, observado nas configurações anteriores.

Apresenta-se a seguir os gráficos de saída, erro médio quadrático no treinamento e erro médio quadrático na validação do sistema 3, comparativos entre a taxa de aprendizagem fixa (RNTF) e adaptativa (RNTA), como também a evolução da taxa de aprendizagem. Com 12 neurônios na camada oculta.

O gráfico a seguir apresenta a saída da rede neural com 12 neurônios na camada oculta

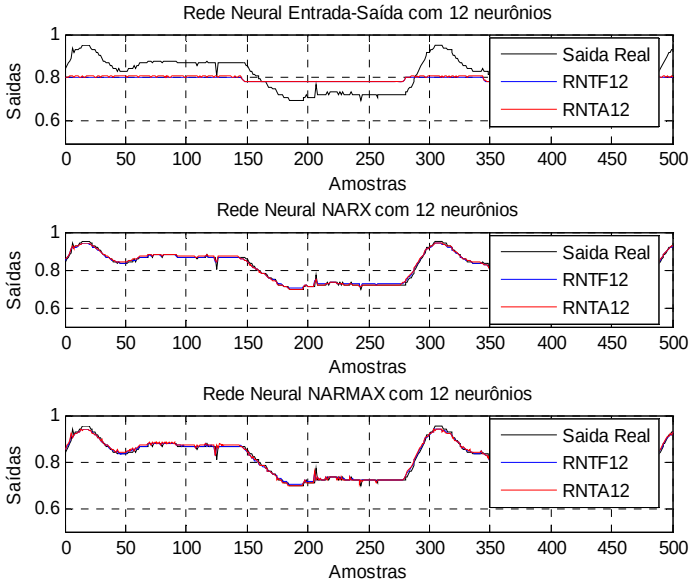

Figura 4.41 - Gráfico com a saída validada pela rede com 12 neurônios na camada oculta para o modelo entrada-saída, NARX, NARMAX (taxa de aprendizagem fixa e adaptativa)

O gráfico a seguir apresenta o erro médio quadrático no treinamento da rede neural com 12 neurônios na camada oculta

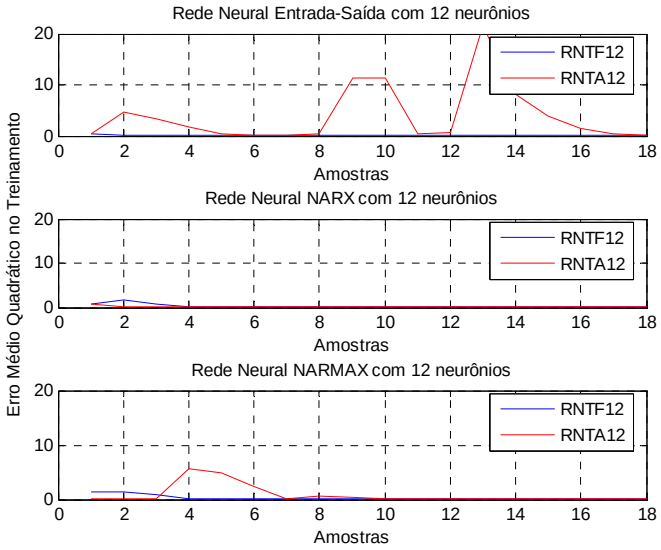

Figura 4.42 - Gráfico do erro médio quadrático no treinamento da rede com 12 neurônios na camada oculta para o modelo entrada-saída, NARX, NARMAX (taxa de aprendizagem fixa e adaptativa)

O gráfico a seguir apresenta o erro médio quadrático na validação da rede neural com 12 neurônios na camada oculta

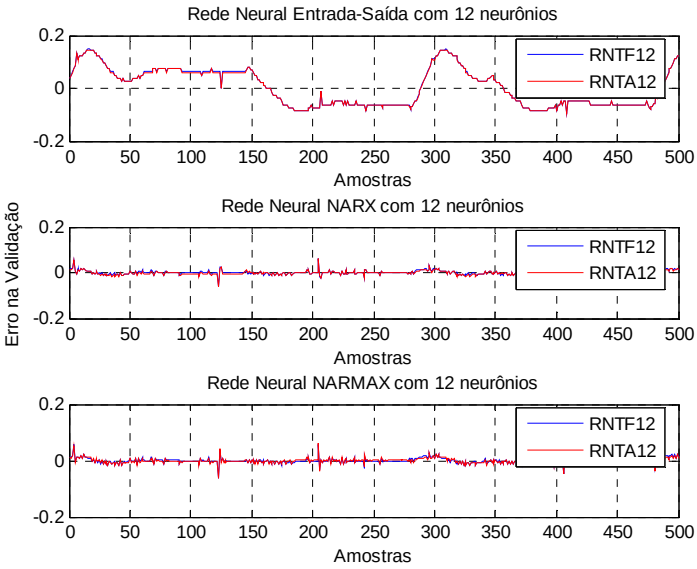

Figura 4.43 - Gráfico do erro médio quadrático na validação da rede com 12 neurônios na camada oculta para o modelo entrada-saída, NARX, NARMAX (taxa de aprendizagem fixa e adaptativa)

O gráfico a seguir apresenta a evolução da taxa de aprendizagem da rede neural com 12 neurônios na camada oculta

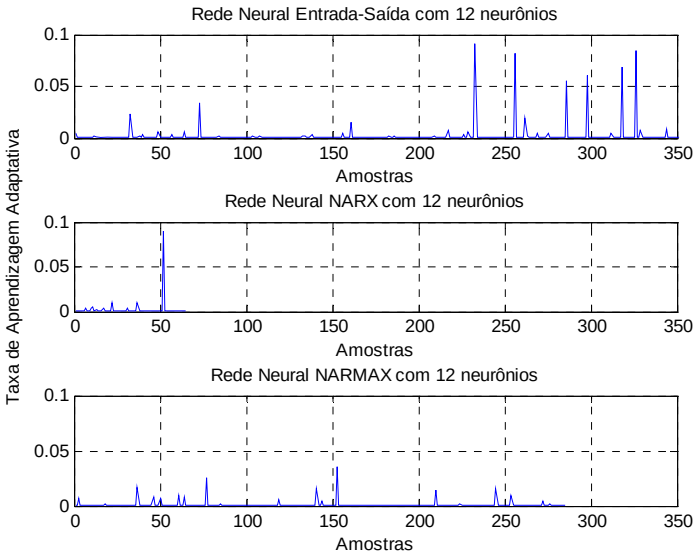

Figura 4.44 - Gráfico da evolução da taxa de aprendizagem com 12 neurônios na camada oculta para o modelo entrada-saída, NARX, NARMAX

As tabelas a seguir apresentam o número de épocas, o tempo de processamento da rede na fase de treinamento, o erro quadrático médio de treinamento e de validação para a arquitetura com 12 neurônios na camada oculta, para os modelos Entrada- Saída, NARX, NARMAX com a taxa de aprendizagem fixa e adaptativa.

Tabela 4.17 – Resumo dos Resultados obtidos pela rede para taxa de aprendizagem fixa com 12 neurônios na camada oculta.

Estrutura	Épocas	Tempo de Processamento	Erro Quadrático Médio - Treinamento	Erro Quadrático Médio - Validação
Entrada-Saída 12 neurônios camada oculta	1.0000000e+003	1.8000000e+001	3.4039677e-003	2.5953076e-003
NARX – 12 neurônios camada oculta.	3.2200000e+002	8.0000000e+000	1.1968592e-002	5.0118625e-005
NARMAX – 12 neurônios camada oculta	3.3400000e+002	6.0000000e+000	1.5328003e-003	1.2033847e-004

Tabela 4.18 – Resumo dos Resultados obtidos pela rede para taxa de aprendizagem adaptativa com 12 neurônios na camada oculta.

Estrutura	Épocas	Tempo de Processamento	Erro Quadrático Médio - Treinamento	Erro Quadrático Médio - Validação
Entrada Saída 12 neurônios camada oculta	1.0000000e+003	3.6900000e+002	7.3614526e-002	2.5700128e-003
NARX – 12 neurônios camada oculta.	6.5000000e+001	2.5000000e+001	1.5407158e-002	4.9248286e-005
NARMAX – 12 neurônios camada oculta	2.8500000e+002	1.3900000e+002	5.0071524e-002	5.7764179e-005

As figuras 4.41 a 4.44 mostram que os resultados obtidos com a taxa de aprendizagem fixa e adaptativa possuem comportamento semelhante. Graficamente os erros médios quadráticos da taxa adaptativa e fixa tendem a ser iguais no final (figura 4.42). Das tabelas 4.17 e 4.18 tem-se que a menor média se dá com o modelo NARX. Observa-se ainda, que a rede neural com taxa de aprendizagem fixa utiliza tempos menores de processamento no treinamento, em contrapartida a rede neural com taxa de aprendizagem adaptativa converge sempre com o menor número de épocas.

Para este sistema, observa-se que as diferentes configurações da rede para a sua representação encontram soluções semelhantes. Desta forma, a configuração com 4 neurônios é a mais indicada, pois ela gera um modelo com menor complexidade e com um custo computacional menor. Observa-se também que os resultados encontrados pela rede neural com taxa fixa e adaptativa levam á resultados semelhantes nas três configurações, demonstrando que a metodologia para encontrar o valor da taxa de aprendizado está correta.

CONCLUSÃO

Neste trabalho foi apresentada uma Rede Neural perceptron multicamadas, treinada pelo algoritmo back-propagation com taxa de aprendizagem variável (adaptativa), a qual teve seus resultados comparados com uma rede idêntica, porém com taxa de aprendizagem fixa.

A taxa de aprendizagem fixa foi determinada após vários testes, e seu coeficiente foi fixado em 0.1, pois foi o valor que apresentou melhor relação erro médio quadrático por tempo de processamento.

Para a obtenção da taxa de aprendizagem variável foi empregado o algoritmo genético, esta escolha foi em função desta técnica fazer uma busca paralela em cada passo no espaço de soluções e ir convergindo para regiões de prováveis soluções ótimas. Os resultados apresentados nos gráficos de evolução da taxa de aprendizagem apresentam variações entre 0 e 0.1 tendo sido estabelecido um espaço de busca de soluções entre 0 e 1. Observa-se que o limitante superior é igual ao valor utilizado na taxa de aprendizagem fixa.

Observando o número de iterações vê-se que a rede neural com taxa variável converge sempre com um número menor de iterações, uma vez que a variação da taxa acelera a convergência sem as oscilações indesejáveis.

Analisando os tempos de processamento das redes neurais com taxa de aprendizagem fixa e variável, observa-se que os tempos de processamento com a taxa variável são maiores. Mas o tempo de processamento para se achar a taxa fixa não está incluso, o que nos leva, quando somado estes tempos, a um valor de tempo muitas vezes maior do que o tempo de processamento da taxa variável.

Comparando os erros médios quadráticos de treinamento e validação da rede neural com taxa de aprendizagem fixa e variável, observa-se que eles são bastante semelhantes em valores. Graficamente visualiza-se que a rede neural com taxa de aprendizagem variável na maioria das vezes apresenta, tanto na etapa do treinamento como na validação, um erro inicial maior, mas conforme o número de iterações vai aumentando o erro vai diminuindo tendo-se um erro final menor que o erro final gerado com a taxa de aprendizagem fixa. Isto demonstra

que a metodologia proposta para determinar o coeficiente de aprendizagem esta correta e é capaz de assegurar um bom desempenho.

Das diferentes topologias de Redes Neurais que foram aplicadas aos sistemas não-lineares com o objetivo de obter modelos capazes de representá-los satisfatoriamente no domínio do tempo, observa-se que na média geral o modelo NARX foi o que melhor representou os sistemas estudados.

Para os três sistemas estudados observa-se que a configuração com 4 neurônios na camada oculta é a mais indicada, pois ela gera bons resultados sendo um modelo com menor complexidade e com um custo computacional menor.

Analisando os resultados obtidos, observa-se que o objetivo deste trabalho foi atingido, pois ele apresenta uma técnica de obtenção de forma automática para a taxa de aprendizagem, eliminando uma etapa tediosa do treinamento de redes neurais que é a procura do valor (coeficiente) da taxa para seu uso posterior.

De um modo geral, as redes neurais artificiais se constituem numa boa técnica de estimação de parâmetros para modelos não-lineares, com possibilidade de aplicações nas mais diversas áreas do conhecimento.

Como trabalho futuro sugere-se o estudo também de um coeficiente variável para o momentum que neste trabalho foi considerado constante.

REFERÊNCIAS BIBLIOGRÁFICAS

AGUIRRE, Luiz Antônio. **Introdução à identificação de sistemas: técnicas lineares e não lineares aplicadas a sistemas reais**. 2. ed. Ver. e ampl. Belo Horizonte: Ed. UFMG, 2004.

AGUIRRE, L. A., RODRIGUES, G. G., JÁCOME, C.R.F. Identificação de Sistemas não lineares utilizando modelos narmax polinomiais – uma revisão e novos resultados. **Controle e Automação**, (1998b) 9(2): 90-106.

ASSIS, A. J. **Identificação e controle de Processos não-lineares utilizando redes neurais artificiais.** Tese de Doutorado **-** Unicamp, Campinas, Brasil 2001.

BASSANEZI, R. C. **Ensino-aprendizagem com modelagem matemática**. São Paulo: Contexto, 2002.

BILLINGS, S. A. Identification of nonlinear systems – a survey. **In proë. IEE, Part D**, pages 272-285, 1980.

BOMBERGERR, J. D., D. E. SEBORG, and B. A. Ogunnaike, RBFN identification of a solution copolymerization model. **Proc. of the 1998 *ACC***, June 24-26, Philadelphia, PA (1998).

CHEN, S. & BILLINGS, S. A. Recursive prediction error parameter estimation for non-linear models. **International Journal of Control**, vol. 49, no. 2, pp. 569-594, 1989.

CAMPOS, M. M de e SAITO, K. **Sistemas Inteligentes em controle e Automação de Processos.** Rio de Janeiro: Editora ciência Moderna Ltda, 2004.

CYBENKO, G. Approximation by superpositions of a sigmoidal function, **Mathematics for Control, Signals and Systems** 2: 303–314, 1989.

DAVIS, M. e VINTER, R. **Stochastic Modelling and Control:** Chapman and Hall. 1985.

DE MORAIS LIMA, C. A**. Emprego de Teoria de Agentes no Desenvolvimento de Dispositivos Neurocomputacionais Híbridos e Aplicação ao Controle e Identificação de Sistemas Dinâmicos**, Tese de Mestrado, FEEC – Unicamp, Campinas, Brasil 2000.

FERREIRA, W. P. "**Análise Dinâmica de Contingências de Sistemas de Energia Elétrica por Redes Neurais Baseadas na Teoria da Ressonância Adaptativa**". Tese de Doutorado – UNESP, São Paulo, Brasil. 2003.

GARCIA, C. **Modelagem e simulações de processos industriais e de sistemas eletromecânicos.** São Paulo: EDUSP. 1997.

GOLDBERG, David E. **Genetic algorithms in search, optimization, and machine learning**. Addison Wesley Longman: Inc, 1989.

HAYKIN, S. **Redes Neurais Princípios e Práticas**. 2a. Edição. Porto Alegre: Editora Bookman, 2001.

HUAMANÍ, I. R. L, **Redes Neurais Fuzzy Aplicadas em Identificação e Controle de Sistemas.** Dissertação de Mestrado - Unicamp, Campinas, Brasil, 2003.

JORGENSEN, S. B. HANGOS, K. M. Grey box modelling for control: Qualitative models as a unifying framework, **Int. J. of Adaptative Control and Signal Processing,** 9(6): 547-562. 1995

KOVÁCS, Z. L. **Redes Neurais Artificiais:** Fundamentos e Aplicações. Um Texto Básico. 2ª Edição. São Paulo: Edição Acadêmica, 1996.

KOVÁCS, Z. L. **O cérebro e sua Mente:** Uma introdução à Neurociência computacional. São Paulo: Edição Acadêmica, 1997.

LEONTARITIS, I. And BILLINGS, S. Input-output parametric models for non-linear systems part I: deterministic non-linear systems. **Int. J. Control**, 41(2): 329-344, (1985a).

LJUNG, Lennart. **System Identification:** Theory for the user. Englewood Cliffs, 2nd edition New Jersey: Prentice-Hall, 1997.

LJUNG, Lennart. **System Identification:** Theory for the user. Englewood Cliffs, 2nd edition New Jersey: Prentice-Hall, 1999.

MICHALEWICZ, Z. **Genetic Algorithms + Data Structures = Evolution Programs**. Springer Verlag, 1999.

NARENDRA, K. e PARTHASARATHY, K. Identification and control of dinamical system using neural networks. **IEEE Transactions on Neural Networks,** vol. 1, No. 1, pp. 4-27, 1990.

NERRAND, O., ROUSSEL-RAGOT, P.,URBANI, D., PERSONNAZ, L. & DREYFUS, G. Training Recurrent Neural Networks why and How? An Illustration in Dynamical Process Modeling, **IEEE Transactions on Neural Networks** 5(2): 178-184, 1994.

POTTMANN, M. And PEARSON, R. K. Block-oriented NARMAX models with output multiplicities. **AIChE Journal**, 41(1): 131-140, 1998.

PRÖL, T. and KARIM, M. Model-predictive pH control using realtime NARX approach. **AIChE Journal**, 40(2):269-282, 1994.

RICH,E. e KNIGHT, K **Inteligência Artificial.** 2ª edição. São Paulo: Makron Books, 1993.

RUSSELL, S e NORVIG, P **Inteligência Artificial.** Tradução da segunda edição de publiCare Consultoria. Rio de Janeiro: Elsevier, 2004.

WIDROW, B. & Ho., M. Adaptive Switching Circuits, **Wescon Conv**. Rec. pp. 96–140, 1960.

Bibliografia pesquisada

BARCELLOS, João C. H. **Algoritmos Genéticos Adaptativos:** Um estudo comparativo. São Paulo: Dissertação Mestrado - Escola Politécnica da Universidade de São Paulo, São Paulo, 2000.

BARONE, D.A.C., BORGES, N.C.K, COELHO, J. e SCHNEIDER, A.M. Algoritmo Genético no controle da trajetória de um veículo. Florianópolis: **III Congresso Brasileiro de Redes Neurais**, 1997.

BHARATH, R. **Neural Network Computing.** Includes Index, United States of America, 1994.

BOLTON, W. **Engenharia de Controle**. Traduzido por Valcere V. Rocha e Silva. São Paulo: Makron Books, 1995.

COELHO, L. S. e COELHO, A. A. R. Algoritmos Evolutivos em identificação e controle de processos: uma visão integrada e perspectivas, **SBA controle & automação**, v.10, n.01, pp. 13-30, 1999.

COELHO, L. S. e COELHO, A. A. R. Estudo comparativo de configurações de Algoritmos Genéticos aplicados à identificação de processo multivariável. Florianópolis: **XIII Congresso Brasileiro de Automática**, 2000.

FARDIN, J.F., ARRUDA, L.V.R. & AMARAL, W.C. Identificação relevante para controle preditivo utilizando Algoritmo Genético. Vitória: **3° Simpósio Brasileiro de automação inteligente**, 1997.

FÁVARO, S. **Estimação de pólos e zeros de modelos utilizando algoritmos genéticos**. Dissertação de Mestrado – CEFET, Paraná, 1999.

HOLLAND, J.H. **Adaptation in Natural and Artificial System**. The University of Michigan Press, Ann Arbor, 2nd edition, 1975.

JOHANSSON, R. **System Modeling Identification**. Englewood Cliffs, New Jersey: Prentice-Hall, 1993.

KRISTINSSON, K. e DUMONT, G. A. System Identification and control using genetic algorithms, **IEEE Trans. On Sys. Man, and Cybernetics**, n.22, pp. 1033-1046, 1992.

LI, Y. Modern Information Technology for Control Systems Design and Implementation. **Proc. Second Asiapacific Conference on Control and Measurement**, Wuhan-Chongping, China, 1995.

MATSUMOTO, É, Y. **Matlab 6.5:** fundamentos de programação. São Paulo: Érica, 2002.

OGATA, K. **Engenharia de controle moderno**. 3 ed. Rio de Janeiro: Livros Técnicos e Científicos Editora S.A., 1998.

SILVA, W. L de A. **Métodos de identificação de sistemas auxiliados por computador**. Campina Grande: UFPB, 1997. Dissertação (Mestrado em Engenharia Elétrica), Universidade Federal da Paraíba, 1997.

TAN, K. C., LI, Y., MURRAY-SMITH, D.J. e SHARMAN, K. C. System Identification and Linearization using Genetic Algorithms with Simulated Annealing, **Proc. IEE/IEEE Int. Conf. On GA in Eng. Syst.: Innovations and Applications**, Sheffield, U.K., pp. 164-169, 1995.

TANOMARU, J. Motivação, fundamentos e aplicações de Algoritmos Genéticos. Curitiba: **II Congresso Brasileiro de Redes Neurais**, 1995.

VELLOSO, M.L., MENDONÇA, J.M., PACHECO, M.A. e VELLASCO, M.M.B.R. Otimização de planejamento de horários por Algoritmos Genéticos. Curitiba: **II Congresso Brasileiro de Redes Neurais**, 1995.

Livros Grátis

(http://www.livrosgratis.com.br)

Milhares de Livros para Download:

Baixar livros de Literatura
Baixar livros de Literatura de Cordel
Baixar livros de Literatura Infantil
Baixar livros de Matemática
Baixar livros de Medicina
Baixar livros de Medicina Veterinária
Baixar livros de Meio Ambiente
Baixar livros de Meteorologia
Baixar Monografias e TCC
Baixar livros Multidisciplinar
Baixar livros de Música
Baixar livros de Psicologia
Baixar livros de Química
Baixar livros de Saúde Coletiva
Baixar livros de Serviço Social
Baixar livros de Sociologia
Baixar livros de Teologia
Baixar livros de Trabalho
Baixar livros de Turismo