

**ROBERTO CÉLIO LIMÃO DE OLIVEIRA**

**REDE NEURAL COM DINÂMICA INTERNA APLICADA  
A PROBLEMAS DE IDENTIFICAÇÃO E CONTROLE  
NÃO-LINEAR**

**FLORIANÓPOLIS  
1999**

**UNIVERSIDADE FEDERAL DE SANTA CATARINA**  
**PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA**  
**ELÉTRICA**

**REDE NEURAL COM DINÂMICA INTERNA APLICADA**  
**A PROBLEMAS DE IDENTIFICAÇÃO E CONTROLE**  
**NÃO-LINEAR**

Tese submetida à Universidade Federal de Santa Catarina como parte dos  
requisitos para a obtenção do grau de Doutor em Engenharia Elétrica

**ROBERTO CÉLIO LIMÃO DE OLIVEIRA**

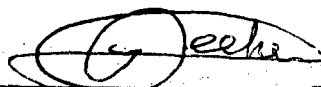
Florianópolis, Maio de 1999

# REDE NEURAL COM DINÂMICA INTERNA APLICADA A PROBLEMAS DE IDENTIFICAÇÃO E CONTROLE NÃO-LINEAR

'Esta Tese foi julgada adequada para obtenção do Título de Doutor em Engenharia Elétrica, Área de Concentração em Sistemas de Informação, e aprovada em sua forma final pelo Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal de Santa Catarina'



Fernando Mendes de Azevedo, D. Sc.  
Orientador

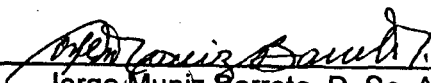


Ilidemar Cassana Decker, D. Sc.  
Coordenador do Programa de Pós-Graduação em Engenharia Elétrica

Banca examinadora:



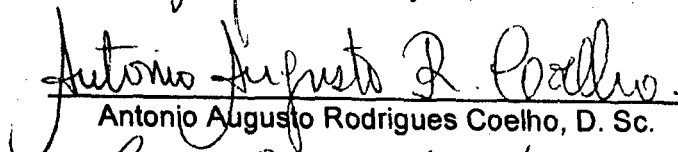
Fernando Mendes de Azevedo, D. Sc.  
Presidente



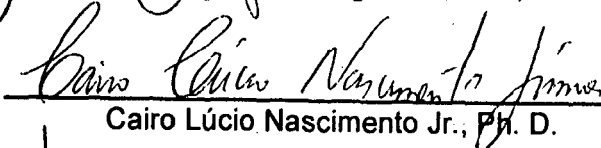
Jorge Muniz Barreto, D. Sc. A.



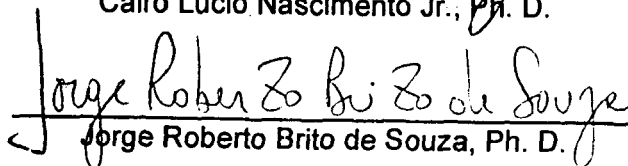
Renato Garcia-Ojeda, D. Sc.



Antonio Augusto Rodrigues Coelho, D. Sc.



Cairo Lúcio Nascimento Jr., Ph. D.



Jorge Roberto Brito de Souza, Ph. D.

## Dedicatória

*A meus pais pela criação e educação a mim proporcionada, em especial a minha mãe pela importância dada aos estudos.*

*A Kátia, Brenda e Tiago pela paciência e privações suportadas, em especial a minha esposa Kátia pelo apoio e dedicação a mim despendido.*

## **Agradecimentos**

Aos Professores Fernando Mendes de Azevedo e Jorge Muniz Barreto, orientador e co-orientador, respectivamente, deste trabalho, pela dedicação, e amizade compartilhadas neste trabalho, além das sugestões e discussões surgidas durante o transcurso do mesmo.

Aos Professores Renato, Savi, Jefferson, Raimes, Antonio Coelho , aos amigos Marcus Aurélio, Miguel, Fernanda, Kathya, Vânia, Lourdes, Lúcio, Marcos Vinícius e demais colegas de pós-graduação da Universidade Federal de Santa Catarina, principalmente do Grupo de Pesquisas em Engenharia Biomédica, que me proporcionaram uma agradável convivência durante a minha permanência na UFSC.

A CAPES-PICD e UFPa, pela concessão dos recursos necessários para a realização do curso.

Resumo da Tese apresentada à UFSC como parte dos requisitos necessários para o  
obtenção do grau de Doutor em Engenharia Elétrica

## **REDE NEURAL COM DINÂMICA INTERNA APLICADA A PROBLEMAS DE IDENTIFICAÇÃO E CONTROLE NÃO- LINEAR**

**Roberto Célio Limão de Oliveira**

Maio/1999

Orientador : Fernando Mendes de Azevedo, D. Sc.

Área de Concentração : Sistemas de Informação

Palavras-chave : redes neurais, dinâmica interna, não-linear, identificação, controle

Número de Páginas : 160

Este trabalho trata da aplicação de um novo modelo de rede neural artificial recorrente discreto em problemas de identificação e controle de processos dinâmicos não-lineares. A identificação neural é realizada através do método série-paralelo com a rede neural servindo como modelo direto do processo. O controle neural utilizado é do tipo *feedforward* com o modelo neural encontrando o modelo inverso do processo e servindo como um controlador *feedforward*.

A partir de uma definição formal do neurônio artificial, que separa função de ativação responsável pela dinâmica (linear ou não-linear) e função de saída

estática (linear ou não-linear), é possível desenvolver um novo modelo de rede neural. Este novo modelo neural contém uma parte dinâmica linear e uma parte estática não-linear. A parte dinâmica linear é representada através dos estados dos neurônios dinâmicos lineares (saída da função de ativação) e a parte estática não-linear é representada através do sinal de saída dos neurônios estáticos (saída da função de saída). A grande vantagem desta rede neural é o estudo de estabilidade da mesma que emprega técnicas clássicas de sistemas lineares.

Esta rede neural é empregada para identificar e/ou controlar processos não-lineares orientados à blocos que são os modelos de Wiener, de Hammerstein, de Wiener-Hammerstein e de Hammerstein-Wiener. Estes modelos também apresentam partes dinâmicas lineares e partes estáticas não-lineares. A rede também é empregada para identificar e/ou controlar processos não-lineares que não são orientados à blocos. Na tarefa de identificação neural não-linear utiliza-se a informação do erro de predição na entrada do modelo neural em situações em que existe o ruído de medida não-linear na saída do processo identificado.

A arquitetura de controle utilizada faz uso de um controlador neural *feedforward* em paralelo com um controlador clássico do tipo PID, com os dois tipos de controladores em cascata com o processo. O PID tem como entrada o sinal de erro entre uma saída desejada para o processo e a saída do processo. O controlador neural tem como entrada a saída desejada para o processo; o ajuste dos seus parâmetros é feito a partir do sinal de saída do controlador PID. O sinal de controle geral aplicado ao processo é composto da soma do sinal de saída do controlador PID e do sinal de saída da rede neural. O treinamento do controlador neural tem como objetivo minimizar a influência do sinal de controle gerado pelo PID, maximizando a influência do sinal de controle gerado pela rede neural.

Abstract of Thesis presented to UFSC as a partial fulfillment of the requirements for  
the degree of Doctor in Electrical Engineering

# **NEURAL NETWORKS WITH INTERNAL DYNAMICS APPLIED TO NON-LINEAR IDENTIFICATION AND CONTROL PROBLEMS**

**Roberto Célio Limão de Oliveira**

May/1999

Advisor : Fernando Mendes de Azevedo, D. Sc.

Area of Concentration : Information Systems

Keywords : neural networks, internal dynamics, non-linear, identification, control

Number of Pages : 160

This work concern the application of a new discret recurrent neural net model in identification and/or control problems of non-linear dynamic processes. The identification neural is accomplished by the series-parallel method with the neural net serving as direct model of the process. The neural control used is of the type feedforward with the model neural finding the inverse model of the process and serving as a feedforward controller.



Starting from a formal definition of the artificial neuron, that separates activation function responsible for the dynamics (linear or non-linear) and static output function (linear or non-linear) it is possible to develop a new neural net model. This new neural model contains a linear dynamic part and a non-linear static part. The linear dynamic part is represented through the states of the linear dynamic neurons (output of the activation function) and the non-linear static part is represented through of the output signal of the static neurons (output of the output function). The great advantage of this neural net is the study of stability of the same that uses classic techniques of linear systems.

This neural net is used to identify and/or to control no-linear block-oriented processe that are the models of Wiener, of Hammerstein, of Wiener-Hammerstein and of Hammerstein-Wiener. These models also present linear dynamic parts and non-linear static parts. The neural net is also used to identify and/or to control non-linear processes that are not block-oriented. In the task of non-linear neural identification the information of the error prediction is used in the input of the neural model in situations where there is non-linear measure noise in the output of the identified process.

The used control architecture makes use of a feedforward neural controller in parallel with a classic controller of the type PID with the two types of controllers in cascade with the process. The PID has as input the error sign between a output wanted and the output process. The neural controller has as input the output wanted and the adjustment of its parameters is made from the signal of output PID controller. The training of the neural controller has as objective to minimize the influence of the signal control generated by PID and to maximize the influence of the signal control generated by the neural net.

## 1 - Introdução

A Teoria de Controle tem evoluído durante a sua história sempre no sentido de desenvolver controladores autônomos para processos reais, que possam se adaptar automaticamente as modificações ocorridas no ambiente. Nesta situação, normalmente, não se conhece totalmente o processo, o mesmo contém não-linearidades inerentes a qualquer sistema do mundo real e apresenta variações paramétricas. A maior dificuldade encontrada na solução deste tipo de problema são as não-linearidades. Para o caso de sistemas lineares, o controle adaptativo linear apresenta soluções bem consolidadas. Entretanto, para o caso não-linear o mesmo não acontece.

Com o objetivo de estudar de uma forma geral o problema do controle não-linear, desde o final da década de oitenta, tem-se utilizado o paradigma conexionista das Redes Neurais Artificiais (RNA's) no controle de processos não-lineares (*IEEE Control Systems Magazine*, 1988, 1989, 1990). A motivação para o uso desta nova ferramenta foi a sua facilidade em realizar complexos mapeamentos não-lineares e a sua capacidade de aprendizado (Irwin et al., 1995).

Junto com as RNA's apareceram outras ferramentas computacionais para tratar problemas complexos em controle, como por exemplo a lógica fuzzy, técnicas da computação evolutiva e os sistemas especialistas. Basicamente estas quatro ferramentas computacionais, RNA, lógica fuzzy, computação evolutiva e sistemas especialistas, formam a base de desenvolvimento do denominado Controle Inteligente (Antsaklis, 1994). Este novo ramo da teoria de controle, por definição, trata de processos complexos, onde não se tem o modelo matemático dos mesmos para gerar o controlador explicitamente.

Sistemas complexos, na sua maior abrangência, envolvem não apenas não-linearidades, mas, também, processos que necessitam de procedimentos e variáveis declarativas para a sua descrição, adaptação e/ou aprendizado as mudanças

ocorridas no ambiente e nos parâmetros dos mesmos, além de procedimentos de otimização extremamente complexos. A seguir, será mostrado um histórico da teoria de controle, com o início nos primeiros mecanismos controlados, conhecidos pela civilização ocidental, e terminando no controle inteligente.

## 1.1 Breve Histórico da Teoria de Controle

A história da Teoria de Controle tem registros de processos controlados desde os tempos dos faraós do Egito, onde existia o chamado dispositivo de Hero (Paraskevopoulos, 1996). Este dispositivo era o portão de um templo que se fechava ou abria de acordo com a presença ou ausência de uma chama em um altar. Já no século quinze, a mesma técnica de "controle em malha-aberta" era utilizada em algumas máquinas primitivas.

A partir de 1788, com o controle de caldeiras desenvolvido por James Watt para uso em locomotivas (D'Azzo e Houpis, 1988), iniciou-se a utilização do "controle em malha-fechada", o qual era baseado no sinal de erro que determinava o quanto a saída do processo estava distante do valor desejado. Ou seja, o uso do controle automático deu o primeiro passo da sua evolução. Neste estágio, pode-se dizer que existe o primeiro grau de inteligência do controlador.

No século dezenove, Maxwell em 1868 (Maxwell, 1868) desenvolveu o primeiro trabalho matemático em controle com aplicação na caldeira de Watt (prova da estabilidade do controlador de Watt). Nesta época, os trabalhos em controle de processos eram mais "arte" do que ciência, devido aos poucos resultados teóricos existentes. Um dos trabalhos mais significantes deste período foi o critério de estabilidade de Routh (Routh, 1877) desenvolvido em paralelo com Hurwitz (Bissel, 1992), sem que um tivesse o conhecimento sobre o trabalho do outro.

O início do século vinte marca uma transição da "arte" para a ciência. Neste começo de século a matemática é mais utilizada na formação da base dos estudos da Teoria de Controle, com ênfase no uso de equações diferenciais. O intervalo entre

1930 e 1940 é destaque na história do controle de processos devido aos importantes trabalhos teóricos e práticos de Nyquist (Nyquist, 1932), Black (Black, 1934) e do alemão Oppelt (Bissel, 1992), este último com a teoria e uso da função descritiva.

De 1940 até 1956, outros resultados significantes foram produzidos, podendo-se citar os trabalhos de Ziegler e Nichols (Ziegler e Nichols, 1942), Bode (Bode, 1945), Wiener (Wiener, 1949) e Evans (Evans, 1950). Além destes trabalhos é importante citar dois fatos ocorridos neste período : o uso da Transformada de Laplace por volta de 1950 (Takahashi et al., 1972) e a realização da Primeira Conferência Internacional em Controle Automático no ano de 1951, na cidade de Cranfield, Inglaterra, presidida por Arnold Tustin, na época professor da Universidade de Birmingham (Bissel, 1992).

Todos os resultados, do século passado até 1956, constituem o que se convencionou chamar de "controle clássico" (Paraskevopoulos, 1996). A partir de 1957 inicia-se a fase de ciência para a Teoria de Controle, com os trabalhos apresentando uma forte base teórica combinada com aplicações práticas.

O período de 1957 até meados da década de oitenta é caracterizado como "controle moderno" (Paraskevopoulos, 1996). Aqui a facilidade no uso de computadores tem uma grande influência no desenvolvimento dos trabalhos em controle. É nesta época que se desenvolve o controle digital direto, a abordagem de controle no espaço de estados, o controle ótimo, o controle multivariável, o controle adaptativo, o controle robusto e a massificação da aplicação da teoria de estabilidade de Lyapunov.

Alguns dos resultados mais significantes deste período foram desenvolvidos por Åström (Åström, 1970; Åström e Wittenmark, 1997 (2a. Edição); Åström e Wittenmark, 1995 (2a. Edição)), Athans (Athans e Falb, 1966), Bellman (Bellman, 1957), Brockett (Brockett, 1965), Doyle (Doyle et al., 1989), Jury (Jury, 1964), Kalman (Kalman et al., 1969; Kalman, 1960; Kalman e Bucy, 1961), Luenberger

(Luenberger, 1971), Rosenbrock (Rosenbrock, 1970), Wonham (Wonham, 1979), Zames (Zames, 1981), dentre outros<sup>1</sup>.

Na década de setenta aparecem as técnicas de controle robusto e observa-se também avanços nos estudos do controle adaptativo, onde o controlador se ajusta às variações ocorridas no processo. Neste momento pode-se dizer que o segundo nível de inteligência do controlador é obtido.

No final da década de oitenta, inicia-se o desenvolvimento do que hoje se denomina "controle inteligente". Os controladores inteligentes são implementados através das técnicas de inteligência artificial, tais como : sistemas especialistas, lógica fuzzy, redes neurais e algoritmos evolutivos. Com base em cada uma destas técnicas o controlador pode apresentar características como: tratar variáveis linguísticas, aprender por exemplos, trabalhar de maneira similar à um operador humano especialista, etc. Aqui, pode-se dizer que o controlador chegou ao seu terceiro nível de inteligência<sup>2</sup>.

Os resultados mais significativos na área de controle inteligente são os trabalhos de Narendra (Narendra e Pharthasarathy, 1990; Levin e Narendra, 1996) que formalizaram o uso de redes neurais com a Teoria de Controle, Åström (Åström et al., 1986; Åström, 1989) que conceituaram o uso de sistemas especialistas no controle de processos. Takagi (Takagi e Sugeno, 1985) que desenvolveu uma técnica de implementar a lógica fuzzy. Kawato (Kawato et al., 1987) que inspirado no controle motor humano desenvolveu uma arquitetura de controle neural, Jordan (Jordan e Jacobs, 1990) que foi um dos primeiros trabalhos em controle utilizando redes neurais com realimentação. Mandani (Mandani, 1974) que também desenvolveu uma técnica de implementação da lógica fuzzy, Werbos

---

<sup>1</sup> Para obter maiores detalhes sobre o desenvolvimento da Teoria de Controle é aconselhável a leitura dos trabalhos de Otto (Otto, 1971), Fuller (Fuller, 1976) e os periódicos do IEEE (IEEE-TAC, 1971; IEEE-TAC, 1981) e da Automatica (IFAC-Automatica, 1981).

<sup>2</sup> Diferente deste texto, Landau (Landau, 1993) denomina de terceiro nível de inteligência do controlador aos controladores adaptativos que têm parâmetros variantes no tempo, tais como : fator de esquecimento, reinicialização da matriz de covariância, etc.

(Werbos, 1989) que foi um dos primeiros trabalhos utilizando o algoritmo backpropagation em identificação e controle, Kristinsson (Kristinsson e Dumont, 1992) que foi um dos primeiros trabalhos utilizando algoritmo genético em controle de processos, Passino (Passino e Lunardhi, 1993) que estudou o problema de estabilidade para sistemas de controle utilizando sistemas especialistas, Lee (Lee, 1990) que agrupou todos os conceitos da lógica fuzzy na Teoria de Controle. Além deste trabalhos, alguns livros condensam os resultados de vários pesquisadores relevantes deste campo do conhecimento, tais como : White (White e Sofge, 1992), Levine (Levine, 1996), Gupta (Gupta e Sinha, 1996) e Irwin (Irwin et al., 1995).

## 1.2 Controle Inteligente

A partir dos trabalhos da Ciência da Computação no campo da Inteligência Artificial, palavras como controle inteligente, sistemas inteligentes e atuadores inteligentes têm inundado as publicações sobre controle de processos dinâmicos. No contexto de controle a palavra "inteligente" ainda não tem um significado claro. De dicionários tira-se que "inteligência significa a habilidade de compreender, raciocinar e aprender", ou "inteligência é definida como a capacidade de adquirir e aplicar o conhecimento". Levando-se em conta o significado de inteligência conclui-se que dificilmente um controlador projetado pelas metodologias até hoje conhecidas será inteligente. Uma das poucas habilidades inteligente hoje incluída nos controladores é a do aprendizado.

Pela definição de inteligência, um controlador básico com realimentação do sinal de saída do processo pode ser denominado de inteligente, já que adquire conhecimento através da leitura dos sinais de saída do processo e de referência, aplicando este conhecimento na geração do sinal de controle.

Entretanto, para os estudiosos da cognição, que inclui psicólogos, biólogos, educadores, matemáticos e filósofos, este significado e definição de inteligência é demasiado simplório. Neste trabalho, escapando da discussão sobre inteligência, o controle inteligente será tratado como uma nova metodologia da Teoria de Controle.

---

Como nova metodologia, o controle inteligente não é melhor que as outras metodologias já existentes. Ele é sim adequado para tratar certos problemas de controle, como aqueles em que não existe um modelo explícito do processo e/ou aqueles que envolvem processos não-lineares.

Nos sistemas de controle inteligente, a metodologia de construção do controlador é heurística e baseada nas técnicas de Inteligência Artificial. Estes controladores inteligentes são, normalmente, implementados para emular certas funções cognitivas humanas, com o objetivo de controlar processos dinâmicos complexos. Entretanto na implementação final nenhuma mágica é conseguida. O controlador inteligente resultante é apenas um sistema não-linear, heurísticamente construído, as vezes adaptativo, o qual é apreciado para análise, segundo a Teoria de Controle. Por exemplo, vários tipos de controladores inteligentes com aprendizado, são tipos de sistemas não-lineares adaptativos (Antsaklis, 1994).

Todas as características, apresentadas pelo *controlador inteligente*, são utilizadas, predominantemente, no controle de sistemas onde aparecem não-linearidades, dinâmica não-modelada e parâmetros variantes no tempo ou então em situações onde não existe a possibilidade de se encontrar um modelo matemático convincente para o processo (por exemplo o controle motor necessário para que o ser humano ande sobre as duas pernas (Barreto e Proychev, 1994)). O controle convencional, ou não-inteligente, também apresenta controladores que enfrentam estes tipos de situações, como por exemplo controladores PID (Proporcional, Integral e Derivativo) industriais. A diferença entra as duas abordagens para controle de processos, inteligente e convencional, está no enfoque do projeto: "O enfoque do controle inteligente está no projeto de controladores que emulem ou realizem certas funções dos seres humanos, animais ou sistemas biológicos para resolver problemas de controle" (Passino, 1996). Já o enfoque do controle "dito" não-inteligente está no projeto de controladores baseados no modelo matemático do processo. Nos próximos parágrafos faz-se uma breve descrição das quatro técnicas de Inteligência Artificial utilizadas na implementação de controladores inteligentes.

---

Dentre as várias técnicas usadas na construção do controlador inteligente, a utilização de algoritmos evolutivos é inspirada na teoria da evolução de Darwin e genética de Mendel (que consideram, principalmente, os mecanismos de seleção natural). A idéia é gerar controladores a partir das melhores características de outros controladores que apresentam desempenho modesto. Entretanto, estes controladores necessitam ser projetados, inicialmente, através de alguma outra abordagem da teoria de controle.

Os controladores projetados através da lógica fuzzy, denominados de *controladores fuzzy*, necessitam de um amplo conhecimento sobre o processo a ser controlado. Esta necessidade deve-se ao fato de que os mesmos são implementados para imitar o comportamento de um controlador "não-inteligente", sintonizado de alguma maneira (através de um operador humano, através de métodos de otimização, através de tentativa e erro, por exemplos, etc) e que executa esta tarefa com excelência, ou seja, ele deve se comportar como um especialista<sup>3</sup>. A implementação do controlador *fuzzy* é conseguida através do uso de variáveis linguísticas do tipo alto, baixo, médio, quente, frio, etc. Vale a pena ressaltar que a lógica fuzzy apresenta uma base matemática consistente no seu desenvolvimento (espaço de estados, estabilidade por Lyapunov, controladores com estrutura variável) embora este desenvolvimento ainda não tenha sido usado com propriedade na construção dos controladores fuzzy.

Os controladores construídos com base nos Sistemas Especialistas Simbólicos, mais precisamente Baseados em Regras, são controladores que, da mesma forma que os controladores fuzzy, procuram emular o comportamento de um especialista. A sua implementação é realizada, normalmente, através de regras do tipo IF-THEN-ELSE retiradas do conhecimento do especialista. Diferentemente da lógica fuzzy, neste caso, não existe um desenvolvimento matemático próprio por trás

---

<sup>3</sup>Controladores Fuzzy podem ser considerados como um sistema especialista. Esta consideração é baseada na definição geral de Sistema Especialista encontrada em De Azevedo (De Azevedo, 1993).



das relações definidas nas regras. Um exemplo clássico é a utilização de sistemas especialistas para sintonizar um controlador PID (Hang et al., 1996).

Diferentemente das três abordagens de controle inteligente citadas anteriormente, o uso de RNA é caracterizada pelo não conhecimento sobre a planta a ser controlada e pelo não conhecimento de como gerar o controle para esta planta. A tarefa de controlar o processo é aprendida durante uma fase de treinamento, onde o aprendizado normalmente é obtido através da minimização de uma função custo pré-definida. No entanto, faz-se necessário o conhecimento de exemplos, através de pares entrada/saída desejada, que são utilizados para o treinamento da rede, no caso de treinamento *off-line*.

Um controlador baseado em RNA, denominado de *controlador neural*, tem a vantagem de : trabalhar indistintamente com processos SISO (Single-Input Single-Output) e MIMO (Multiple-Input Multiple-Output), operar naturalmente com plantas não-lineares, não necessitar do conhecimento sobre a planta a ser controlada e adaptar-se adequadamente à novas situações não apresentadas durante o aprendizado através da sua capacidade de generalização. Entretanto, pelo fato do aprendizado ser, normalmente, um procedimento de otimização, o mesmo tem a desvantagem de apresentar um desempenho que depende da sintonia inicial dos seus parâmetros de projeto.

As quatro técnicas da Inteligência Artificial, discutidas, podem ser combinadas em sistemas híbridos para a construção de controladores inteligentes. O objetivo é tentar minimizar as deficiências de uma técnica, com a ajuda da eficiência das outras técnicas. Como exemplo pode-se citar : sistemas especialistas e lógica fuzzy (Passino, 1996); algoritmo genético e sistemas especialistas (Warwick, 1996); rede neural, sistemas especialistas e lógica fuzzy (Handelman e Lane, 1996); redes neurais e sistemas especialistas (Pomerleau et al., 1991); redes neurais e algoritmo genético (Harp e Samad, 1991); redes neurais e lógica fuzzy (Werbos, 1996; Nauck et al., 1997); dentre outros.

### 1.3 Redes Neurais Aplicadas em Identificação e Controle de Processos

As RNA's são utilizadas em identificação e controle de processos não-lineares devido a sua facilidade de realizar mapeamentos não-lineares e de ter a capacidade de aprender. Para que uma RNA possa ser utilizada em identificação e/ou controle é necessário que a mesma tenha a característica de representar a dinâmica do sistema no seu processamento da informação.

No Anexo, deste trabalho, são encontradas algumas definições que permitem incorporar formalmente a informação de dinâmica no modelo neural. Basicamente existem duas maneiras de representar a informação de dinâmica na rede neural : através de uma representação externa e de uma representação interna. Na representação externa usa-se na entrada da rede um conjunto de sinais atrasados no tempo, o que indica uma representação entrada/saída para o modelo do processo. Já na representação interna a informação da dinâmica é realizada através de laços de realimentação, internos ou externos aos neurônios, o que possibilita uma representação no espaço de estados para o modelo do processo.

Neste trabalho é desenvolvido um novo modelo de rede neural, com dinâmica interna, que apresenta uma parte dinâmica linear e uma parte estática não-linear. Este novo modelo neural, devido à sua estrutura particular, é adequado para tratar processos orientados à blocos que possam ser divididos em uma parte linear com dinâmica e uma parte não-linear estática, que são os modelos de Wiener (Greblicki, 1992) e Hammerstein (Stoica e Söderström, 1982). Embora tenha uma estrutura interna particular, este novo modelo neural apresenta, como os outros modelos neurais existentes, a característica de um aproximador não-linear universal. Esta capacidade é mostrada através da utilização do mesmo em problemas de identificação e controle de sistemas dinâmicos não-lineares, orientados à blocos ou não.

No procedimento de identificação não-linear são utilizadas as técnicas de validação do modelo proposto por Billings e Voon (Billings e Voon, 1986; Billings et al., 1992), que foram desenvolvidas para validar modelos não-lineares, neurais ou não. Em aplicações onde são utilizadas as RNA's, estas técnicas têm sido usadas para validar modelos neurais não-lineares com dinâmica externa. Neste trabalho as mesmas são utilizadas para validar modelos neurais não-lineares com dinâmica interna. Na tarefa de controle, o novo modelo neural é utilizado como um controlador *feedforward* em cascata com um processo não-linear. O uso do controlador neural *feedforward* foi originalmente proposto por Kawato e co-autores (Kawato et al., 1987), sendo que aqui é utilizada a proposta de Kawato com algumas modificações propostas por De Oliveira (De Oliveira et al., 1991) e Nascimento Jr. (Nascimento Jr., 1994).

#### 1.4 Estrutura da Tese

Este trabalho é constituído de seis capítulos. O capítulo 2 mostra como representar a dinâmica em RNA's, seguindo o formalismo apresentado no Anexo. Neste capítulo mostra-se as duas formas básicas de representação da dinâmica em modelos neurais : interna e externa. A contribuição original deste capítulo é o desenvolvimento de um novo modelo neural com dinâmica interna que apresenta partes dinâmicas lineares e partes não-lineares estáticas. Outra contribuição é o desenvolvimento de um algoritmo de aprendizado do tipo *backpropagation* para esta nova rede neural desenvolvida.

O capítulo 3 introduz o leitor nos procedimentos de identificação dinâmica não-linear utilizando RNA's. Este capítulo mostra as várias estruturas não-lineares usadas na identificação neural, com as mesmas sendo divididas em duas grandes classes : modelos entrada-saída que utilizam RNA's com dinâmica externa e modelos no espaço de estados que utilizam RNA's com dinâmica interna. Além disto, o capítulo descreve o método de validação utilizado para medir o desempenho da identificação neural. As maiores contribuições deste capítulo são duas : utilização de

---

testes de validação de modelo baseados em medidas de correlação para RNA's com dinâmica interna, e utilização da informação do erro de predição<sup>4</sup> na entrada do modelo neural com dinâmica interna no procedimento de identificação de processos dinâmicos não-lineares com ruído de medida não-linear.

O capítulo 4 mostra a aplicação do novo modelo neural desenvolvido, na tarefa de identificar processos dinâmicos que contém blocos dinâmicos lineares e blocos não-lineares estáticos. A contribuição original deste capítulo é apresentar um procedimento unificado de identificação, utilizando RNA's com dinâmica interna, para processos dinâmicos não-lineares orientados à blocos, dos tipos acima mencionados.

O capítulo 5 mostra a utilização do novo modelo neural com dinâmica interna na tarefa de controlar processos dinâmicos não-lineares. A arquitetura de controle utilizada é a do controlador neural *feedforward*, originalmente chamada de *feedback-error-learning*. A contribuição deste capítulo é a utilização de uma RNA com dinâmica interna como controlador *feedforward* na tarefa de controlar processos dinâmicos não-lineares contínuos ou discretos, orientados à blocos ou não.

Finalmente, o capítulo 6 apresenta as conclusões gerais sobre este trabalho e sugestões para futuros trabalhos.

---

<sup>4</sup> Neste trabalho chama-se, sem distinção, o sinal de diferença entre a saída do processo e a saída do modelo no procedimento de identificação de "erro de predição" ou de "resíduo".

## 2 - Representação da Dinâmica em Redes Neurais Artificiais

### 2.1 Introdução

A representação da dinâmica nas RNA's utilizadas em Sistemas de Controle é uma necessidade devido ao fato de se estar trabalhando com sistemas dinâmicos, o que impõe que o modelo gerado pelo procedimento de identificação ou o controlador neural implementado tenham informações de dinâmica incorporado ao seu processamento. Inicialmente os trabalhos sobre RNA's não se preocupavam com o problema da incorporação da dinâmica, já que as mesmas não foram concebidas originalmente para aplicações em sistemas de controle, embora alguns modelos neurais já incluíssem caminhos de realimentação na sua implementação, como por exemplo a rede de Hopfield (Hopfield, 1984). Ao visualizar a sua utilização para problemas de controle, Jordan (Jordan, 1986) propôs um modelo que incluía a informação de dinâmica, originalmente com o objetivo de utilizá-lo no controle de processos dinâmicos, além de processamento da voz. Mas foi com o trabalho de Narendra (Narendra e Parthasarathy, 1990) que o problema de como incluir a dinâmica no modelo neural ficou explicitado. A forma como esta questão foi abordada neste trabalho, não mostrava claramente as diferenças entre as duas formas básicas de incorporação da dinâmica no modelo neural, que são as representações externa e interna<sup>1</sup>. Este desenvolvimento foi devidamente abordado no trabalho de De Azevedo (De Azevedo, 1993), onde são apresentadas definições formais de neurônio artificial e RNA, com as duas formas de incorporação da informação de dinâmica na RNA. Estas definições são apresentadas no Anexo deste

---

<sup>1</sup> O trabalho de Narendra e Parthasarathy (Narendra e Parthasarathy, 1990) foi um marco no estudo das RNA's em sistemas de controle. O motivo foi que, pela primeira vez na literatura, a utilização da RNA em identificação e controle não-linear foi abordada através do formalismo matemático da Teoria de Controle.

trabalho. Juntamente com as definições de De Azevedo, aparecem alguns teoremas de Barreto (Barreto, 1996) que são aperfeiçoamentos do trabalho original de De Azevedo. A partir da definição formal de neurônio artificial e RNA, pode-se verificar claramente a opção de incluir dinâmica na forma interna ou na forma externa na RNA, e observar que estas duas situações estão intimamente relacionadas com a representação do modelo utilizado, ou seja, se o modelo é uma representação entrada/saída ou uma representação no espaço de estados do processo dinâmico estudado.

## 2.2 Modelo Neural com Dinâmica

As formas de inclusão da informação da dinâmica na RNA são baseadas no modelo de neurônio artificial mostrado na Figura 2. 1. Neste modelo as entradas  $w_i, u_i$  são combinadas usando uma função  $\phi$  para produzir um estado de ativação do neurônio que através da função  $\lambda$  irá gerar o sinal de saída do neurônio. A função de ativação  $\phi$  pode incluir ou não dinâmica no seu processamento. A utilização de uma função de ativação diferente da função de saída, ao contrário do usual na literatura sobre RNA's que normalmente as utiliza como se fossem uma só, fornece ao neurônio uma maior flexibilidade de uso. Mas, principalmente, permite uma definição formal de neurônio dinâmico e de neurônio estático, como visto no Anexo.

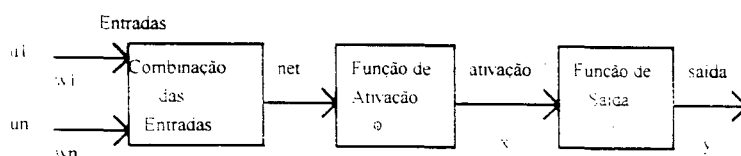


Figura 2. 1 - Elementos básicos de um neurônio artificial

As características deste modelo permitem definir o modelo geral de um neurônio como sendo um sistema dinâmico. Permitindo também, junto com os conceitos de sistemas (Kalman et al., 1969), formalizar vários tipos de neurônios (Anexo).

Na notação usualmente adotada na literatura, o valor de  $net$  é;

$$net_i(t) = \sum_{j=1}^n w_{ij} u_j(t) \quad (2.1)$$

onde  $w_{ij}$  é um número real que representa a conexão da entrada do  $i$ -ésimo neurônio com a saída do  $j$ -ésimo neurônio. Após a determinação de  $net_i(t)$ , o valor do estado de ativação do neurônio é atualizado através da função de ativação  $\phi$  e, finalmente, o valor de saída do neurônio é produzido através da função de saída  $\lambda$ . Analiticamente, a ativação e a saída dos neurônios são dadas pelas seguintes equações;

$$x(t+h) = \phi(x(t), net(t), t, h) \quad (2.2)$$

$$y(t) = \lambda(x(t), net(t), t) \quad (2.3)$$

Destas relações formais, pode-se obter diferentes modelos a partir de modificações na função de ativação  $\phi$  e na função de saída  $\lambda$ . Examinando a equação (2.2) verifica-se que os estados futuros do neurônio são afetados pelo estado atual do neurônio, e pelo valor do sinal de entrada  $net$ . Este tipo de neurônio, que possui memória, é conhecido como *neurônio dinâmico*. Por outro lado, considerando-se a função  $\phi$  como constante, tem-se neurônios que não possuem memória, ou seja, o estado atual é igual ao estado anterior. Neste caso, o neurônio é conhecido como *neurônio estático*. A equação (2.3) determina um mapeamento estático do estado atual do neurônio, incluindo a sua entrada, para o sinal de saída do mesmo.

Embora os neurônios sejam elementos computacionais extremamente interessantes, isoladamente não são poderosos para processar ou representar o conhecimento. Entretanto, se um conjunto de neurônios é interconectado em uma rede de neurônios, denominada de RNA, emerge uma série de propriedades

computacionais não encontradas em um simples neurônio. Uma RNA é um sistema composto por vários neurônios formando um sistema complexo (Anexo), onde os neurônios são ligados por conexões e normalmente agrupados em camadas. Alguns neurônios recebem excitações do exterior e são denominados *neurônios de entrada*, formando a *camada de entrada* da RNA. Outros interagem, de alguma forma, com o mundo exterior e são denominados *neurônios de saída*, formando a *camada de saída* da RNA. Os neurônios que não são entrada nem saída, são conhecidos como *neurônios internos* e formam a *camada intermediária* da RNA.

Um exemplo de RNA é mostrado na Figura 2. 2 . Este tipo de RNA, multicamadas e *feedforward*, é o mais utilizado em sistemas de controle (Irwin et al., 1995; Gupta e Rao, 1994). Este modelo é composto de várias camadas de neurônios, por isto é chamado multicamadas; os neurônios de uma camada se conectam apenas com os neurônios da camada imediatamente posterior, determinando um fluxo de informação na forma *feedforward*. Entretanto existem vários modelos de RNA que permitem uma ampla conexão entre neurônios de diferentes camadas e também dentro da mesma camada (De Oliveira et al., 1998).

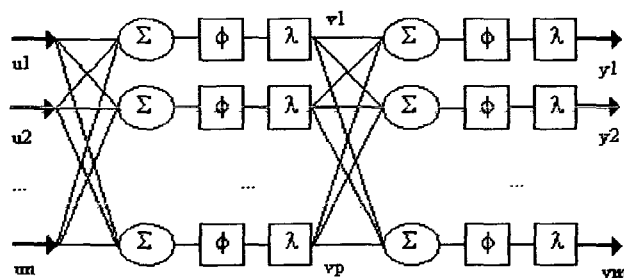


Figura 2. 2 - Representação de uma RNA *feedforward* multicamadas, com duas camadas de neurônios

Os modelos neurais podem ser contínuos ou discretos. Neste trabalho utiliza-se apenas os modelos discretos de RNA's, devido a facilidade de implementá-los via software. Além disto, são considerados apenas modelos multicamadas. As seções a seguir mostram as duas formas de incluir a informação de dinâmica nos modelos



neurais multicamadas discretos.

### 2.2.1. Modelo Neural com Representação da Dinâmica na Forma Externa

Nos modelos de RNA's com dinâmica externa existem apenas neurônios estáticos. Ou seja, a informação de dinâmica é fornecida externamente. Esta informação é codificada por um conjunto de sinais atrasados no tempo. A Figura 2. 3 mostra uma RNA *feedforward* multi-camadas discreta com dinâmica externa.

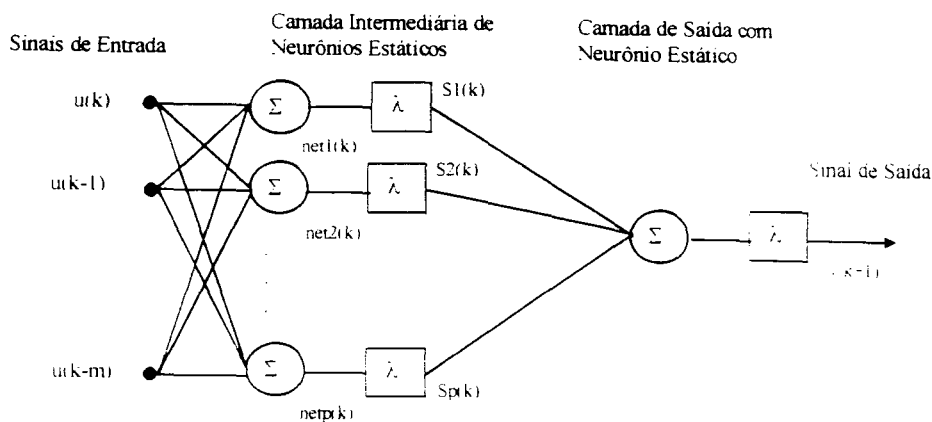


Figura 2. 3 - RNA *feedforward* multi-camadas discreta com informação de dinâmica representada através dos sinais de entrada atrasados no tempo

Da Figura 2. 3, tem-se que:

$$y(k+1) = f(u(k), u(k-1), \dots, u(k-m), W) \quad (2.4)$$

onde  $W$  indica a matriz que contém os pesos que conectam os sinais de entrada aos neurônios intermediários e os pesos que conectam os neurônios intermediários ao neurônio de saída. A equação (2.4) indica um modelo NAR (Non-linear AutoRegressive model structure), mas com a manipulação adequada dos sinais de entrada da rede, a rede pode representar modelos NARMA (Non-linear AutoRegressive Moving Average model structure), NARMAX (Non-linear AutoRegressive Moving Average model structure with exogenous input), etc

(Sjoberg, 1995). A equação (2.4) é claramente uma representação entrada/saída.

Devido as similaridades entre estas técnicas neurais com o controle adaptativo, este modelo que usa um vetor de regressão  $\varphi(k) = [u(k), u(k-1), \dots, u(k-m)]^T$  na sua entrada, é o mais utilizado em sistemas de controle. Devido a esta proximidade, foram feitos inúmeros trabalhos que utilizam as técnicas adaptativas para sistemas lineares (Ljung, 1987; Åström e Wittenmark, 1995; Goodwin e Payne, 1977) junto com este modelo neural (Chen et al., 1990b; Billings et al., 1992; Ng, 1994; Saint-Donat et al., 1991). Entretanto, como será visto a seguir, a outra forma de representar a dinâmica na rede neural, vislumbra a possibilidade de utilização do conceito de estado, o que torna a rede mais poderosa para tratar sistemas dinâmicos.

### 2.2.2. Modelo Neural com Representação da Dinâmica na Forma Interna

Neste caso, a RNA *feedforward* contém neurônios dinâmicos. A informação de dinâmica será fornecida internamente. Esta informação é determinada através da função  $\phi$  do neurônio. Para o caso de uma RNA *feedforward* discreta a Figura 2. 4 mostra uma possível configuração (para facilitar a apresentação será considerado apenas dinâmica linear nos neurônios).

Da Figura 2. 4 tem-se que;

$$x(k+1) = Ax(k) + W^1 u(k) \quad (2.5)$$

$$S(k) = \lambda(x(k)) \quad (2.6)$$

$$y(k) = \lambda(W^s S(k)) \quad (2.7)$$

ou,

$$y(k) = g(x(k), u(k), A, W) \tag{2.8}$$

onde, neste caso, a matriz  $A$  é uma matriz diagonal,  $A = \text{diag}[a_{11}, a_{22}, \dots, a_{nn}]^T$ , a variável  $W$  inclui os pesos  $W^I$  que conectam o sinal de entrada aos neurônios intermediários e os pesos  $W^S$  que conectam os neurônios intermediários ao neurônio de saída. O vetor  $x(.)$  é o vetor de ativação dos neurônios dinâmicos da rede. Observando apenas as equação (2.5) e equação (2.8) nota-se claramente que esta é uma representação no espaço de estados.

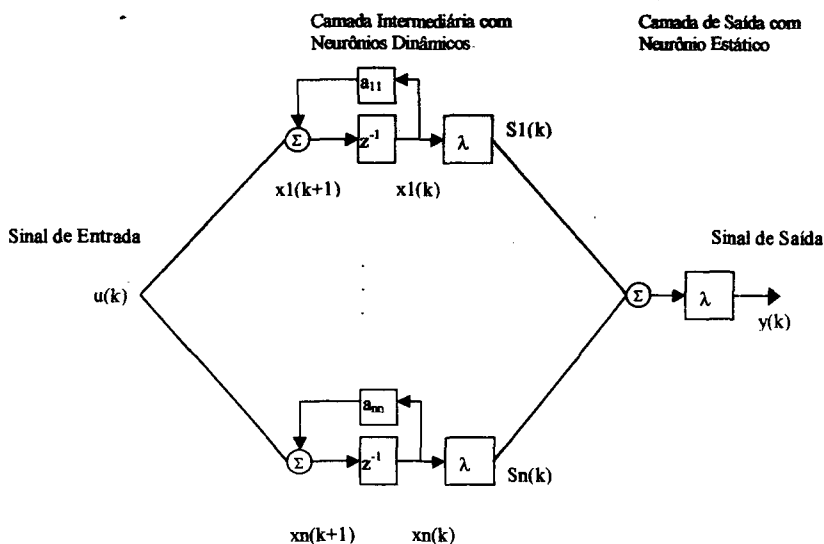


Figura 2. 4 - RNA *feedforward* multi-camadas discreta com informação de dinâmica representada através dos neurônios dinâmicos da camada intermediária, sendo esta dinâmica linear de primeira ordem

A linearidade da equação (2.5) pode ser facilmente substituída por uma não-linearidade, como por exemplo;

$$x(k + 1) = h(x(k), u(k), A, W^I) \tag{2.9}$$

A matriz  $A$  pode assumir as mais variadas formas. Normalmente a matriz  $A$  é cheia indicando que todos os neurônios dinâmicos se conectam entre si, ou contém a diagonal principal e outras duas diagonais acima e abaixo da principal, indicando que cada neurônio dinâmico se conecta apenas com os dois vizinhos mais próximos,

ou contem apenas a diagonal principal indicando que não existe conexão entre os neurônios dinâmicos, exceto com eles próprios.

A equação (2.5) ou a equação (2.9) permite o uso dos estados dos neurônios como uma estimativa dos estados do processo dinâmico estudado. Neste trabalho, esta possibilidade não é explorada. O motivo é que o algoritmo de aprendizado desenvolvido para esta modelo neural, não é adequado para tratar estimação de estados e sim para tratar a estimação de parâmetros.

### **2.2.3. Tipos de Modelos Neurais com Representação da Dinâmica na Forma Interna**

Redes neurais com dinâmica interna foram desenvolvidas desde o começo dos anos oitenta, após o início do surgimento do ramo conexionista da Inteligência Artificial<sup>2</sup>. Embora estes modelos, com dinâmica interna, não fossem, à princípio, voltados para aplicação em problemas de controle dinâmico, os mesmos apresentavam características interessantes à comunidade de controle. Nesta seção apresenta-se alguns modelos neurais pioneiros que incluem a informação dinâmica na forma interna, adequados ou não à controlar e/ou identificar processos dinâmicos. Além destes modelos pioneiros, são apresentadas outras redes neurais, mais recentes, que já foram originalmente propostas à aplicações em sistemas de controle.

#### **2.2.3.1. Rede de Hopfield**

A rede de Hopfield, assim chamada por ter sido desenvolvida pelo Professor de Biologia e Química da CALTECH (California Institute of Technology) J.J. Hopfield,

---

<sup>2</sup>Os primeiros trabalhos sobre redes neurais são da década de 40 (Hebb, 1949), 50 (Rosenblatt, 1958) e 60 (Widrow e Hoff, 1960), sendo que no final dos anos 60 este ramo de pesquisa enfraqueceu por causa do trabalho de Minsky e Papert (Minsky e Papert, 1969) que mostrava sérias limitações para estes modelos computacionais conexionistas e, equivocadamente, não previa avanços na área. Durante os anos 70, surgiram, de maneira isolada, alguns trabalhos nesta área (Fu, 1970; Werbos, 1974; Anderson, 1977).

foi a responsável pelo despertar do interesse dos pesquisadores pela análise das redes neurais artificiais (Hopfield, 1982)<sup>3</sup>, que estava desaparecido da comunidade científica desde o célebre artigo de Minsky e Papert (Minsky e Papert, 1969).

Embora, originalmente, tenha sido apresentada como uma rede discreta, com a saída dos neurônios tendo apenas dois estados,  $\pm 1$ , este modelo tem a sua versão contínua (Hopfield, 1984). O comportamento da rede neural de Hopfield contínua é governado pelo conjunto de equações diferenciais de primeira ordem, apresentado abaixo na forma matricial;

$$\tau \frac{dX}{dt} = -\alpha X + W\lambda(X) + \theta \quad (2.10)$$

onde;

$X = [x_1 \ x_2 \ \dots \ x_n]^T$  é o vetor de estado dos neurônios;

$\tau = \text{diag}(\tau_1 \ \tau_2 \ \dots \ \tau_n)$  é uma matriz diagonal, representando as constantes de tempo responsáveis pela dinâmica dos neurônios;

$\alpha = \text{diag}(\alpha_1 \ \alpha_2 \ \dots \ \alpha_n)$  é uma matriz diagonal, representando os coeficientes de amortecimento responsáveis pela dinâmica dos neurônios;

$W = \begin{bmatrix} w_{11} & w_{12} & \dots & w_{1n} \\ w_{21} & w_{22} & \dots & w_{2n} \\ \dots & \dots & \dots & \dots \\ w_{n1} & w_{n2} & \dots & w_{nn} \end{bmatrix}$  é uma matriz, com  $n^2$  elementos, contendo os pesos

de conexão entre os neurônios da rede;

$\lambda(X) = [\lambda_1(x_1) \ \lambda_2(x_2) \ \dots \ \lambda_n(x_n)]^T$  é a função de saída dos neurônios (normalmente um função tangente hiperbólica) que gera o vetor de saída  $V = \lambda(X)$  dos neurônios;

<sup>3</sup>Outro trabalho que contribuiu para o resurgimento dos estudos em redes neurais artificiais foram as publicações do Parallel Distributed Processing Group (PDP) (Rumelhart et al., 1986a, 1986b, 1986c).

$\theta = [\theta_1 \theta_2 \dots \theta_n]^T$  é o vetor de *bias* da rede neural.

A representação esquemática da equação (2.10) é mostrada na Figura 2.5 . A dinâmica da rede é determinada pelos valores dos  $\tau_i$ 's , dos  $\lambda_i$ 's e dos  $w_{ij}$ 's. Este modelo de rede é adequado para implementação em *hardware*, com a equação (2.10) sendo realizada com o uso de capacitores, resistores, amplificadores operacionais e fontes de corrente (Cichocki e Unbehauen, 1993). O ajuste das conexões da rede de Hopfield pode ser feito através de algoritmo de aprendizado do tipo *backpropagation* (Raol, 1995).

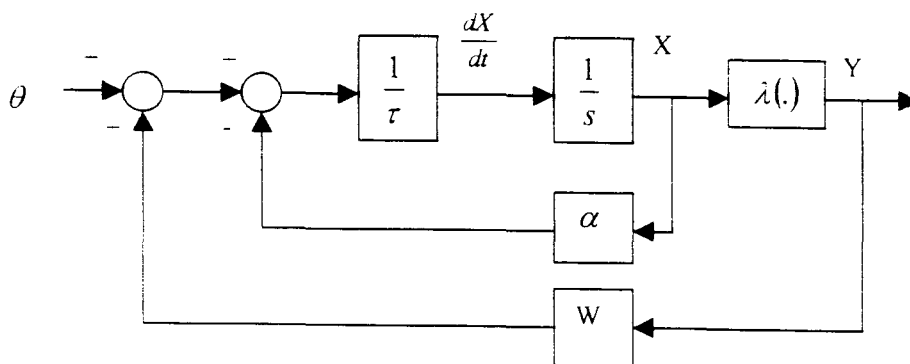


Figura 2. 5 - Diagrama esquemático da rede de Hopfield onde  $V = \lambda(X)$  é o sinal de saída da rede

### 2.2.3.2. Rede de Elman

A rede de Elman (Elman, 1990) é um tipo de rede recorrente discreta de três camadas em que os sinais de saída dos neurônios da camada intermediária são realimentados para a entrada de um conjunto de neurônios, chamados de neurônios de contexto. Esta rede tem apenas um sinal de entrada e um sinal de saída. A figura a seguir mostra a rede de Elman original.

Na Figura 2. 6 as conexões feedforward (\_\_\_) são ajustáveis e as conexões de realimentação (\_\_\_) são fixas de valor unitário, por isto este modelo é chamado de "parcialmente recorrente" (Pham e Liu, 1992). O ajuste das conexões feedforward é feito através de algoritmo de aprendizado do tipo *backpropagation*

(Pham e Liu, 1996).

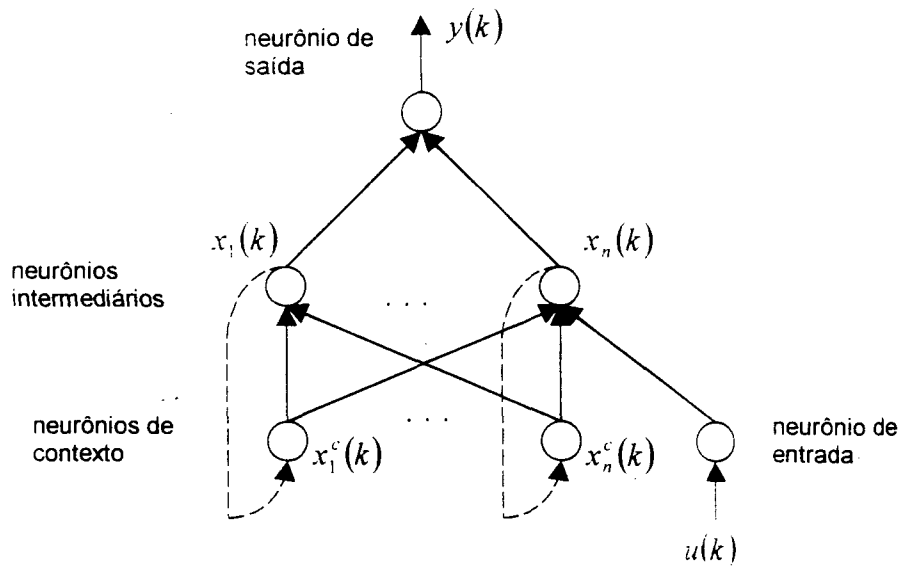


Figura 2. 6 - Rede de Elman original

As equações que regem o comportamento deste modelo são:

$$net_i(k) = \sum_{j=1}^n w_{ij}^x x_j^c(k) + w_i^u u(k) \tag{2. 11}$$

$$x_i(k) = \lambda(net_i(k)) \tag{2. 12}$$

$$x_i^c(k) = x_i(k-1) \tag{2. 13}$$

$$y(k) = \sum_{i=1}^n w_i^s x_i(k) \tag{2. 14}$$

onde;

$net_i(k)$  é o sinal de entrada dos neurônios intermediários:

$x_i(k)$  é o sinal de saída dos neurônios intermediários ou estados da rede:

$x_i^c(k)$  é o sinal de saída dos neurônios de contexto:

$y(k)$  é o sinal de saída da rede;

$u(k)$  é o sinal de entrada da rede;

$\lambda(\cdot)$  é a função de saída dos neurônios intermediários;

$w_{ij}^x$  é o peso da conexão entre os neurônios intermediários e de contexto;

$w_i^l$  é o peso da conexão entre os neurônios intermediários e o sinal de entrada;

$w_i^s$  é o peso da conexão entre o neurônio de saída e os neurônios intermediários.

Originalmente esta rede foi desenvolvida de forma a introduzir informação de tempo no processamento dos modelos conexionistas, de maneira que permitisse a sua utilização no processamento da fala (Elman, 1990). Mais tarde este modelo foi utilizado também para identificação de processos dinâmicos lineares (Pham e Liu, 1992) e não-lineares (Pham e Liu, 1996).

### 2.2.3.3. Rede Neural Recorrente Diagonal

A rede neural recorrente diagonal (Ku e Lee, 1995) foi desenvolvida para aplicações em identificação e controle de processos dinâmicos não-lineares. O objetivo foi projetar uma rede que apresentasse dinâmica interna e ao mesmo tempo poucos parâmetros a serem ajustados. Este modelo é mostrado na figura abaixo, onde,

$$net_i(k) = \sum_{j=1}^m w_{ij}^l u_j(k) + w_i^D x_i(k-1) \quad (2.15)$$

$$x_i(k) = \lambda(net_i(k)), \quad (2.16)$$

$$y(k) = \sum_{j=1}^n w_j^s x_j(k) \quad (2.17)$$

com,



$net_i(k)$  é o sinal de entrada dos neurônios intermediários;

$x_i(k)$  é o sinal de saída dos neurônios intermediários, ou estados da rede;

$w_{ij}^I$  é o peso da conexão entre os sinais de entrada e os neurônios intermediários;

$w_j^D$  é o peso da conexão entre os neurônios intermediários e eles próprios;

$w_j^S$  é o peso da conexão entre os neurônios intermediários e o neurônio de saída;

$\lambda(.)$  é a função de saída dos neurônios intermediários.

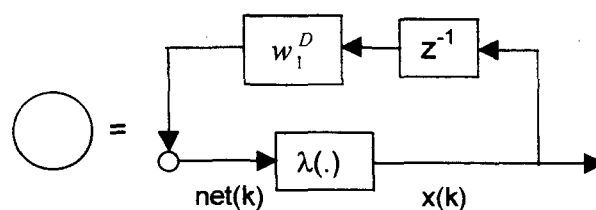
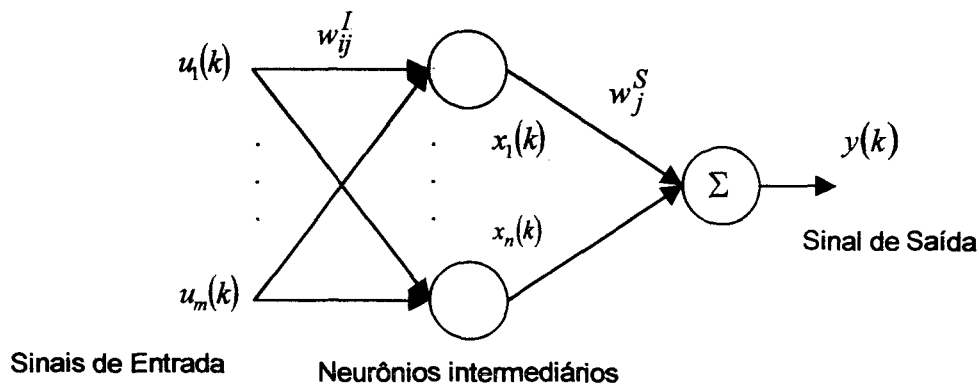


Figura 2. 7 - Rede Neural Recorrente Diagonal

Todos os pesos da rede neural são ajustados através de um algoritmo de aprendizado do tipo *backpropagation* (Ku e Lee, 1995).

Como todo modelo que possui dinâmica interna, esta rede necessita de atenção no que se refere a estabilidade da mesma. Para a rede neural recorrente

diagonal ser estável é necessário que  $0 < |w_j^D| < 1$ . Esta imposição, derivada do fato que a rede é diagonal também é uma das condições de convergência do algoritmo de aprendizado. As condições necessárias e suficientes que garantem a estabilidade deste modelo neural são encontradas a partir da Teoria de Lyapunov (Ku e Lee, 1995).

### 2.3 Modelo Neural com Dinâmica Linear e com Não-Linearidade Estática

O modelo de rede apresentado nesta seção é uma modificação da Rede Neural Recorrente Diagonal que agora conterá neurônios dinâmicos que apresentam dinâmica linear e uma não-linearidade estática. Este novo modelo neural é adequado para tratar sistemas dinâmicos que contenham uma parte linear dinâmica e uma parte não-linear estática. Esta classe de processos dinâmicos é utilizada em algumas importantes aplicações, tais como : controle de pH (Pajunen, 1994), fluxo de fluido (Singh et al., 1980), identificação de sistemas biológicos (Hunter e Korenberg, 1986) e identificação de sistemas lineares com sensores não-lineares (Doebelin, 1983). Além desta implicação prática, a motivação de se utilizar dinâmica interna linear no modelo neural é a de facilitar o estudo de estabilidade do mesmo. Esta rede será apresentada no contexto de modelagem e identificação de sistemas dinâmicos. Embora tenha esta estrutura particular, este novo modelo neural não perde a sua capacidade de aproximador não-linear geral, que serve para modelar um processo não-linear qualquer.

Dado um sistema dinâmico discreto não-linear representado no espaço de estados como determina a equação (2.18) e a equação (2.19),

$$x(k+1) = Ax(k) + Bu(k) \quad (2.18)$$

$$y(k) = \Phi(x(k)) \quad (2.19)$$

ou,

$$y(k) = \Phi(Ax(k-1) + Bu(k-1)) \tag{2.20}$$

onde  $k$  é o instante de amostragem,  $A$  é a matriz  $n \times n$  de dinâmica linear,  $B$  é o vetor entrada  $n \times 1$ ,  $u(k)$  é o sinal de entrada no instante  $k$ ,  $x(k)$  é o vetor de estados  $n \times 1$  no instante de amostragem  $k$ ,  $y(k)$  é o sinal de saída no instante de amostragem  $k$ , e  $\Phi(\cdot)$  é a função não-linear de saída que realiza o mapeamento  $\mathbb{R}^n \rightarrow \mathbb{R}^1$ .

Uma possível representação para o modelo da equação (2.20) através de uma RNA com dinâmica interna, composta de três camadas de neurônios, é mostrada na Figura 2. 8, sendo que a camada de neurônios de entrada serve apenas como um *buffer* para o sinal de entrada.

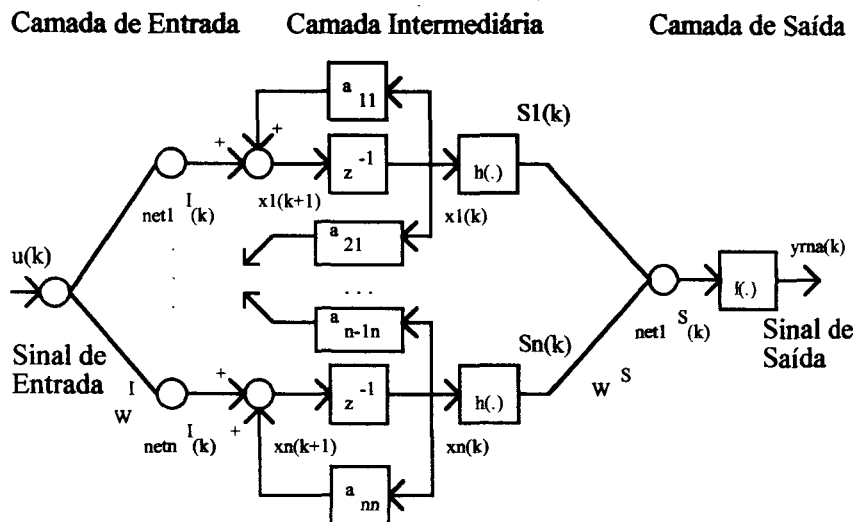


Figura 2. 8 - RNA com dinâmica interna linear, composta por três camadas de neurônios servindo como modelo de um sistema discreto não-linear

A RNA da Figura 2. 8 apresenta neurônios dinâmicos na camada intermediária e neurônio estático na camada de saída. A função ativação dos neurônios intermediários é dada por uma equação a diferenças de primeira ordem. As funções de saída  $h(\cdot)$  e  $k(\cdot)$  são funções estáticas não-lineares, sendo que  $k(\cdot)$  pode ser linear.

Da Figura 2. 8 temos que,

$$S_j(k) = h[x_j(k)] \quad (2.21)$$

$$x_j(k+1) = a_{j1}x_1(k) + \dots + a_{jj}x_j(k) + \dots + a_{jn}x_n(k) + net_j^I(k) \quad (2.22)$$

com

$$net_j^I(k) = w_{j1}^I u(k) \quad (2.23)$$

ou;

$$x_j(k+1) = a_{j1}x_1(k) + \dots + a_{ji}x_i(k) + \dots + a_{jn}x_n(k) + w_{j1}^I u(k) \quad (2.24)$$

Da equação (2.21), com  $x(k)$  retirado da equação (2.24), temos,

$$S_j(k) = h[a_{j1}x_1(k-1) + \dots + a_{ji}x_i(k-1) + \dots + a_{jn}x_n(k-1) + w_{j1}^I u(k-1)] \quad (2.25)$$

ou,

$$S(k) = h[Ax(k-1) + W^I u(k-1)] \quad (2.26)$$

com,

$$A = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \dots & \dots & \dots & \dots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{bmatrix}; \quad W^I = \begin{bmatrix} w_{11}^I \\ w_{21}^I \\ \dots \\ w_{n1}^I \end{bmatrix} \quad (2.27)$$

O sinal de saída  $y_{RNA}(k)$  é igual a,

$$y_{RNA}(k) = l[W^S S(k)] \quad (2.28)$$

da equação (2.26), tem-se que,

$$y_{RNA}(k) = l[W^S h[Ax(k-1) + W^I u(k-1)]] \quad (2.29)$$

ou,

$$y_{RNA}(k) = N[W, x(k-1), u(k-1)] \quad (2.30)$$

que formaliza o modelo dado pela equação (2.20), sendo que o algoritmo de aprendizado ajusta os elementos de  $W$  de tal forma que  $N(.) \approx \Phi(.)$ . Os elementos da matriz  $W$  incluem os elementos das matrizes  $W^I$  e  $W^S$ , além dos elementos  $a_{ij}$  de  $A$ .

Embora a equação (2.30) determine que a atualização dos estados do processo (equação (2.18)) seja linear, o mapeamento do espaço de entrada,  $u(.)$ , para o espaço de saída,  $y(.)$ , realizado pela RNA não restringe o caso em que a equação (2.18) seja não-linear. Neste caso,  $N(.)$  aproxima uma função  $\bar{\Phi}(.)$  e não mais apenas  $\Phi(.)$ . Se a atualização dos estados do processo é não-linear (equação (2.9)), o modelo neural proposto não segue a estrutura interna do processo, ou seja, o espaço de estados da rede será diferente do espaço de estados do processo, embora a RNA consiga preservar a relação entrada-saída determinada pela função  $\bar{\Phi}(.)$ .

Nos modelos de RNA's com dinâmica interna propostos na literatura, tais como Karakasoglu (Karakasoglu et al., 1993), Haykin (Haykin, 1994) e os modelos da seção 2.2.3, não diferenciam a *função de ativação* da *função de saída*. Por isto, estes modelos utilizam na malha de realimentação o sinal de saída do neurônio, dificultando a elaboração do algoritmo de aprendizado e o estudo de estabilidade da rede. O modelo de RNA mostrado na Figura 2. 8 apresenta uma realimentação do estado do neurônio permitindo, a partir das definições de função ativação e função de saída do neurônio, o uso de uma equação linear para representar a dinâmica do neurônio. A utilização da dinâmica linear na função ativação do neurônio facilita o estudo de estabilidade do modelo neural, ou seja, faz-se apenas um cálculo de autovalores da matriz  $A$ , definida na equação (2.25) e na equação (2.27) para determinar se a rede neural com dinâmica interna é estável ou não.

Aqui, é conveniente lembrar que, durante a fase de aprendizado da RNA, o algoritmo de ajuste dos pesos da rede modifica os elementos da matriz  $A$  após cada

iteração. Sendo assim, o cálculo do autovalores de  $A$  se faz necessário em cada iteração. Esta é uma maneira não elegante, que consome algum tempo de processamento numérico, de verificar a estabilidade deste modelo neural já que na etapa de aprendizado os autovalores da matriz  $A$  são variantes no tempo.

### 2.3.1. Algoritmo de Aprendizado

O algoritmo de aprendizado desenvolvido para o modelo de RNA com dinâmica interna, visto na Figura 2. 8, é do tipo *backpropagation* (Rumelhart et al., 1986a). O mesmo ajusta os pesos  $w_{ji}^I$ ,  $w_{ji}^S$  e os parâmetros  $a_{ji}$  da RNA.

O ajuste dos pesos  $w_{ji}^S$  segue as equações padrões do algoritmo *backpropagation* padrão (Rumelhart et al., 1986a). O ajuste dos pesos  $w_{ji}^I$  e dos parâmetros  $a_{ji}$  são mostradas a seguir (não são mostrados todos os passos do procedimento de dedução do *backpropagation*, devido os mesmos já serem de amplo conhecimento na literatura (Haykin, 1994; Zurada, 1992; Krose e van der Smagt, 1993; Dayhoff, 1990; Simpson, 1990)).

Considerando que deseja-se minimizar a função custo do erro quadrático  $E(W)$ ,

$$E(W) = \frac{1}{2} [y(k) - y_{RNA}(k)]^2 \quad (2. 31)$$

onde  $y(k)$  é a saída do processo no instante de amostragem  $k$ , ou saída desejada para a rede,  $y_{RNA}(k)$  é a saída da rede neural no instante de amostragem  $k$  e  $W$  é a matriz de parâmetros a serem ajustados, que inclui  $w_{ji}^I$ ,  $w_{ji}^S$  e  $a_{ji}$ . O efeito de minimizar  $E(W)$  é conseguido através do ajuste da matriz de pesos  $W$  na direção oposta ao gradiente de  $E(W)$ , ou seja,

$$W_{k+1} = W_k - \eta \text{Grad}[E(W)] \quad (2. 32)$$

com  $\eta$  indicando o quanto do gradiente será utilizado na atualização dos pesos  $W$ ,

ou apenas o tamanho do passo na direção do gradiente.

O gradiente  $Grad[E(W)]$  é calculado por,

$$Grad[E(W)] = \frac{\partial E(W)}{\partial w_i} \quad (2.33)$$

Para a camada de saída o algoritmo segue o backpropagation padrão, ou seja;

$$\frac{\partial E(W)}{\partial w_{1i}^S} = \delta_1^S(k) S_i(k) \quad (2.34)$$

com;

$$\delta_1^S(k) = [y(k) - y_m(k)] l'(net_1^S(k)) \quad (2.35)$$

onde  $l'(net)$  indica a derivada de  $l(.)$  em relação a  $net$ .

Na camada intermediária, com os neurônios de dinâmica linear, o cálculo da equação (2.33) torna-se,

$$\frac{\partial E(W)}{\partial w_{j1}^I} = \frac{\partial E(W)}{\partial S_j(k)} \frac{\partial S_j(k)}{\partial w_{j1}^I} \quad (2.36)$$

o valor de  $\frac{\partial E(W)}{\partial S_j(k)}$  é calculado como;

$$\frac{\partial E(W)}{\partial S_j(k)} = \delta_1^S(k) w_{1j}^S \quad (2.37)$$

A derivada  $\frac{\partial S_j(k)}{\partial w_{j1}^I}$  para a camada intermediária é,

$$\frac{\partial S_j(k)}{\partial w_{j1}^I} = \frac{\partial S_j(k)}{\partial x_j(k)} \frac{\partial x_j(k)}{\partial w_{j1}^I} \quad (2.38)$$

ou seja,

$$\frac{\partial S_j(k)}{\partial w_{j1}^I} = h'(x_j(k)) \frac{\partial x_j(k)}{\partial w_{j1}^I} \quad (2.39)$$

onde  $h'(net)$  indica a derivada de  $h(.)$  em relação a  $x$ .

Sendo assim, a partir da equação (2.39) e da equação (2.37), tira-se que a equação (2.36) torna-se;

$$\frac{\partial E(W)}{\partial w_{j1}^I} = \delta_1^s(k) w_{1j}^s h'(x_j(k)) \frac{\partial x_j(k)}{\partial w_{j1}^I} \quad (2.40)$$

ou seja;

$$\frac{\partial E(W)}{\partial w_{j1}^I} = \delta_j^I(k) \frac{\partial x_j(k)}{\partial w_{j1}^I} \quad (2.41)$$

$$\delta_j^I(k) = \delta_1^s(k) w_{1j}^s h'(x_j(k)) \quad (2.42)$$

Da equação (2.24) tem-se que;

$$x_j(k) = a_{j1} x_1(k-1) + a_{j2} x_2(k-1) + \dots + a_{jj} x_j(k-1) + \dots + a_{jn} x_n(k-1) + w_{j1}^I u(k-1) \quad (2.43)$$

ou seja;

$$x_j(k) = a_{j1} x_1(k-1) + \dots + a_{jj} \left\{ a_{j1} x_1(k-2) + \dots + a_{jj} x_j(k-2) + \dots + a_{jn} x_n(k-2) + w_{j1}^I u(k-2) \right\} + \dots + a_{jn} x_n(k-1) + w_{j1}^I u(k-1) \quad (2.44)$$

Sendo assim, o termo  $\frac{\partial x_j(k)}{\partial w_{j1}^I}$  torna-se;



$$\frac{\partial x_j(k)}{\partial w_{j1}'} = a_{jj} \frac{\partial x_j(k-1)}{\partial w_{j1}'} + u(k-1) \quad (2.45)$$

Definindo-se;

$$Q_j(k) = \frac{\partial x_j(k)}{\partial w_{j1}'} ; Q_j(0) = 0 \quad (2.46)$$

a equação (2.45) torna-se;

$$Q_j(k) = a_{jj} Q_j(k-1) + u(k-1) \quad (2.47)$$

De posse da definição dada pela equação (2.46), o  $Grad[E(W)]$  para o elementos do vetor  $W^1$  será;

$$\frac{\partial E(W)}{\partial w_{j1}'} = \delta_j^1(k) Q_j(k) \quad (2.48)$$

com  $Q_j(k)$  determinado pela equação (2.47). A recorrência dada pela equação (2.47) será estável se  $|a_{jj}| < 1$ .

O cálculo de  $\frac{\partial E(W)}{\partial a_{ji}}$  segue o mesmo procedimento anterior, com;

$$\frac{\partial E(W)}{\partial a_{ji}} = \delta_j^1(k) \frac{\partial x_j(k)}{\partial a_{ji}} \quad (2.49)$$

da equação (2.44) tira-se que;

$$\frac{\partial E(W)}{\partial a_{ji}} = \delta_j^1(k) P_j(k) \quad (2.50)$$

onde;

$$P_j(k) = a_{jj} P_j(k-1) + x_i(k-1) \quad (2.51)$$

$$P_j(k) = \frac{\partial x_j(k)}{\partial a_{ji}} ; P_j(0) = 0 \quad (2.52)$$

Assim da mesma forma que a equação (2.47), a recorrência dada pela equação

(2.51) será estável se  $|a_{jj}| < 1$ .

Para uma RNA com dinâmica interna linear, tendo  $v$  sinais de entrada,  $n$  neurônios intermediários e  $r$  sinais de saída, os gradientes  $\frac{\partial E(W)}{\partial w_{ji}^s}$ ,  $\frac{\partial E(W)}{\partial w_{ji}^I}$  e  $\frac{\partial E(W)}{\partial a_{ji}}$  seguem as equações abaixo.

$$\frac{\partial E(W)}{\partial w_{ji}^s} = \delta_j^s(k) S_i(k); \begin{cases} 1 \leq i \leq n \\ 1 \leq j \leq r \end{cases} \quad (2.53)$$

$$\delta_j^s(k) = [y_j(k) - y_{mj}] l'(net_j^s(k)); \{1 \leq j \leq r\} \quad (2.54)$$

$$\frac{\partial E(W)}{\partial w_{ji}^I} = \delta_j^I(k) Q_j(k); \begin{cases} 1 \leq i \leq v \\ 1 \leq j \leq n \end{cases} \quad (2.55)$$

$$\delta_j^I(k) = \left\{ \sum_{i=1}^r [\delta_i^s(k) w_{ij}^s] \right\} h'(x_j(k)); 1 \leq j \leq n \quad (2.56)$$

$$Q_j(k) = a_{jj} Q_j(k-1) + u_j(k-1); \begin{cases} 1 \leq i \leq v \\ Q_j(0) = 0 \\ 1 \leq j \leq n \end{cases} \quad (2.57)$$

$$\frac{\partial E(W)}{\partial a_{ji}} = \delta_j^I(k) P_j(k); \begin{cases} 1 \leq i \leq n \\ 1 \leq j \leq n \end{cases} \quad (2.58)$$

$$P_j(k) = a_{jj} P_j(k-1) + x_j(k-1); \begin{cases} 1 \leq i \leq n \\ P_j(0) = 0 \\ 1 \leq j \leq n \end{cases} \quad (2.59)$$

### 2.3.1.1. Ajuste dos Parâmetros, Estabilidade do Modelo Neural e Convergência do Algoritmo de Aprendizado

O modelo neural com dinâmica interna mostrado nesta seção apresenta, na sua estrutura, uma equação a diferenças de primeira ordem, que é responsável pela dinâmica da rede (equação (2.24)). Devido a linearidade na equação citada, a

estabilidade do modelo pode ser determinada simplesmente pela verificação da posição dos autovalores da matriz de dinâmica  $A$  (equação (2.27)) no plano-Z.

O algoritmo de aprendizado tem a sua convergência determinada pela condição de módulo dos elementos da diagonal principal da matriz  $A$  serem menores que a unidade, ou seja que os elementos da diagonal principal de  $A$  estejam no interior do círculo unitário do plano-Z. Entretanto o próprio algoritmo modifica estes elementos da diagonal principal bem como todos os outros elementos de  $A$ . Por este motivo é desejável que o algoritmo inclua nas suas equações a possibilidade de não aceitar ajustes nos elementos da matriz que viole as condições de estabilidade do modelo e convergência do mesmo.

Neste trabalho não é considerada esta possibilidade. Para que o algoritmo de aprendizado tenha esta capacidade é necessário que o mesmo seja desenvolvido em conjunto com as técnicas de otimização com restrições (White e Jordan, 1992; Kirk, 1970; Athans e Falb, 1966), ou estimação paramétrica via projeção (Feng, 1993; Goodwin e Mayne, 1987; Ruiz-Vargas, 1997; Song, 1998) ou algoritmo genético (Montana e Davis, 1989; Petridis et al., 1992; Pedrycz, 1994; Linkens e Nyongesa, 1996).

## 2.4 Número de Parâmetros

Modelos com dinâmica interna, normalmente apresentam um menor número de parâmetros a serem ajustados quando comparados a modelos com dinâmica externa, o que possibilita um menor número de iterações durante o aprendizado. O uso de uma quantidade menor de parâmetros no modelo com dinâmica interna apresenta uma consonância com os estudos de Identificação de Sistemas onde o modelo a ser identificado deve ser escolhido de maneira a ter o menor número de parâmetros (Ljung, 1991).

Além da diferença citada acima, os modelos com dinâmica interna

possibilitam a utilização dos estados da rede que, a partir das definições de funções de saída e de ativação, mostradas na Figura 2. 1, ficam próximos do conceito usual de estados da Teoria de Controle (as vantagens de uso do espaço de estados no controle de processos são bastante conhecidas na literatura (Kailath, 1980; Chen, 1984)).

Para verificar a diferença entre o número de parâmetros a serem ajustados nas duas abordagens de rede neural (com dinâmica interna e com dinâmica externa), escolheu-se as seguintes topologias, para cada uma das RNA's à serem comparadas :

- a) RNA *feedforward* com dinâmica externa da Figura 2. 3 contendo na sua entrada  $m+1$  sinais atrasados de  $u(.)$  e  $n+1$  sinais atrasados da saída do processo  $y_{processo}(.)$ <sup>4</sup>,  $p$  neurônios estáticos intermediários e 1 neurônio estático de saída.
- b) RNA *feedforward* com dinâmica interna da Figura 2. 8 contendo  $n$  neurônios dinâmicos intermediários conectados uns aos outros acarretando em uma matriz  $A$  (equação (2.27)) cheia, e 1 neurônio estático de saída.

A rede do item (a) apresenta  $(n+m+3)p$  parâmetros a serem ajustados, enquanto a rede do item (b) apresenta  $n(n+2)$  parâmetros. Para que a rede com dinâmica interna do item (b) tenha um maior número de parâmetros do que a rede com dinâmica externa do item (a), é necessário que:

$$n(n+2) > (n+m+3)p \quad (2. 60)$$

ou

---

<sup>4</sup> Neste caso a rede neural está sendo utilizada para identificar um processo dinâmico, com o vetor regressor de entrada da rede sendo composto por amostras do sinal de entrada do processo  $u(k)$  e do sinal de saída do processo  $y(k)$ .

$$n(n+2) - (n+m+3)p > 0 \tag{2.61}$$

A verificação da equação (2.61) é realizada variando-se  $n$ ,  $m$ ,  $p$  dentro do intervalo [1,10]. Na tabela abaixo são mostrados 60, das 271 possíveis, situações em que a equação (2.61) é satisfeita.

n	m	p	$n(n+2) - (n+m+3)p$
2	1	1	2
2	2	1	1
3	1	2	1
3	4	1	5
3	6	1	3
3	8	1	1
4	1	1	16
4	2	1	15
4	2	2	6
4	3	2	4
4	4	1	13
4	6	1	11
4	7	1	10
4	9	1	8
5	1	1	26
5	1	3	8
5	2	3	5
5	3	2	13
5	3	3	2
5	5	1	22
5	5	2	9
5	7	2	5
5	9	2	1
5	10	1	17
6	1	1	38
6	1	3	18
6	2	4	1
6	3	3	12
6	4	1	35

n	m	p	$n(n+2) - (n+m+3)p$
6	5	2	20
6	5	3	6
6	6	3	3
6	7	2	16
6	8	1	31
6	10	1	29
7	1	1	52
7	1	4	19
7	1	5	8
7	2	4	15
7	2	5	13
7	3	4	11
7	4	4	7
7	5	2	33
7	6	3	15
7	8	2	27
7	10	1	43
7	10	3	3
8	1	4	32
8	1	6	8
8	2	6	2
8	3	2	52
8	3	5	10
8	4	4	20
8	4	5	5
8	5	4	16
8	6	4	12
8	7	4	3
8	8	4	1
8	9	3	20

n	m	p	$n(n+2) - (n+m+3)p$
8	10	1	59
8	10	3	17
9	1	2	73
9	2	3	57
9	2	5	29
9	3	4	39
9	4	5	3
9	5	5	14
9	6	3	45
9	6	5	9
9	7	1	80
9	7	5	4
9	8	1	79
9	9	4	15
9	10	1	77
10	1	5	60
10	2	5	45
10	3	5	24
10	4	6	18
10	4	7	1
10	6	1	101
10	6	5	6
10	7	1	100
10	7	3	60
10	8	5	15
10	9	1	98
10	9	5	10
10	10	1	97
10	10	4	28
10	10	5	5

Tabela 2. 1 - Verificação de situações onde o número de parâmetros da rede com dinâmica interna é maior que o número de parâmetros da rede com dinâmica externa ( $n$ ,  $m$  e  $p$  são definidos nos itens (a) e (b))

A partir dos resultados mostrados na Tabela 2. 1, verifica-se que :

a maioria das situações que satisfazem a equação (2.61) implica em um pequeno valor de  $p$  ( $p < 4$ ). Com este número de neurônios na camada intermediária, diminui consideravelmente a capacidade de realizar complexos

mapeamentos da rede neural. A literatura mostra que o número de neurônios estáticos na camada intermediária de uma RNA *feedforward* multi-camadas com dinâmica externa, para identificar sistemas dinâmicos não-lineares é maior do que 8, mesmo para sistemas de segunda ou terceira ordem (Bittanti e Piroddi, 1997; Noriega e Wang, 1998; Narendra e Parthasarathy, 1990; Sjöberg, 1995; Chen e Khalil, 1992; De Azevedo e Barreto, 1994; De Azevedo, 1993; De Oliveira et al., 1991; De Oliveira et al., 1998).

- ii. em um número considerável de situações, o valor de  $m$  é maior do que o de  $n$ , o que é difícil de encontrar em modelos não-lineares ou não. Se o processo é linear, esta seria uma situação onde o modelo apresenta um número de zeros maior que o número de pólos, o que torna o modelo não-causal, portanto de pouca utilidade em sistemas de controle.
- iii. a maioria das situações implica em um valor de  $n$  maior do que 6, indicando uma alta ordem para o modelo. Modelos desta ordem são poucos utilizados na literatura sobre identificação neural não-linear (IEEE Control System Magazine, 1988, 1989, 1990; Irwin et al., 1995; Gupta e Rao, 1994). Normalmente, trabalha-se com modelos de, no máximo, quarta ordem (isto não é válido para o estudo de séries temporais (Deffner, 1996)).

Todas estas evidências indicam que dificilmente, em aplicações de identificação neural não-linear, uma rede com dinâmica interna terá mais parâmetros do que uma rede com dinâmica externa.

## 2.5 Identificação Neural : Modelo com dinâmica interna

Nesta seção será mostrado o comportamento do modelo neural com dinâmica interna desenvolvido na seção 2.3 para representar o modelo não-linear de um sistema dinâmico discreto não-linear. O procedimento de identificação do modelo segue o padrão determinado na Figura 2.9.

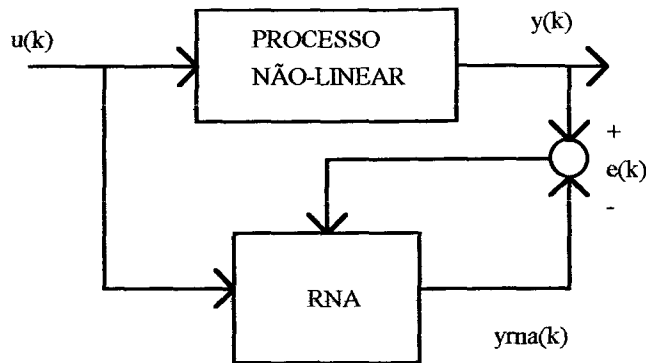


Figura 2. 9 - Identificação neural de um processo não-linear

### 2.5.1. Sistema Dinâmico Discreto Não-Linear

O sistema dinâmico é representado pelas equações

$$\begin{aligned} x_1(k+1) &= x_2(k) \\ x_2(k+1) &= 0.6x_1(k) + 0.3x_2(k) + f[u(k)] \end{aligned} \quad (2.62)$$

e a saída do sistema é dada por

$$y(k) = x_2(k) \quad (2.63)$$

com

$$f[u(k)] = 0.6 \text{sen}[u(k)] \quad (2.64)$$

onde  $x_1(k)$  e  $x_2(k)$  são os estados,  $u(k)$  é o sinal de entrada,  $y(k)$  é o sinal de saída do processo,  $k$  é o instante de amostragem e  $f[.]$  é uma função não-linear. Para o treinamento do modelo neural são utilizados 50 instantes de amostragem, com a entrada do tipo senoidal conforme equação abaixo;

$$u(k) = \text{sen}\left(\frac{2k\pi}{NPT}\right) \quad (2.65)$$

onde  $NPT$  equivale ao "número de pontos de treino", ou simplesmente número de amostras, onde  $NPT = 50$ . A Figura 2. 10 mostra a função não-linear  $f[u]$ .

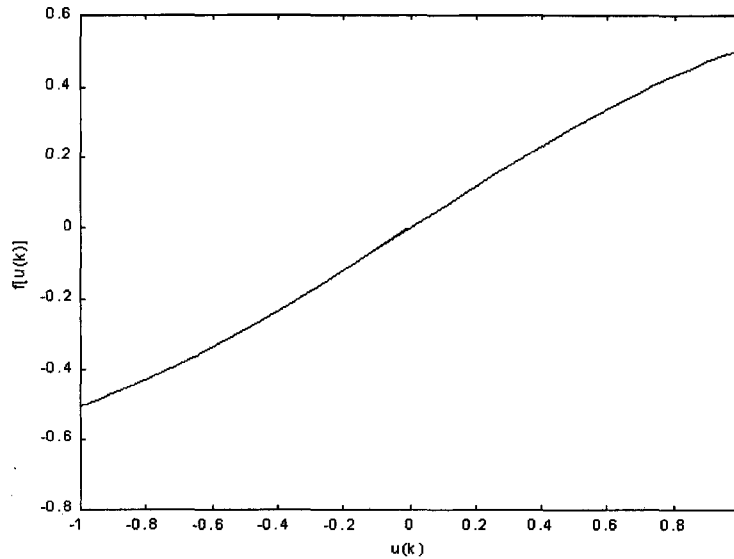


Figura 2. 10 - Relação não-linear  $f[u(k)]$  para o processo dinâmico identificado

### 2.5.2. Modelo Neural com dinâmica interna

A identificação do processo discreto não-linear descrito pela equação (2.62) através da RNA com dinâmica interna, segue a descrição da Figura 2. 8, onde observa-se a utilização apenas do sinal  $u(k)$  como entrada da rede. A rede neural utilizada é um modelo de 5 camadas de neurônios, onde a primeira camada funciona apenas como um *buffer* para o sinal de entrada, com o sinal de *bias*<sup>5</sup> em todas as camadas exceto na última camada. A rede é composta por :

- Primeira camada : um único neurônio com função ativação identidade, significando um neurônio estático, e função de saída do tipo linear;
- Segunda camada : oito neurônios com função ativação identidade, significando neurônios estáticos, e função de saída do tipo tangente hiperbólica;
- Terceira camada : quatro neurônios com função ativação identidade,

<sup>5</sup> O *bias* na camada é um sinal de valor constante unitário que se conecta com todos os neurônios da camada através de pesos, mas não recebe nenhuma ligação dos neurônios da camada anterior.



significando neurônios estáticos, e função de saída do tipo tangente hiperbólica;

- Quarta camada : dois neurônios com função ativação do tipo equação a diferenças de primeira ordem, onde o estado do neurônio é afetada apenas pelo seu próprio estado, conforme equação abaixo,

$$x_i(k+1) = a_{ii}x_i(k) + net_i(k) \quad (2.66)$$

que equivale a matriz  $A$  abaixo,

$$A = \begin{bmatrix} a_{11} & 0 \\ 0 & a_{22} \end{bmatrix} \quad (2.67)$$

e função de saída do tipo tangente hiperbólica;

- Quinta camada : um único neurônio com função ativação identidade, significando um neurônio estático, e função de saída do tipo linear com saturação no valor 10.

Neste primeiro exemplo de utilização do modelo neural com dinâmica interna não serão ajustados os parâmetros  $a_{ii}$ 's da matriz  $A$ , o que não ocorrerá nos próximos resultados mostrados neste trabalho. Ou seja, supondo algum conhecimento sobre o processo a ser identificado, utilizou-se os valores de:

$$A = \begin{bmatrix} 0.939 & 0 \\ 0 & -0.639 \end{bmatrix} \quad (2.68)$$

na rede neural, a partir de uma representação canônica diagonal para o processo da equação (2.62).

O resultado para a fase de aprendizado é mostrado na Figura 2.11 e na Figura 2.12 para um número de iterações igual a 13853. Para este número de iterações faz-se a medida do erro médio quadrático, conforme equação abaixo, cujo valor é igual a 0.3466.

$$EMQ = \sqrt{\frac{1}{NPT} \sum_{k=1}^{NPT} [y_i(k) - y_{RNA}(k)]^2} \tag{2.69}$$

A evolução da função E(W), que é o valor EMQ, em cada iteração de ajuste dos parâmetros do modelo neural, é mostrada na Figura 2.13.

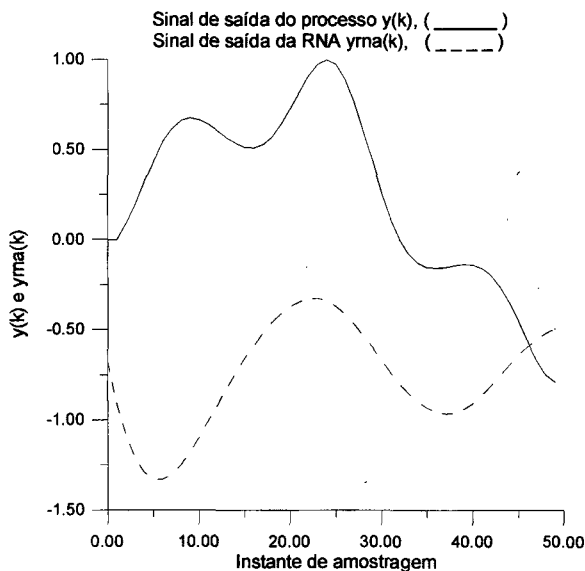


Figura 2.11 - Comportamento do Modelo Neural no início do aprendizado

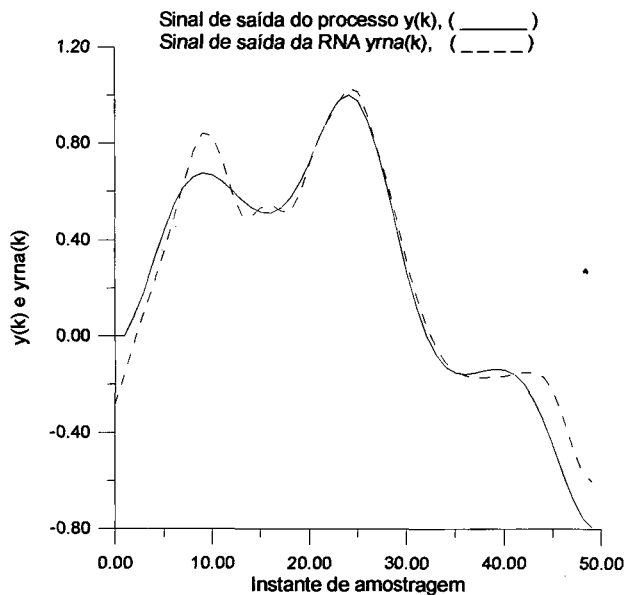


Figura 2.12 - Comportamento do Modelo Neural após aprendizado

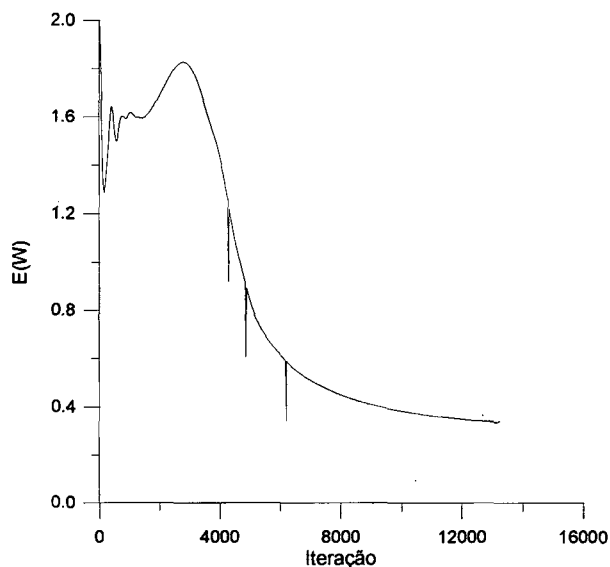


Figura 2. 13 - Evolução da Função  $E(W)$  durante o aprendizado

A curva da Figura 2. 13 indica que o aprendizado segue o padrão do algoritmo *backpropagation* (Rumelhart, 1986b; Nascimento Jr., 1994; De Azevedo, 1993). Com os parâmetros  $a_{ij}$ 's da matriz  $A$  fixos, o modelo neural não apresenta problemas de estabilidade.

## 2.6 Discussões

A utilização de RNA's em problemas de controle de processos não-lineares é uma realidade com inúmeras aplicações (Kawato et al., 1987; Chen et al., 1990a; De Oliveira et al., 1991; Gupta e Rao, 1994; Irwin et al., 1995; Gupta e Sinha, 1996; Bittanti e Piroddi, 1997; Song, 1998) e com vários resultados teóricos (Pineda, 1987; Narendra e Parthasarathy, 1990; Cichocki e Unbehauen, 1991; Levin e Narendra, 1991; Feng, 1993; Levin e Narendra, 1993; Sjöberg, 1995; Kurdila et al., 1995; Ruiz-Vargas, 1997; Yu e Annaswamy, 1997; Noriega e Wang, 1998). A maneira mais usada de se representar a informação de dinâmica nos modelos neurais é a abordagem entrada/saída, através da representação por dinâmica externa. Entretanto, buscando aplicar todos os recursos disponíveis da abordagem no espaço de estados, já consolidados para processos lineares, tem-se procurado

---

desenvolver redes que incluam a representação por dinâmica interna. Dentre os modelos neurais com dinâmica interna normalmente se utiliza sinais de realimentação da saída da rede ou da saída dos neurônios.

Neste trabalho mostra-se um novo modelo neural que realimenta o estado do neurônio e não o seu sinal de saída. Isto permite uma modelagem linear para a dinâmica interna do neurônio, facilitando o estudo de estabilidade da rede neural. Esta realimentação linear do estado do neurônio permite o uso da rede para identificar processos que possuam partes dinâmicas lineares e partes estáticas não-lineares ou outros processos não-lineares que não tenham esta particularidade. A identificação de um processo não-linear com a estrutura diferente da apresentada pelo modelo neural proposto será mostrada no próximo capítulo.

## 3 - Identificação Neural Não-Linear

### 3.1 Introdução

Ao se utilizar a Teoria de Controle, na prática, é necessário construir uma ponte entre o mundo real e a teoria matemática que rege o projeto de controladores. Esta ponte é o procedimento de modelagem ou identificação, onde o modelo descreve, segundo algum objetivo, o mundo real. A teoria matemática que mostra como modelos matemáticos de sistemas dinâmicos podem ser construídos a partir de medidas observadas é o que se costuma denominar de "*Identificação de Sistemas*".

Na identificação de sistemas, normalmente, utiliza-se um conjunto de modelos parametrizados, sendo necessário determinar uma "*estrutura do modelo*". A partir desta estrutura, dados são utilizados para encontrar o melhor modelo do conjunto considerado. A escolha da estrutura do modelo é determinada a partir de algum conhecimento prévio sobre o sistema que gera as medidas observadas. Quando não existe conhecimento sobre o sistema a ser modelado é comum utilizar um modelo *entrada-saída*.

O grau de conhecimento sobre o processo a ser identificado determina a classe de modelo a ser utilizado. De acordo com o nível de informações sobre o processo identificado pode-se classificar as classes de modelos em :

- modelos caixa-branca
- modelos caixa-cinza
- modelos caixa-preta

Os modelos caixa-branca indicam que existe um conhecimento total sobre o processo, ou seja, se conhece todas as relações entre as variáveis que descrevem o

comportamento dinâmico do sistema. Estes modelos não são realísticos porque, mesmo sabendo-se com exatidão as equações que regem a dinâmica do processo sempre existirão parâmetros que têm seus valores modificados com o passar do tempo, com a temperatura, etc.

Os modelos caixa-cinza indicam que existe algum conhecimento sobre o processo, mas não se conhecem alguns parâmetros ou algumas relações entre as variáveis que descrevem o comportamento dinâmico do sistema. Estes modelos são mais realísticos que os modelos caixa-branca.

Os modelos caixa-preta indicam uma falta total de conhecimento sobre o processo. Modelos caixa-preta pertencem à uma família de estrutura de modelos que apresentam uma flexibilidade adequada e são hábeis em aproximar uma grande classe de relações entrada/saída. Em outras palavras, modelo caixa-preta é uma estrutura padrão que pode ser utilizada para aproximar uma grande variedade de sistemas (Sjöberg, 1995).

Em aplicações reais é impossível obter uma estrutura de modelo que descreva o sistema de forma exata. Em vez de tentar encontrar esta cópia exata, faz-se algumas considerações sobre o sistema a ser identificado, tal que a estrutura do modelo seja a mais próxima do sistema real. Uma consideração utilizada na identificação de sistemas é que o sistema desconhecido seja linear. Isto é muito difícil de ser verdade no mundo real, mas em muitos casos esta consideração é uma aproximação adequada.

A teoria de sistemas lineares é bem desenvolvida e existem inúmeros resultados que podem ser aplicados ao se utilizar modelos lineares na identificação de sistemas (Ljung e Söderström, 1983; Ljung, 1987; Goodwin e Payne, 1977; Söderström e Stoica, 1989; Johansson, 1993). Entretanto, se o modelo é necessariamente não-linear ou se é relaxada a consideração de linearidade, torna-se extremamente difícil trabalhar com estes modelos não-lineares. As dificuldades são decorrentes do estudo de estabilidade do modelo e da escolha da estrutura do

modelo - "Uma função não-linear pode ser não-linear de várias maneiras" (Sjöberg, 1995).

A utilização de modelos entrada-saída em identificação linear através de modelos AR (AutoRegressive), ARMA (AutoRegressive Moving Average) e ARMAX (AutoRegressive Moving Average with eXogenous inputs) não apresenta dificuldades. Entretanto o uso destes modelos nas suas versões não-lineares, NAR (Non-linear AutoRegressive), NARMA (Non-linear AutoRegressive Moving Average), NARMAX (Non-linear AutoRegressive Moving Average with eXogenous inputs), apresenta grandes dificuldades. O motivo destas dificuldades já foi descrito na frase de Sjöberg no parágrafo anterior. Devido a este fato, alguns pesquisadores têm apresentado vários trabalhos que tratam apenas classes de processos não-lineares (Beghelli e Giudorzi, 1976; Thathachar e Ramaswamy, 1973; Billings e Fakhouri, 1979; Simpson, 1973; Billings e Fakhouri, 1982). E mais recentemente tem-se utilizado modelos de RNA's para identificar sistemas não-lineares.

As RNA's têm sido bastante exploradas na identificação de sistemas dinâmicos não-lineares (Sjöberg, 1995; Narendra e Levin, 1995; Narendra e Parthasarathy, 1990; Karakasoglu et al., 1993; Saxén e Saxén, 1994; Tsung, 1994; Billings e Chen, 1995; Hunt e Sbarbaro, 1995), devido as mesmas serem inerentemente modelos caixa-preta não-lineares e também por terem a habilidade de aproximar complexos mapeamentos não-lineares.

Com a capacidade de realizar complexos mapeamentos não-lineares, mesmo com pouco ou nenhum conhecimento sobre a planta não-linear, as RNA's permitem superar problemas causados por dinâmicas não-modeladas, comumente encontrados quando utilizam-se modelos lineares. Com o processo de adaptação do aprendizado, as RNA's podem enfrentar a questão da variação paramétrica do processo. Entretanto, em decorrência do aprendizado ser um procedimento de otimização, aparecem dificuldades que são comuns nos métodos numéricos de minimização ou maximização de função, tais como : escolha da condição inicial, escolha do tamanho do passo na direção do gradiente, etc. A estes problemas

acrescenta-se o efeito da sobre-parametrização decorrente da escolha do número de neurônios no modelo neural. A sobre-parametrização é decorrência natural da falha na escolha da estrutura do modelo.

### 3.2 Modelagem Através de Redes Neurais Artificiais

A propriedade de mapeamento não-linear das RNA's é o principal motivo da sua utilização em controle de processos. O treinamento de uma RNA utilizando dados entrada-saída de um processo não-linear pode ser considerado como um problema de aproximação de um funcional não-linear. A teoria de aproximação é uma área de estudos clássica da matemática. Desde o famoso Teorema de Weierstrass (Rudin, 1976) é conhecido que polinômios podem aproximar tão bem quanto possível uma função contínua. Mais recentemente, tem sido feito um esforço considerável na aplicação destas ferramentas matemáticas na investigação da capacidade de aproximação das RNA's (Cybenko, 1988; Cybenko, 1989; Funahashi, 1989; Hornik et al., 1989; Turchetti et al., 1998). Estes trabalhos provam que uma função contínua pode ser arbitrariamente bem aproximada por uma RNA feedforward multicamada com somente uma camada intermediária de neurônios (nestas pesquisas as RNA's são estudadas como modelos paramétricos).

A RNA representa um modelo paramétrico, sempre que sua topologia (número de camadas e número de neurônios) for definida previamente, independente do problema de aproximação. Por outro lado, se a topologia da RNA puder ser definida em função do problema de aproximação, a mesma representa um modelo não-paramétrico.

A dificuldade com os modelos paramétricos deve-se ao fato de, para um dado problema de aproximação, não se conhecer o número ótimo de neurônios nem o número ótimo de camadas. Ou seja, a precisão do aproximador neural depende do número de elementos de cada camada de neurônios. Algumas linhas de pesquisa interessante sobre este tema podem ser encontradas em Chester (Chester, 1990), onde é mostrado um suporte teórico para a observação empírica de que RNA's com



duas camadas intermediárias parecem ter maior precisão e melhor generalização que uma RNA com uma camada intermediária.

De uma forma geral, existe uma grande quantidade de resultados teóricos relatando aproximação de funções através de RNA's (Hunt e Sbarbaro, 1995; Hornik et al., 1989; Girosi e Poggio, 1989; Hecht-Nielsen, 1987). Estes resultados, a princípio, determinam importantes teoremas que auxiliam, em última análise, explicar o desempenho das RNA's. Mas os mesmos não são construtivos porque não definem a arquitetura (se a rede é direta ou recorrente) e a topologia de uma RNA apropriada para um dado problema.

### 3.2.1. Estruturas Não-Lineares

Da mesma maneira que a função de transferência é uma representação genérica para modelos lineares, a RNA é uma representação genérica para modelos não-lineares. Há algum tempo, conhece-se que a estrutura de modelo ARX (AutoRegressive with eXogenous inputs) é capaz de descrever qualquer sistema linear corrompido por ruído aditivo (Ljung e Wahlberg, 1992). Uma razão para considerar modelos mais elaborados, tais como o ARMAX e os modelos BJ (Box-Jenkin) é que os mesmos podem fornecer uma representação mais sucinta, onde o mesmo resultado de aproximação é obtido com menos parâmetros. Estes modelos caixa-preta lineares diferem, entre eles, somente no que diz respeito ao tratamento do ruído (no caso livre de ruído, eles são equivalentes).

Nesta seção apresenta-se uma família de modelos não-lineares, retirados do trabalho de Sjöberg (Sjöberg, 1995), onde, da mesma forma que no caso linear, a diferença entre eles está na forma como o ruído é modelado.

No problema de identificação de processos dinâmicos, as verdadeiras propriedades do ruído são normalmente desconhecidas. Sendo assim, a família de modelos, baseados em diferentes considerações para o ruído, deve ser vista como uma família de razoáveis modelos candidatos, mais do que modelos que reflitam uma descrição verdadeira do ruído.

Um objetivo da identificação é gerar um bom modelo preditor, isto é, um modelo que estime a saída do processo  $y(t)$ ,

$$E[y(t) | \varphi(t)] \quad (3.1)$$

onde  $\varphi(t)$  contém informações disponíveis até o instante  $t$ , e é chamado de vetor de regressão. Considere a estrutura de modelo parametrizado candidata  $g(\theta, \varphi(t))$  onde  $\theta$  é o vetor de parâmetros do modelo; o objetivo da estimação é encontrar uma estimativa  $\hat{\theta}$  tal que os dados possam ser descritos de acordo com,

$$y(t) = g(\theta, \varphi(t)) + d(t) \quad (3.2)$$

onde  $d(t)$  é uma seqüência ruído branco cuja variância é minimizada com respeito a algum critério definido (Sjöberg, 1995).

Ao propor um modelo candidato igual ao mostrado na equação (3.2), surgem algumas questões a serem resolvidas :

1. Que variáveis, construídas a partir dos dados observados passados, devem ser escolhidas para compor o vetor regressor  $\varphi(t)$  ?
2. Como deve ser escolhido o mapeamento não-linear  $g(\cdot, \cdot)$  do espaço determinado pelo vetor regressor  $\varphi(t)$  para o espaço de saída ?

A primeira questão surge na modelagem não-linear, sendo que o modelo linear é completamente especificado pela escolha de  $\varphi(t)$ . A segunda questão não aparece em sistemas de identificação linear, já que  $g(\cdot, \cdot)$  é linear em  $\varphi(t)$ . Entretanto, para modelos não-lineares existem muitas possibilidades de escolha para  $g(\cdot, \cdot)$ . Através da força bruta, testes de tentativa e erro, é possível fazer a escolha de  $g(\cdot, \cdot)$  a partir de um conjunto de mapeamentos não-lineares disponíveis. As RNA's com dinâmica interna ou externa, são algumas das  $g(\cdot, \cdot)$  possíveis. Ao se concluir este procedimento de escolha, tem-se a estrutura de modelo mais geral possível para um vetor de regressão particular.

O prejuízo do uso de um modelo com desnecessária flexibilidade, determinado pelo número de parâmetros, é que pode-se aumentar a contribuição da variância do erro na aproximação, sem redução substancial da polarização (Sjöberg, 1995).

### 3.2.1.1. Importância do Vetor Regressor $\varphi(t)$

Para identificação de sistemas lineares, cada adição de um elemento no vetor regressor, significa mais um parâmetro a ser estimado. Se o sistema pode ser descrito por um vetor regressor menor, então isto diminui a contribuição negativa da variância do erro de estimação na identificação (Sjöberg, 1995). Esta conclusão permanece nos modelos não-lineares, sendo que existe uma outra razão, mais forte ainda, para restringir o tamanho do vetor de regressão na identificação de sistemas não-lineares. Uma vez que  $\varphi(t)$  tenha sido escolhido, a função,

$$g : \varphi(t) \rightarrow \hat{y}(t) \quad (3.3)$$

deve ser estimada com  $d = \dim[\varphi(t)]$ . Isto é um problema de estimação de uma função não-linear que mapeia  $\mathbb{R}^d \rightarrow \mathbb{R}$ , tornando a questão da estimação mais complexa a medida que  $d$  aumenta. Isto é a famosa “*maldição da dimensionalidade*”, tão importante em técnicas de otimização, como a “*programação dinâmica*”, e é a segunda razão para a escolha de um vetor de regressão com o menor número de elementos possíveis. Um estudo preliminar sobre a escolha do tamanho do vetor de regressão, para modelos neurais com dinâmica externa, pode ser encontrado em Narendra (Narendra e Levin, 1995).

Uma alternativa à estas considerações sobre o vetor de regressão pode ser a restrição dos tipos de funções  $g(\cdot, \cdot)$  a serem utilizadas. Muitas das estruturas de modelos que são abordadas nesta seção podem ser vistas como restrições em  $g(\cdot, \cdot)$ , motivadas pela modelagem do ruído. Na identificação linear, isto equivale a escolha do uso de um modelo entrada-saída ou do uso de um modelo no espaço de estados (Sjöberg, 1995).

Um modelo no espaço de estados pode ser visto como um caso especial de um modelo entrada-saída, onde os estados são saídas virtuais, construídas apenas para servirem como elementos regressores no próximo instante de tempo.

- Modelo discreto entrada-saída

$$y(k) = f(y(k-1), \dots, y(k-n), u(k-1), \dots, u(k-m)) \quad (3.4)$$

- Modelo discreto no espaço de estados

$$x(k+1) = h[x(k), u(k)] \quad (3.5)$$

$$y(k) = l[x(k)] \quad (3.6)$$

ou

$$y(k) = l[h[x(k-1), u(k-1)]] \quad (3.7)$$

que torna-se

$$y(k) = N[x(k-1), u(k-1)] \quad (3.8)$$

Com este ponto de vista, para identificação não-linear pode-se fazer a seguinte pergunta : Existe alguma vantagem em utilizar saídas virtuais como elementos regressores em vez de utilizar valores atrasados da saída original ? Uma boa resposta a esta pergunta pode ser : Com saídas virtuais, ou seja com um modelo no espaço de estados, é possível obter uma descrição mais sucinta do sistema, tendo um vetor de regressão de menor dimensão quando comparado com o seu possível modelo entrada-saída equivalente (Sjöberg, 1995).

### 3.2.1.2. Modelos Entrada-Saída

A escolha de uma estrutura de modelo através da abordagem entrada-saída, pode ser dividida na escolha de um vetor regressor e na escolha de uma função não-linear. O vetor de regressão  $\varphi(t)$  deve ter as seguintes propriedades :

- pequena dimensão
- não depender do vetor de parâmetros  $\theta$ , isto é, o modelo não deve ser recorrente

Estes objetivos servem como um primeiro passo do procedimento de identificação não-linear (Sjöberg, 1995).

### 3.2.1.2.1. Escolha do Vetor Regressor

Da identificação de sistemas lineares, tem-se que os possíveis vetores de regressão podem ser aqueles que têm como elementos (Sjöberg, 1995) :

- i) entradas passadas  $u(k-i)$ , que representam os modelos de *Resposta Impulsiva Finita* (FIR - Finite Impulse Response);
- ii) saídas medidas passadas  $y(k-i)$ ;
- iii) saídas simuladas passadas  $\hat{y}(k-i|\theta)$ ;
- iv) resíduos passados,  $\varepsilon(k-i) = y(k-i) - \hat{y}(k-i|\theta)$ , utilizando o modelo corrente;
- v) resíduos simulados passados,  $\varepsilon_u(k-i) = y(k-i) - \hat{y}_u(k-i|\theta)$ , utilizando somente entradas passadas e o modelo corrente<sup>1</sup>;

Seguindo a nomenclatura dos modelos lineares (Ljung e Söderström, 1983), pode-se dividir os modelos não-lineares em vários grupos diferentes dependendo da escolha do vetor de regressão, como pode ser visto em Chen (Chen et al., 1990b; Chen e Billings, 1992). Dependendo da escolha de  $\varphi(t)$  os modelos são divididos em;

- Modelos NFIR (Non-linear Finite Impulse Response), os quais utilizam somente  $u(k-i)$  como regressores.
- Modelos NARX, os quais utilizam  $u(k-i)$  e  $y(k-i)$  como regressores.

<sup>1</sup> A saída simulada  $\hat{y}_u$  é obtida como a saída de;

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

onde  $q$  é o equivalente no tempo do operador  $z^{-1}$  no domínio  $z$ , substituindo  $\varepsilon$  e  $\varepsilon_u$  por zeros no vetor de regressão  $\varphi(k, \theta)$ .

- Modelos NOE (Non-linear Output Error), os quais utilizam  $u(k-i)$  e  $\hat{y}_u(k-i|\theta)$  como regressores. Neste caso a saída do modelo é  $\hat{y}_u(k|\theta)$ .
- Modelos NARMAX, os quais utilizam  $u(k-i)$ ,  $y(k-i)$  e  $\varepsilon(k-i)$  como regressores.
- Modelos NBJ (Non-linear Box-Jenkin), os quais utilizam  $u(k-i)$ ,  $\hat{y}(k-i|\theta)$ ,  $\varepsilon(k-i|\theta)$  e  $\varepsilon_u(k-i|\theta)$  como regressores.

Em Narendra (Narendra e Parthasarathy, 1990) outra notação é utilizada para estes mesmos modelos. O modelo NARX é chamado modelo Série-Paralelo e o modelo NOE é chamado modelo Paralelo. Os modelos NOE, NBJ e NARMAX correspondem a RNA's recorrentes, porque partes da entrada da RNA (o vetor regressor) consistem de saídas passadas da rede. Nestes casos, torna-se difícil determinar sobre que condições o modelo preditor obtido é estável e é feito um esforço extra para calcular o gradiente correto para o procedimento iterativo de aprendizado que ajusta os parâmetros do modelo.

### 3.2.1.3. Modelos no Espaço de Estados

O objetivo com o modelo através da abordagem do espaço de estados, é o mesmo do modelo através da abordagem entrada-saída, ou seja, obter uma boa estimativa dos valores de saída futura através do uso de informações conhecidas. Supondo que os dados podem ser descritos por uma equação de estados na forma :

$$x(k+1) = f(x(k), u(k), w(k)) \quad (3.9)$$

$$\hat{y}(k) = g(x(k), u(k)) + d(k) \quad (3.10)$$

para alguma escolha das funções  $f(\cdot)$  e  $g(\cdot)$ , e para alguma sequência de ruído branco gaussiano  $w(k)$  e  $d(k)$ . Neste caso existem dois lugares, no sistema, onde o ruído atua: um nos estados do sistema e um na saída do sistema. O ruído que aparece na atualização dos estados,  $w(k)$ , é chamado de "ruído do processo", e o ruído na equação de saída,  $d(k)$ , é chamado de "ruído de medida".

Em geral, modelos no espaço de estados correspondem a modelos recorrentes, os quais fazem a estimação dos parâmetros de uma forma mais elaborada do que, por exemplo, o modelo NARX.

Como foi dito anteriormente, um modelo no espaço de estados pode ser visto como um caso especial de um modelo entrada-saída, onde os estados são saída virtuais adicionais, construídas somente para serem utilizadas como regressores no próximo instante de amostragem (equação (3.3) e equação (3.8)). Conseqüentemente, modelos entrada-saída podem ser vistos como casos genéricos dos modelos no espaço de estados. Um modelo NARX;

$$\hat{y}(k) = g(\theta, \varphi(k)) \tag{3.11}$$

$$\varphi(k) = [y(k-1), \dots, y(k-n), u(k-1), \dots, u(k-m)] \tag{3.12}$$

pode ser convertido na forma de espaço de estados como visto a seguir.

Seja o vetor de estados composto por duas partes :

$$x(k) = \begin{bmatrix} x^1(k) \\ x^2(k) \end{bmatrix} \tag{3.13}$$

então o modelo NARX pode ser escrito como;

$$x^1(k+1) = \begin{bmatrix} 0 & 0 & 0 & \dots & 0 & 0 & 0 \\ 1 & 0 & 0 & \dots & 0 & 0 & 0 \\ 0 & 1 & 0 & \dots & 0 & 0 & 0 \\ \dots & & & & & & \\ 0 & 0 & 0 & \dots & 0 & 0 & 0 \\ 0 & 0 & 0 & \dots & 1 & 0 & 0 \\ 0 & 0 & 0 & \dots & 0 & 1 & 0 \end{bmatrix} x^1(k) + \begin{bmatrix} 1 \\ 0 \\ 0 \\ \dots \\ 0 \\ 0 \\ 0 \end{bmatrix} y(k) \tag{3.14}$$

$$x^2(k+1) = \begin{bmatrix} 0 & 0 & 0 & \dots & 0 & 0 & 0 \\ 1 & 0 & 0 & \dots & 0 & 0 & 0 \\ 0 & 1 & 0 & \dots & 0 & 0 & 0 \\ \dots & & & & & & \\ 0 & 0 & 0 & \dots & 0 & 0 & 0 \\ 0 & 0 & 0 & \dots & 1 & 0 & 0 \\ 0 & 0 & 0 & \dots & 0 & 1 & 0 \end{bmatrix} x^2(k) + \begin{bmatrix} 1 \\ 0 \\ 0 \\ \dots \\ 0 \\ 0 \\ 0 \end{bmatrix} u(k) \quad (3.15)$$

$$\hat{y}(k) = g(\theta, x(k)) \quad (3.16)$$

onde a escolha das variáveis de estado será mostrada na próxima seção. Os outros modelos entrada-saída também podem ser convertidos na forma de espaço de estados da mesma maneira.

O modelo representado pela equação (3.13) e pela equação (3.16) é, entretanto, um caso particular de modelo no espaço de estados. A atualização do estado, equação (3.14) e equação (3.15), é linear com parâmetros ajustáveis, sendo que esta atualização pode ser vista como uma maneira de selecionar os elementos do vetor regressor para a função de saída  $g(\cdot)$ , representada pela equação (3.16). Convém observar que este tipo particular de modelo no espaço de estados pode ser representado através de uma RNA com dinâmica interna, como mostrado na seção (2.3).

Os modelos no espaço de estados onde a atualização dos estados é fixada, podem ser vistos como uma pré-filtragem dos dados para a obtenção dos elementos do vetor regressor (Sjöberg, 1995; Walberg, 1991).

A mudança de um modelo entrada-saída para um modelo no espaço de estados pode ser vista como a retirada da flexibilidade da equação (3.16), e colocando-a na equação (3.14) e na equação (3.15). Ao incluir parâmetros na atualização dos estados, tem-se uma representação de estados que é adaptada a partir dos dados (Sjöberg, 1995). Isto pode fornecer uma descrição mais sucinta para o sistema, com um menor número de elementos no vetor regressor.



### 3.3 Formas de Identificação Neural

Embora existam várias questões teóricas em aberto, os resultados discutidos nas seções anteriores demonstram que as RNA's têm um uso bastante promissor na identificação de sistemas dinâmicos não-lineares. Sem referenciar uma estrutura de RNA particular, esta seção discutirá algumas arquiteturas para treinamento de RNA's de forma que as mesmas representem sistemas dinâmicos não-lineares e seus modelos inversos.

Uma questão importante na identificação de sistemas é a "*identificabilidade*" do sistema (Ljung, 1987; Ljung e Söderström, 1983; Söderström e Stoica, 1989), ou em outras palavras: dada uma estrutura de modelo particular, o sistema em estudo pode ser representado com esta estrutura? Na falta de resultados teóricos concretos para responder esta pergunta quando se utiliza RNA's, embora Narendra (Narendra e Levin, 1995) tenha alguns teoremas preliminares sobre este assunto, considera-se que os sistemas a serem estudados pertencem a uma classe de sistemas para a qual a RNA escolhida está apta a representar.

#### 3.3.1. Modelagem Direta

Uma estrutura para treinar uma RNA de tal modo que a mesma represente a dinâmica direta de um sistema, é mostrada na figura a seguir. A RNA é colocada em paralelo com o sistema e o erro entre a saída do sistema e a saída da rede (erro de predição ou resíduo) é utilizado como sinal de ajuste dos pesos da RNA.

Como observado por Jordan (Jordan e Rumelhart, 1992), esta arquitetura é um problema clássico de aprendizado supervisionado, onde o professor (o sistema) provê os valores desejados (suas saídas) que servem para adaptar o aprendiz (a RNA). No caso particular de uma RNA feedforward com dinâmica externa, a retropropagação do erro de predição (*backpropagation*) através da rede fornece um possível algoritmo de aprendizado.

Uma questão importante é a natureza dinâmica do sistema estudado e como caracterizar esta dinâmica na RNA. Uma possibilidade de fazer esta caracterização é introduzir dinâmica na RNA através de laços de realimentação, determinando assim o uso de RNA's recorrentes (visto na seção (2.2.2)). Outras possibilidades de uso de RNA's recorrentes podem ser encontradas nos trabalhos de Zbikowski (Zbikowski, 1994) e Roisenberg (Roisenberg, 1998). Uma outra maneira de introduzir a informação de dinâmica na RNA é através do uso de um conjunto de sinais atrasados no tempo na entrada da RNA (mostrado na seção (2.2.1)). Outra forma de caracterizar a dinâmica na RNA é através do uso de neurônios dinâmicos, visto na seção (2.3) e mostrado nos trabalhos de Willis (Willis et al., 1991), Karakasoglu (Karakasoglu et al., 1993) e Ku (Ku e Lee, 1995).

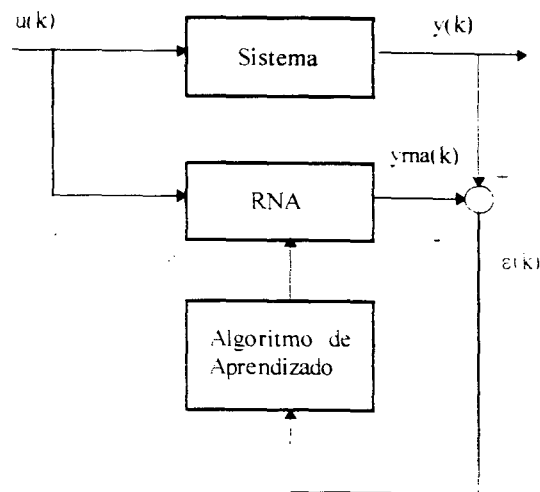


Figura 3. 1 - Identificação Neural Através da Modelagem Direta

Nesta discussão, é utilizada a representação de dinâmica na RNA, através do conjunto de sinais atrasados no tempo na entrada da rede, sendo também possível utilizar a representação através de neurônios dinâmicos, com a passagem do modelo entrada-saída para o modelo no espaço de estados (mostrado na seção (2.1.3)).

Supondo que o sistema é governado pela seguinte equação à diferenças não-linear:

$$y(k+1) = f(y(k), \dots, y(k-n+1), u(k), \dots, u(k-m+1)) \quad (3.17)$$

deste modo, a saída do sistema,  $y(\cdot)$ , no instante  $k+1$  depende (como definido pelo mapeamento não-linear  $f(\cdot)$ ) dos valores  $n$  passos atrás da saída do sistema e dos valores  $m$  passos atrás da entrada do sistema. Aqui não serão representados os distúrbios do sistema. Métodos que incluem distúrbios podem ser encontrados em Chen (Chen et al., 1990b), no capítulo 3 de Sjöberg (Sjöberg, 1995) e em Hunt (Hunt e Sbarbaro, 1995).

A representação no espaço de estados para o sistema descrito na equação (3.17) pode ser obtido através da definição das seguintes variáveis de estado:

$$\begin{aligned} x_1^1(k) &= y(k) & x_1^2(k) &= u(k) \\ x_2^1(k) &= y(k-1) & x_2^2(k) &= u(k-1) \\ &\dots & & \\ x_n^1(k) &= y(k-n+1) & x_m^2(k) &= u(k-m+1) \end{aligned} \quad (3.18)$$

cuja equação de estado é,

$$\begin{bmatrix} x^1(k+1) \\ x^2(k+1) \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & \dots & 0 & 0 \\ 1 & 0 & 0 & \dots & 0 & 0 \\ 0 & 1 & 0 & \dots & 0 & 0 \\ \dots & & & \dots & & \\ 0 & 0 & 0 & \dots & 0 & 0 \\ 0 & 0 & 0 & \dots & 1 & 0 \end{bmatrix} \begin{bmatrix} x^1(k) \\ x^2(k) \end{bmatrix} + \begin{bmatrix} f(x(k), u(k)) \\ 0 \\ 0 \\ \dots \\ 0 \\ u(k) \\ 0 \\ 0 \\ \dots \\ 0 \\ 0 \end{bmatrix} \quad (3.19)$$

definindo o vetor de estados,

$$x(k) = \begin{bmatrix} x^1(k) \\ x^2(k) \end{bmatrix} \quad (3.20)$$

a equação (3.19) fica na sua forma mais geral, definida por:

$$\begin{aligned} x(k+1) &= F[x(k), u(k)] \\ y(k) &= h[x(k)] \end{aligned} \quad (3.21)$$

Casos especiais do modelo dado pela equação (3.21) podem ser vistos em Narendra (Narendra e Parthasarathy, 1990) onde a saída do sistema apresenta uma relação linear com valores passados de  $y$  e  $u$ , ou na seção (2.2.3.1) onde a atualização dos estados é linear em relação aos valores passados de  $y$  e  $u$ .

A abordagem mais utilizada na literatura para as RNA's, e a mais óbvia, é a entrada-saída (seção (2.2.2)). Chamando a saída da rede de  $y_m$ , tem-se:

$$y_m(k+1) = g(y(k), \dots, y(k-n+1), u(k), \dots, u(k-m+1), W) \quad (3.22)$$

onde  $g(\cdot)$  representa o mapeamento não-linear realizado pela RNA (isto é, a aproximação de  $f(\cdot)$ ) e  $W$  é a matriz de parâmetros ajustáveis da rede (isto é, os pesos da RNA) através do algoritmo de aprendizado, sendo que o mais utilizado é o algoritmo de aprendizado *backpropagation* (Rumelhart et al., 1986a), embora existam outros métodos de aprendizado também empregados para esta estrutura de modelo neural (Chen et al., 1990b; Chen et al., 1990c). Note que a entrada da RNA inclui os valores passados da saída do sistema "real", ou seja, a RNA não apresenta laços de realimentação, sendo assim um modelo de RNA com dinâmica externa.

Supondo que, após um período de treinamento suficiente, a RNA consiga uma boa representação do sistema, isto é,  $y_m(\cdot) \approx y_r(\cdot)$ , então pode-se propor um outro treinamento subsequente, onde a saída da RNA fornecerá os sinais atrasados no tempo para a sua própria entrada. Nesta situação tem-se uma RNA recorrente, com laços de realimentação, que pode ser utilizada independentemente do sistema "real" para a qual foi treinada. Esta outra estrutura de RNA é descrita por:

$$y_m(k+1) = g(y_m(k), \dots, y_m(k-n+1), u(k), \dots, u(k-m+1), W) \quad (3.23)$$

O modelo neural dado pela equação (3.23) também pode ser utilizada para treinar a rede, sem o pré-treinamento citado acima. Esta possibilidade foi discutida por Narendra (Narendra e Parthasarathy, 1990), sendo que o algoritmo de treinamento derivado desta estrutura é muito “pesado” computacionalmente, devido o cálculo de algumas matrizes Jacobianas, que são os parâmetros de um sistema de equações à diferenças lineares de primeira ordem, cuja solução é necessária para realizar o ajuste dos pesos da rede. Uma possível alternativa para a estrutura determinada pela equação (3.23) é a utilização de modelos de RNA com neurônios dinâmicos da seção (2.3).

Os trabalhos de Narendra apresentam alguns resultados teóricos preliminares quanto a controlabilidade, estabilidade e identificabilidade dos modelos dados pela equação (3.22) (Narendra e Levin, 1995; Levin e Narendra, 1993). Existe, também, um extenso trabalho de Zbikowski (Zbikowski, 1994) sobre o uso de RNA recorrentes em sistemas de controle, com resultados teóricos sobre algumas condições de fraca (*weak*) controlabilidade e observabilidade utilizando a álgebra de Lie (Hermann e Krener, 1977). Embora a abordagem neural apresentada na seção (2.3) possa ser diretamente correlacionada com os filtros lineares ortonormais de Laguerre e Kautz, que têm aplicações extensivamente discutidas nos trabalhos de Wahlberg (Wahlberg, 1991; Wahlberg, 1994) e algumas generalizações no artigo de van den Hof (van den Hof et al., 1994), a mesma não apresenta na literatura algum desenvolvimento teórico.

No contexto de identificação de sistemas lineares invariantes no tempo, as duas estruturas de modelo, a equação (3.22) e a equação (3.23), têm sido extensivamente consideradas (Narendra e Annaswamy, 1989). Estas duas estruturas também têm sido discutidas na literatura de processamento de sinais (Widrow e Stearns, 1985). A estrutura da equação (3.22) (chamada de modelo Série-Paralelo por Narendra e chamada de modelo NARX por Sjöberg) é defendida no contexto de identificação pelos resultados de estabilidade. De outra maneira, a equação (3.23)

(chamada de modelo Paralelo por Narendra e chamada de modelo NOE por Sjöberg) pode ser preferida quando trabalha-se com ruído no sistema, uma vez que isto evita problemas de polarização, causados por ruído na saída de um sistema real (Widrow e Stearns, 1985; Sjöberg, 1995).

### 3.3.2. Modelagem Inversa

Modelos inversos de sistemas dinâmicos são importantes na tarefa de controlar processos e na literatura existem algumas estratégias de controle que os utilizam (Kawato et al., 1987; Hunt e Sbarbaro, 1991, De Azevedo, 1993). A seguir são mostradas algumas das abordagens de obtenção de modelos inversos.

Conceitualmente, a abordagem mais simples é a “*modelagem inversa direta*”, como mostrado na figura abaixo, que também é chamada de “*aprendizado inverso generalizado*” (Psaltis et al., 1988; De Oliveira et al., 1991), onde tem-se um conjunto de treinamento específico que é conseguido para uma dada entrada do sistema. Nesta abordagem a saída do sistema é utilizada como entrada da RNA e a saída da RNA é comparada com a entrada do sistema, e este erro modifica os pesos da RNA. Esta estrutura força a RNA a representar o modelo inverso do sistema, mas apresenta algumas desvantagens;

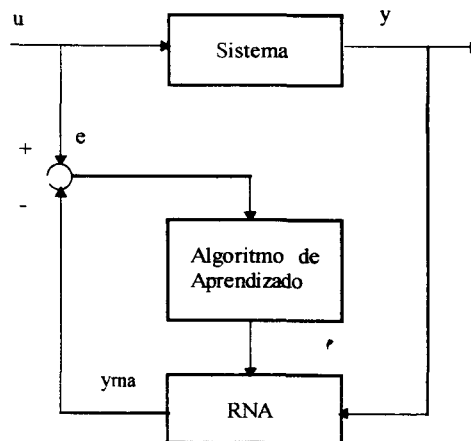


Figura 3.2 - Modelagem Neural Inversa

- O procedimento de aprendizado não é feito de forma direcionada a uma meta específica (Jordan e Rumelhart, 1992) e o conjunto de treino deve ser escolhido sob o conjunto de todas as possíveis entradas do sistema, tarefa esta que nem sempre é possível de se realizar *a priori*. Este procedimento não é direcionado a uma meta específica porque conceitualmente controlar um sistema é fazer com que a saída deste sistema comporte-se de um determinada maneira, o que não acontece aqui, onde a RNA gera um sinal correspondente ao sinal de entrada do sistema, não interessando para onde vai a saída do sistema excitado por esta entrada.
- Se o mapeamento não-linear inverso não é único, então modelos inversos incorretos podem ser obtidos. Este é um problema clássico em robótica (Campos, 1990).

O primeiro ponto citado acima, está fortemente entrelaçado com o conceito geral de “*excitação persistente*” (ou entradas persistentemente excitantes), sendo que na literatura de controle adaptativo, condições que assegurem excitação persistente, o que resulta na convergência da identificação paramétrica para modelos lineares, são bem estabelecidas (Åström e Wittenmark, 1995; Ljung, 1987). Para o uso de RNA's em identificação, também são desejados métodos que caracterizem excitação persistente, entretanto, o que existe são algumas discussões preliminares sobre esta questão (Narendra e Parthasarathy, 1990; Kurdila et al., 1995).

Uma segunda abordagem para a modelagem inversa, que evita os problemas citados anteriormente, é conhecida como “*aprendizado inverso especializado*” (Psaltis et al., 1988; De Oliveira et al., 1991) que, algumas vezes, é confundido como sendo simplesmente “*modelagem inversa*”, como ocorre no trabalho de Jordan e Rumelhart, 1992), quando na verdade existem as duas estruturas principais para a modelagem inversa.

A estrutura de aprendizado inverso especializado é mostrada na Figura 3. 3(a). Nesta abordagem a RNA, como modelo inverso, precede o sistema e recebe como entrada um conjunto de treino que estende-se sobre o espaço de saída operacional desejada do sistema controlado, ou seja, corresponde a um sinal de comando ou sinal de referência. Esta estrutura também pode conter uma RNA trabalhando como o modelo direto do sistema, que fica colocada em paralelo com o sistema real, como mostrado na Figura 3. 3(b).

No caso da estrutura da Figura 3. 3(a), o sinal de erro para o algoritmo de aprendizado é a diferença entre o sinal de referência,  $r$ , e a saída do sistema,  $y$ . Entretanto, esta estrutura, quando utilizada com um algoritmo do tipo aprendizado supervisionado, apresenta um defeito terrível que é o fato de ajustar os pesos da RNA com um sinal de erro,  $e = r - y$ , que não está relacionado com o sinal de saída da rede  $u$ . Isto pode ser resolvido com a estrutura da Figura 3. 3(b), onde o sinal de ajuste dos pesos do modelo neural inverso é equivalente a um sinal  $e = u - u^*$ , conseguido com a "passagem" do sinal  $r - y_m$  através do modelo neural direto. Neste momento os pesos da RNA, que funciona como modelo direto do sistema, tem seus valores fixos. Estes pesos são ajustados em um etapa anterior, por exemplo através de uma estrutura igual a da Figura 3. 1.

Se o sistema real, por algum motivo, não pode ser utilizado com a estrutura da Figura 3. 3(b), pode-se utilizar a estrutura alternativa mostrada na Figura 3. 3(c), onde o sinal de erro que serve para ajustar os pesos do modelo neural inverso, é a diferença entre o sinal de comando,  $r$ , e a saída do modelo neural direto,  $y_m$ , que já foi treinado anteriormente e agora tem os seus pesos fixos.

Jordan (Jordan e Rumelhart, 1992) mostra que ao se utilizar a saída do sistema real, é possível produzir um modelo inverso exato, mesmo que se encontre um modelo direto inexato. Obviamente, isto não acontece quando se utiliza a estrutura da Figura 3. 3(c), porque neste caso o sinal de erro é propagado através do modelo neural direto, carregando consigo a inexatidão, que possa existir, nesta modelagem direta.



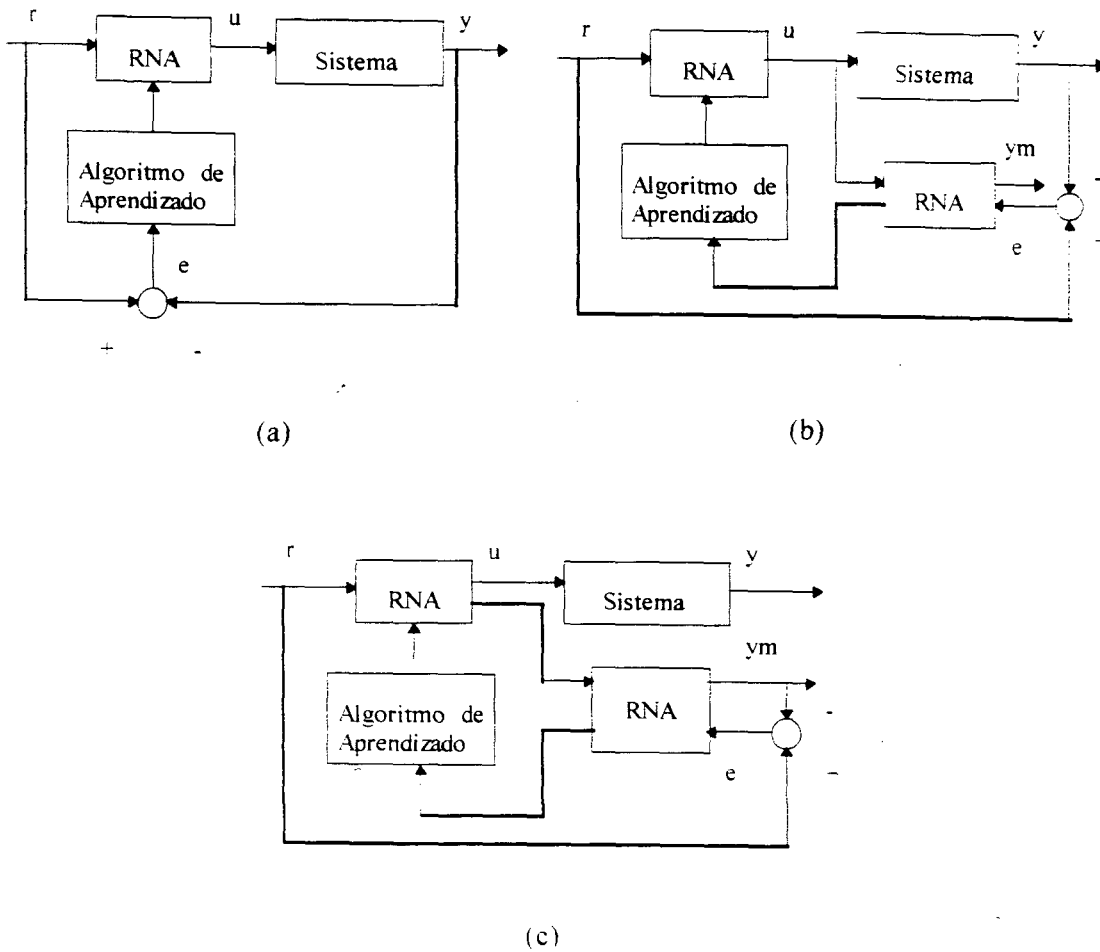


Figura 3. 3 - Estrutura para Modelagem Inversa

Em comparação com o aprendizado inverso generalizado, o aprendizado inverso especializado tem as seguintes características :

- O procedimento é direcionado a uma meta específica, uma vez que é baseado no erro entre a saída desejada do sistema,  $r$ , e a saída atual,  $y$  ou  $ym$ . Em outras palavras, o sistema recebe entradas durante a fase de treinamento que corresponde a entradas operacionais que irá receber subsequentemente.
- Nos casos onde o mapeamento direto do sistema não é único, um modelo inverso particular será encontrado (Jordan e Rumelhart, 1992).

Considerando a abordagem entrada-saída para a RNA que irá modelar o sistema inverso, tira-se da equação (3.17) que a função inversa  $f^{-1}(\cdot)$  leva a geração de  $u(t)$ , que requererá o conhecimento do valor futuro  $y(k+1)$ . Para superar este problema, substitui-se esta variável por  $r(k+1)$ , o qual é suposto ser disponível no instante de amostragem  $k+1$ . Isto é uma consideração razoável, uma vez que  $r$  é normalmente relacionado ao sinal de referência, que supõe-se ser conhecido um passo à frente.

Deste modo, a relação entrada-saída não-linear para a RNA modelando o inverso do sistema é;

$$u(k) = h[y(k), \dots, y(k-n+1), r(k+1), u(k-1), \dots, u(k-m+1)] \quad (3.24)$$

isto é, o modelo neural inverso recebe como entradas a saída atual e passadas do sistema, o sinal de referência (ou de treinamento) futuro e os valores passados da entrada do sistema. Nos casos onde é desejável treinar a rede como modelo inverso sem o sistema real, os valores de  $y$  da equação (3.24) são simplesmente substituídos pela saída da RNA que representa o modelo direto,  $y_m$ .

### 3.4 Características Importantes da Identificação Neural

É fácil aplicar RNA's a um conjunto de dados, sem conhecer muito bem a Teoria de Identificação de sistemas, e utilizar o que parecem ser resultados altamente confiáveis. Mas, como em outras formas de identificação ou aproximação, também é fácil ser enganado pelos resultados obtidos sem uma melhor avaliação teórica. Deste modo é importante estar plenamente consciente das propriedades das RNA's e dos algoritmos empregados na tarefa de identificar sistemas dinâmicos não-lineares. Sendo que estas propriedades podem ser utilizadas para formular métodos que validem os resultados obtidos na identificação.

As características dos modelos neurais que serão apresentadas aqui, são retiradas dos trabalhos de Billings (Billings et al., 1991; Billings et al., 1992; Billings e Chen, 1995).

### 3.4.1. Expansão da RNA

Da seção (2.2) tira-se que existem duas maneiras de representar a dinâmica em uma RNA, ambas podendo ser utilizadas na identificação de sistemas dinâmicos : uma consiste no uso de um conjunto de sinais atrasados no tempo na entrada da RNA, e outra consiste no uso de neurônios dinâmicos em uma das camadas intermediárias da RNA.

Analisando o primeiro caso, verifica-se que se o modelo de um sistema envolve uma expansão dos sinais  $u(k-3)$ ,  $y(k-9)$  e  $y(k-14)$ , a RNA só representará bem o sistema se utilizar estas três variáveis, caso contrário o modelo será uma representação pobre do sistema, independentemente do algoritmo de treinamento empregado. Devido a este fato, a escolha do conjunto de sinais de entrada da RNA é crucial para o seu desempenho (um bom estudo sobre este assunto pode ser encontrada no trabalho de De Azevedo (De Azevedo, 1993) e de Cruz (Cruz et al., 1998)).

O problema citado acima não é fácil de resolver, porque na realidade o retardo apropriado dos sinais  $u$ 's e  $y$ 's pode ser distribuído sobre um grande intervalo, e o procedimento de sobre-especificar os sinais de entrada da RNA leva à problemas de dimensionalidade, sobre-treinamento e aprendizado lento.

Este tipo de problema, permanece quando se utilizam neurônios dinâmicos na RNA, onde a escolha recai no número de estados da rede necessários para resultar em uma boa representação do sistema, já que em situações práticas do mundo real, não se conhece os estados do sistema dinâmico não-linear.

### 3.4.2. Validação do Modelo

Testes de validação do modelo são procedimentos para avaliar a inadequabilidade de um modelo, independentemente das discrepâncias encontradas nos modelos neurais, tais como o conjunto de sinais de entrada incorreto, o número de neurônios da camada intermediária insuficiente, os dados com ruído ou uma rede que não converge.

Os bem conhecidos testes de validação para modelos lineares (Ljung e Söderström, 1983) não são apropriados para modelos neurais. Sjöberg em seu trabalho (Sjöberg, 1995) mostra alguns métodos para validação de modelos neurais, como por exemplo o método chamado de 'Detectando Não-Linearidades'. Aqui será apresentado um conjunto de condições desenvolvido por Billings (Billings e Voon, 1983; Billings e Voon, 1986) adequado para a validação de modelos não-lineares, neurais ou não.

Se um modelo de um sistema é adequado, então os resíduos ou o erro de predição,  $\varepsilon(k) = y(k) - y_m(k)$ , deve ser não-previsível para todas as combinações lineares e não-lineares das entradas e saídas passadas (Billings et al., 1992). A determinação de testes simples, que possam detectar esta condição, é complexa, mas pode-se mostrar que as seguintes relações garantem esta condição (Billings e Voon, 1986) :

$$\begin{aligned}
 \Phi_{\varepsilon\varepsilon}(\tau) &= E\{\varepsilon(t-\tau)\varepsilon(\tau)\} = \delta(t) \\
 \Phi_{u\varepsilon}(\tau) &= E\{u(t-\tau)\varepsilon(\tau)\} = 0, \forall \tau \\
 \Phi_{u^2\varepsilon}(\tau) &= E\{[u^2(t-\tau) - \bar{u}^2]\varepsilon(\tau)\} = 0, \forall \tau \\
 \Phi_{u^2\varepsilon^2}(\tau) &= E\{[u^2(t-\tau) - \bar{u}^2]\varepsilon^2(\tau)\} = 0, \forall \tau \\
 \Phi_{\varepsilon(\varepsilon u)}(\tau) &= E\{\varepsilon(\tau)\varepsilon(t-1-\tau)u(t-1-\tau)\} = 0, \tau \geq 0
 \end{aligned} \tag{3.25}$$

onde  $\Phi_{ZY}(\tau)$  indica a função correlação cruzada entre as funções  $Z(\tau)$  e  $Y(\tau)$ ,  $\bar{u}^2$  representa o valor médio de  $u^2(\tau)$  e  $u^2(\tau) = u^2(\tau) - \bar{u}^2$ . Na prática, utiliza-se a

correlação normalizada e faz-se o cálculo da estimativa da função correlação amostrada entre duas sequências  $\Psi_1$  e  $\Psi_2$ , como sendo;

$$\hat{\Phi}_{\Psi_1\Psi_2}(\tau) = \frac{\sum_{k=1}^{N-\tau} \Psi_1(k)\Psi_2(k+\tau)}{\left[ \sum_{k=1}^N \Psi_1^2(k) \sum_{k=1}^N \Psi_2^2(k) \right]} \quad (3.26)$$

onde  $N$  é o número de amostras utilizadas. A normalização assegura que todas as funções correlações encontram-se no intervalo  $-1 \leq \hat{\Phi}_{\Psi_1\Psi_2}(\tau) \leq 1$  independente da amplitude dos sinais.

O modelo será adequado se todas as correlações satisfizerem a equação (3.25), sendo que a condição de igualdade a zero é substituída pela condição de permanecer em 95% do intervalo de confiança, definido por  $1.96/\sqrt{N}$ .

Embora seja difícil provar que para as RNA's, as quais apresentam fortes não-linearidades nos parâmetros, os testes da equação (3.25) irão detectar todas as possíveis deficiências do modelo, várias simulações têm mostrado o sucesso da aplicação destes testes (Billings e Chen, 1995, Billings et al., 1992; Suykens et al., 1995; Sbárbaro e Johansen, 1997; De Oliveira et al., 1997; Turner et al., 1995; Zhang e Morris, 1995).

Este método de validação de modelos serve também para sistemas lineares. Neste caso só é necessário a verificação da primeira condição de auto-correlação do erro de predição. Esta correlação exige uma igualdade com a função impulso, porque esta é a forma da auto-correlação do ruído branco. O ruído branco é o que se encontra na prática, de forma aditiva, na saída do processo.

O cálculo das correlações cruzadas pode ser aplicado tanto para o conjunto de dados utilizados na estimação dos parâmetros do modelo, como para um conjunto

de dados de teste que não tenha sido utilizado durante a estimação paramétrica. O resultado em ambos conjuntos é idêntico (Sjöberg, 1995).

### 3.4.3. Ruído e Polarização

Muitas das aplicações de RNA's na identificação de sistemas dinâmicos não-lineares ignora o fato de que o ruído irá sempre estar presente se os dados forem coletados de um sistema real. A menos que os efeitos do ruído sejam bem compreendidos e possam ser apropriadamente compensados, modelos incorretos ou polarizados serão encontrados nos procedimentos de modelagem e identificação.

A polarização<sup>2</sup> é um conceito da teoria de estimação bem conhecido e, muitas das vezes, o seu efeito indesejável pode ser eliminado somente com o uso e identificação de um modelo do ruído. Se o sistema a ser modelado é não-linear, não existe razão para assumir que o ruído irá ser puramente linear. O ruído no modelo irá, geralmente, envolver produtos cruzados das entradas com as saídas.

Muitas das vezes a polarização é difícil de detectar, porque mesmo que um modelo neural polarizado seja obtido, o mesmo irá apresentar uma boa predição para o conjunto de dados utilizados no treinamento. Isto já é esperado porque a RNA foi treinada para minimizar uma função quadrática do erro de predição, ajustando a curva aos dados. Mas isto não significa que a RNA forneça um modelo dos mecanismos básicos do sistema.

Dado um sistema,

$$y(k) = f(y(k-1), \dots, y(k-n), u(k-1), \dots, u(k-m)) + d(t) \quad (3.27)$$

onde  $d(t)$  é o ruído do sistema, uma maneira de evitar a polarização é estimar também o modelo do ruído. Uma RNA que pode acomodar esta condição foi

<sup>2</sup> Para obter maiores informações sobre o efeito e as causas da polarização na identificação paramétrica, o leitor pode ler Barreiros (1995), Coelho (1991), Ljung (1987) e Norton (1986).

proposta por Billings (Billings et al., 1992) e consiste no aumento da RNA, pela adição de um mapeamento linear dos termos do erro de predição atrasados  $\varepsilon(k-1)$ , ...,  $\varepsilon(k-p)$ , calculados na iteração anterior. Esta arquitetura de RNA é vista na figura a seguir.

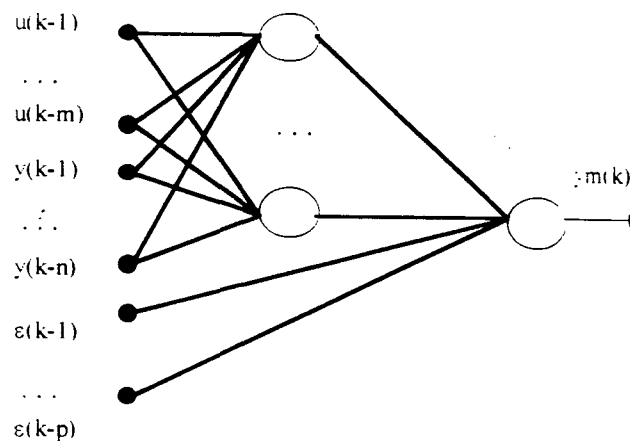


Figura 3. 4 - Rede Neural Artificial com um Modelo Linear do Ruído

Uma outra alternativa, mais geral, é simplesmente utilizar os sinais atrasados no tempo do erro de predição como novas entradas da rede que se conectam normalmente a camada intermediária de neurônios (Chen et al., 1990a). Esta arquitetura é mostrada a seguir.

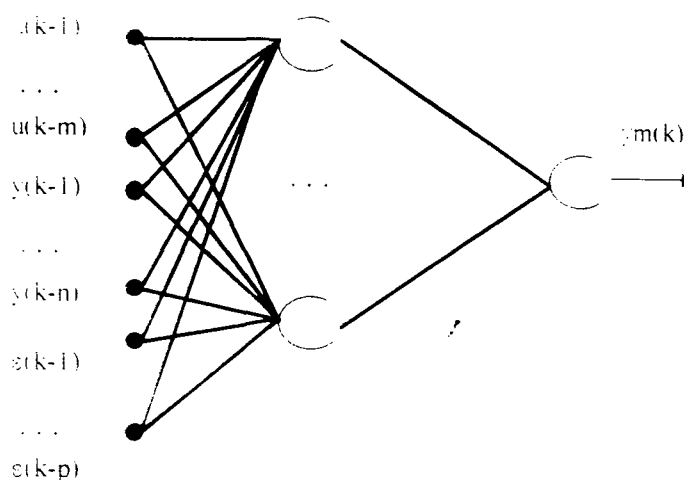


Figura 3. 5 - Rede Neural Artificial com um Modelo Não-Linear do Ruído

O sinal  $\varepsilon(t)$  é utilizado devido a impossibilidade, em aplicações práticas, de usar o sinal  $d(t)$ . Para RNA's com neurônios dinâmicos, não existem estudos sobre o uso do sinal de erro de predição, na tarefa de identificação de sistemas dinâmicos não-lineares, exceto o trabalho seminal de De Oliveira (De Oliveira et al., 1997a).

#### 3.4.4. Sobre-Parametrização da Rede Neural Artificial

Quando escolhe-se um número maior do que o necessário de elementos de sinais de entrada da RNA ou de neurônios da camada intermediária, é comum ocorrer a sobre-parametrização da RNA, que fica caracterizado quando a rede, durante a sessão de treino, para um dado conjunto de treino, deixa de aprender o que foi treinado em  $q$  iterações e começa a desaprender o que já tinha aprendido.

Isto acontece porque existem alguns parâmetros que são mais importantes que outros. No processo de aprendizado quando a função custo decresce significa que os parâmetros realmente necessários para a identificação têm maior influência que os desnecessários; quando a função custo começa a crescer, após um período de decrescimento máximo, isto significa que os parâmetros que são desnecessários começam a ter mais influência que os parâmetros necessários. Uma boa descrição do processo de sobre-parametrização é encontrado nos trabalhos de Sjöberg (Sjöberg e Ljung, 1992) e de Moody (Moody, 1992).

Na prática, para evitar a sobre-parametrização, utilizam-se modelos que vão crescendo em complexidade, onde a complexidade está relacionada com o número de neurônios, com o número de sinais de entrada (no caso da abordagem entrada-saída) e com o número de laços de realimentação (no caso das redes recorrentes) da RNA, e seleciona-se o modelo de menor complexidade que satisfaz os testes de validação do modelo. Duas das abordagens mais comuns para evitar a sobre-parametrização em RNA's são as chamadas abordagens de *punição* e de *regularização*. A primeira consiste na utilização de uma RNA bastante complexa, onde são feitas punições com a eliminação de pesos, até a redução máxima do tamanho da rede, sendo que existem inúmeras versões para este conceito de



punição (Von Zuben, 1996). A regularização aumenta a capacidade de generalização da RNA através da adição de uma função de penalidade extra na função custo da RNA a ser minimizada, de modo a penalizar a sobre-parametrização (Sjöberg, 1995).

A tarefa de encontrar uma RNA de menor complexidade, para um dado problema, também pode ser realizada através de métodos de computação evolucionária (Barreto, 1996; Roisenberg, 1998; Brasil, 1999; Dias, 1999).

### 3.5 Exemplo de Identificação Neural Não-Linear

Nesta seção, os exemplos simulados servem para mostrar a utilidade dos modelos de RNA com dinâmica interna na tarefa de identificar sistemas dinâmicos que apresentem ruído na saída. Os exemplos consideram ruído branco ou colorido de forma aditiva na saída do processo, ou ruído branco influenciando não-linearmente na saída do processo. O procedimento de identificação do modelo segue o padrão determinado na Figura 3. 1. Devido ao uso da RNA com dinâmica interna não é necessário o uso do conjunto de sinais atrasados no tempo na entrada da rede.

#### 3.5.1. Sistema Dinâmico Discreto Não-Linear

O sistema dinâmico não-linear é representado pelas equações.

$$\begin{aligned} x_1(k+1) &= -0.7x_2(k) + x_3(k) \\ x_2(k+1) &= \tanh[0.3x_1(k) + x_2(k) + (1 + 0.3x_2(k))u(k)] \\ x_3(k+1) &= \tanh[-0.8x_1(k) + 0.6x_2(k) + 0.2x_2(k)x_3(k)] \end{aligned} \quad (3.28)$$

$$y(k) = [x_1(k)]^2 + d(k) \quad (3.29)$$

onde  $k$  é o instante de amostragem.  $x(k)$  é o vetor de estados.  $u(k)$  é o sinal de entrada.  $y(k)$  é o sinal de saída e  $d(k)$  é a sequência de ruído branco, com média 0.0 e variância 0.001.

O conjunto de treino é formado por 50 amostras coletadas para um sinal de entrada do tipo senoidal, mostrado abaixo, onde  $NPT$  é o número de amostras.

$$u(k) = \text{sen}\left(\frac{2\pi k}{NPT}\right) \quad (3.30)$$

A relação entrada/saída para o sistema, levando-se em conta o sinal de entrada determinado pela equação (3.30) e o sinal de saída livre de ruído, é mostrado na figura abaixo, revelando uma forte relação não-linear. O sinal de saída  $y(k)$  está normalizado entre  $\pm 1$ .

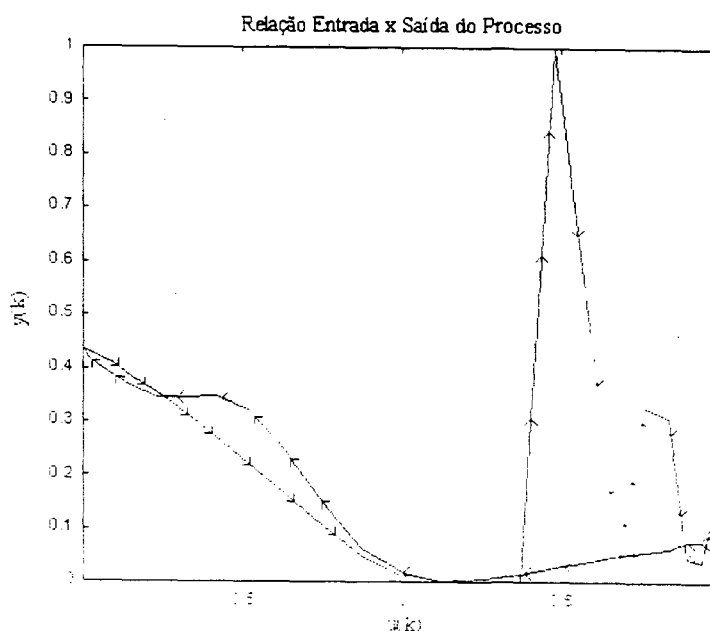


Figura 3.6 - Relação Entrada/Saída para o Sistema Dinâmico Não-Linear

### 3.5.2. Simulação 01

Nesta simulação a RNA utilizada é um modelo formado por 4 camadas de neurônios, com:

- Primeira camada : um único neurônio com função ativação identidade, significando um neurônio estático, e função de saída do tipo tangente hiperbólica.

- **Segunda camada** : seis neurônios com função ativação do tipo equação a diferenças de primeira ordem, onde a entrada do neurônio é afetada pelos estados dos neurônios vizinhos mais próximos conforme equação abaixo e função de saída do tipo tangente hiperbólica,

$$x_i(k+1) = a_{i-1i}x_{i-1}(k) + a_{ii}x_i(k) + a_{i+1i}x_{i+1}(k) + net_i(k) \quad (3.31)$$

que equivale a matriz A mostrada abaixo.

$$A = \begin{bmatrix} a_{11} & a_{12} & 0 & 0 & 0 & 0 \\ a_{21} & a_{22} & a_{23} & 0 & 0 & 0 \\ 0 & a_{32} & a_{33} & a_{34} & 0 & 0 \\ 0 & 0 & a_{43} & a_{44} & a_{45} & 0 \\ 0 & 0 & 0 & a_{54} & a_{55} & a_{56} \\ 0 & 0 & 0 & 0 & a_{65} & a_{66} \end{bmatrix} \quad (3.32)$$

- **Terceira camada** : doze neurônios com função ativação identidade, significando neurônios estáticos, e função de saída do tipo tangente hiperbólica.
- **Última camada** : um único neurônio com função ativação identidade, significando um neurônio estático, e função de saída do tipo tangente hiperbólica.

A identificação se deu em duas etapas : uma sessão de treinamento, onde a RNA teve os seus pesos ajustados em função dos dados coletados, e outra sessão onde foram calculadas as estimativas das funções correlações amostradas determinadas na equação (3.25). Nas duas etapas utiliza-se o sinal  $y(k)$  normalizado. O motivo da normalização é adequar numericamente os pares  $(u(k), y(k))$  para efeito de treinamento da rede neural (este artifício é bastante empregado nas aplicações de redes neurais em identificação e controle (De Azevedo, 1993; Filho et al., 1994)).

### 3.5.2.1. Sistema com Ruído Branco

A primeira simulação utiliza o sinal de ruído  $d(k)$ , da equação (3.29), como sendo igual a  $e(k)$ ,  $d(k) = e(k)$ , que é uma sequência ruído branco gaussiano de

média zero e variância 0.001. A variância é baixa devido as características de não-linearidade do sistema que amplifica bastante os sinais de alta-frequência, ou seja, uma variância maior geraria na saída do sistema apenas ruído.

A Figura 3. 7 mostra o comportamento da RNA no início do treinamento. Após 18350 iterações, interrompe-se a sessão de treinamento com um valor de Erro Médio Quadrático (EMQ), determinado pela Equação (2.69), igual 0.2061, como mostra a Figura 3.8 .

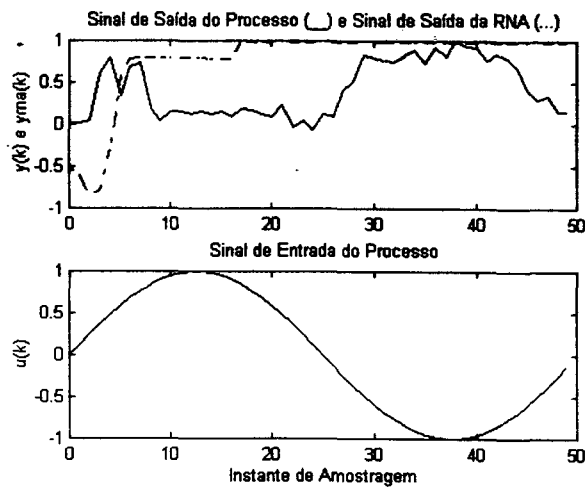


Figura 3. 7 - Comportamento do Modelo Neural no início do aprendizado

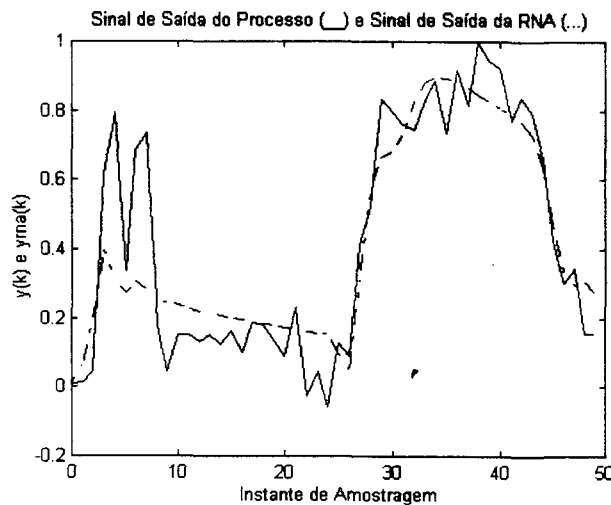


Figura 3. 8- Comportamento do Modelo Neural após fase de aprendizado

A evolução da função  $E(W)$ , que é o valor do EMQ em cada iteração de ajuste dos parâmetros do modelo neural, é ilustrada na figura a seguir. Convém lembrar que uma iteração indica a apresentação das  $NPT$  amostras para treinamento, e que cada amostra do treinamento proporciona um ajuste nos parâmetros da RNA.

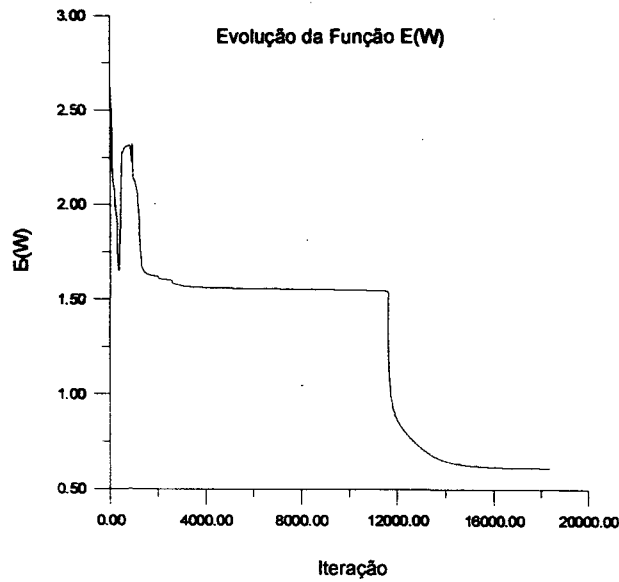


Figura 3. 9 - Evolução do erro médio quadrático durante o aprendizado

As estimativas das correlações cruzadas são mostradas a seguir.

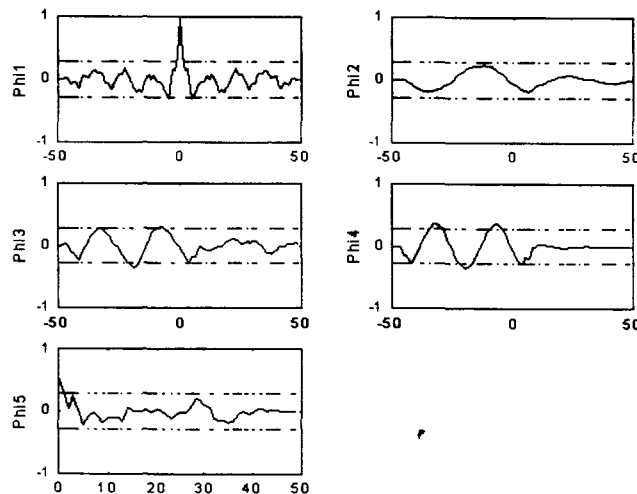


Figura 3. 10 - Estimativas das Funções Correlações Cruzadas :  $\text{Phi1}=\Phi_{\varepsilon\varepsilon}(\tau)$ ;  $\text{Phi2}=\Phi_{u\varepsilon}(\tau)$ ;  $\text{Phi3}=\Phi_{u^2\varepsilon}(\tau)$ ;  $\text{Phi4}=\Phi_{u^2\varepsilon^2}(\tau)$ ;  $\text{Phi5}=\Phi_{\varepsilon(\varepsilon u)}(\tau)$

Como o ruído é aditivo na saída do processo, o modelo neural não necessita ter informações sobre o mesmo. Ou seja, não é preciso o sinal de  $\varepsilon(k)$  na entrada da rede. A discrepância encontrada nas medidas de correlação não são importantes por que foram poucos pontos que ficaram fora da banda de confiança. Devido a sua estrutura particular, este comportamento da rede neural só foi conseguido com o uso de seis neurônios dinâmicos na segunda camada, três a mais do que o processo apresenta. Para um número menor de neurônios dinâmicos na segunda camada, a rede não conseguia acompanhar a saída do processo.

### 3.5.2.2. Utilizando Ruído Colorido

Nesta simulação utiliza-se o sinal de erro  $d(t)$ , da equação (3.29), como sendo um ruído colorido do tipo,  $d(k) = e(k) + 0.6e(k-1)$ , com  $e(k)$  sendo uma sequência ruído branco gaussiano de média zero e variância 0.001.

A Figura 3. 11 mostra o comportamento da RNA no início do treinamento. Após 154886 iterações, interrompe-se a sessão de treinamento com um valor de EMQ, determinado pela Equação (2.69), igual 0.1711, como mostra a Figura 3. 12.

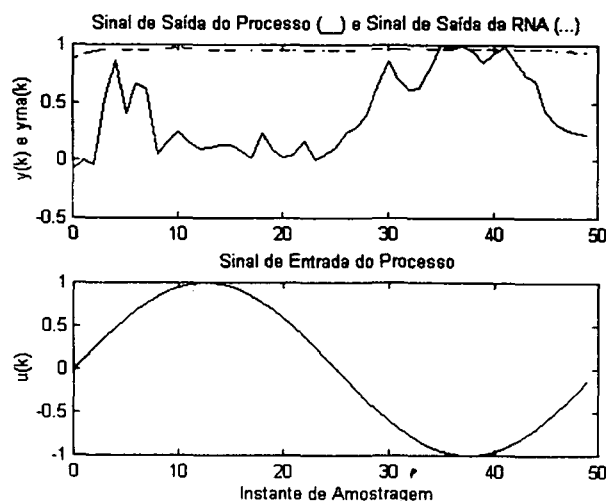


Figura 3. 11 - Comportamento do Modelo Neural no início do aprendizado

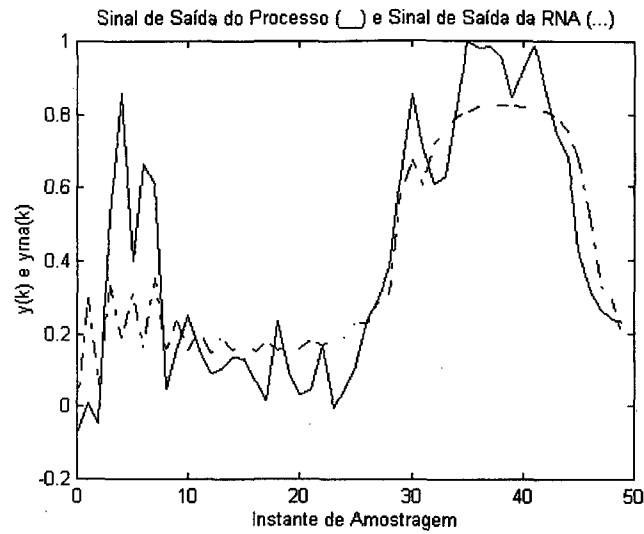


Figura 3. 12 - Comportamento do Modelo Neural após fase de aprendizado

A evolução da Função  $E(W)$ , é mostrado na Figura 3. 13.

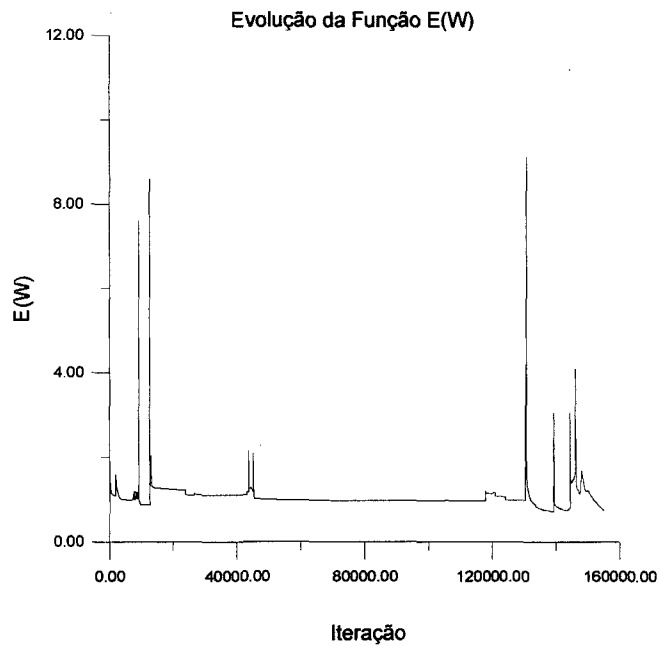


Figura 3. 13 - Evolução da Função  $E(W)$  durante o aprendizado

As estimativas das correlações cruzadas são mostradas na Figura 3. 14.

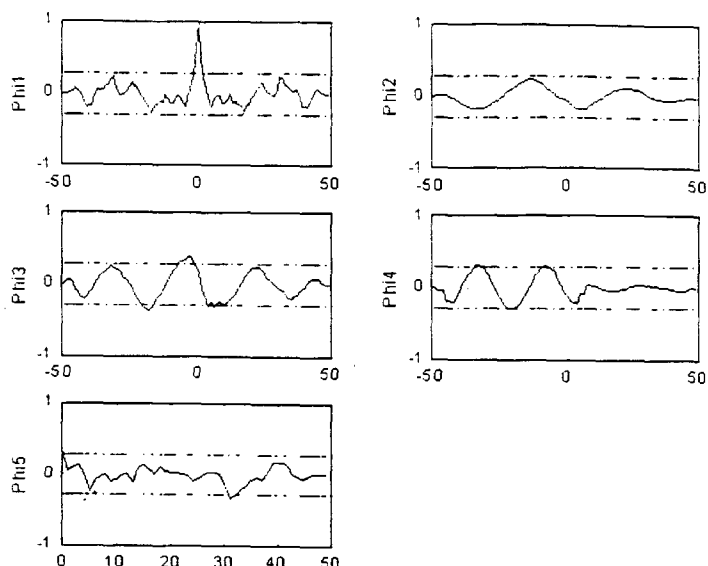


Figura 3. 14 - Estimativas das Funções Correlações Cruzadas : (a)  $\Phi_{\varepsilon\varepsilon}(\tau)$ ; (b)  $\Phi_{u\varepsilon}(\tau)$ ; (c)  $\Phi_{u^2\varepsilon}(\tau)$ ; (d)  $\Phi_{u^2\varepsilon^2}(\tau)$ ; (e)  $\Phi_{\varepsilon(\varepsilon u)}(\tau)$

Com o ruído colorido aditivo na saída, o modelo neural teve maiores dificuldades para seguir o sinal de saída do processo. Estas dificuldades são explicitadas através do número de iterações na sessão de treinamento e no comportamento oscilatório da função  $E(W)$ . Novamente devido às particularidades na estrutura do modelo neural foram necessários seis estados na rede, embora o processo só apresente três estados.

### 3.5.3. Simulação 02

Nesta simulação utiliza-se o sinal de erro de predição,  $\varepsilon(k)$ , como um outro sinal de entrada da rede. A RNA utilizada é um modelo formado por 4 camadas de neurônios, com :

- Primeira camada : dois neurônios, com função ativação identidade, significando neurônios estáticos, e função de saída do tipo tangente hiperbólica (um neurônio recebe o sinal  $u(k)$  e o outro neurônio recebe o sinal  $\varepsilon(k)$ ).



- Segunda camada : três neurônios com função ativação do tipo equação a diferenças de primeira ordem, onde a entrada do neurônio é afetada pelos estados de todos os outros neurônios conforme equação abaixo e função de saída do tipo tangente hiperbólica,

$$x_i(k+1) = a_{i1}x_1(k) + a_{i2}x_2(k) + \dots + a_{ii}x_i(k) + \dots + a_{in}x_n(k) + net_i(k) \quad (3.33)$$

que equivale a matriz  $A$  mostrada abaixo.

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \quad (3.34)$$

- Terceira camada : doze neurônios com função ativação identidade, significando neurônios estáticos, e função de saída do tipo tangente hiperbólica.
- Última camada : um único neurônio com função ativação identidade, significando um neurônio estático, e função de saída do tipo tangente hiperbólica.

A identificação se deu em duas etapas : uma sessão de treinamento, onde a RNA teve os seus pesos ajustados em função dos dados coletados, e outra sessão onde foram calculadas as estimativas das funções correlações amostradas determinadas na equação (3.25).

Nesta simulação, o sinal de erro  $d(t)$ , da equação (3.29) é um ruído colorido do tipo,  $d(k) = e(k) + 0.6e(k-1)$ , com  $e(k)$  sendo uma sequência ruído branco gaussiano de média zero e variância 0.001.

A Figura 3. 15 mostra o comportamento da RNA no início do treinamento. Após 1000 iterações, interrompe-se a sessão de treinamento com um valor de EMQ, determinado pela Equação (2.69), igual 0.1609, como mostra a Figura 3. 16.

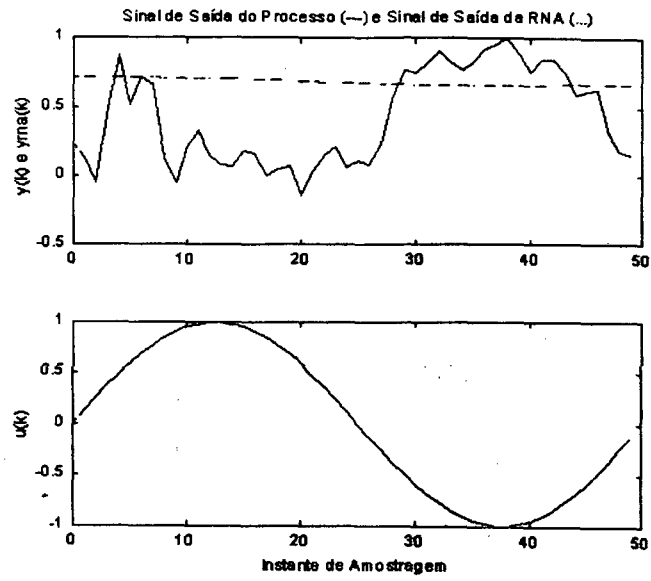


Figura 3. 15 - Comportamento do Modelo Neural no início do aprendizado

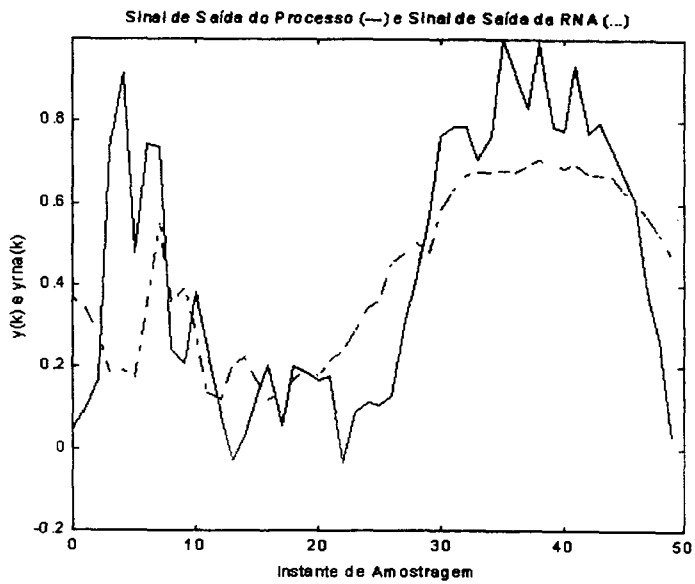


Figura 3. 16 - Comportamento do Modelo Neural após fase de aprendizado

A evolução da Função  $E(W)$ , é mostrado na Figura 3. 17.

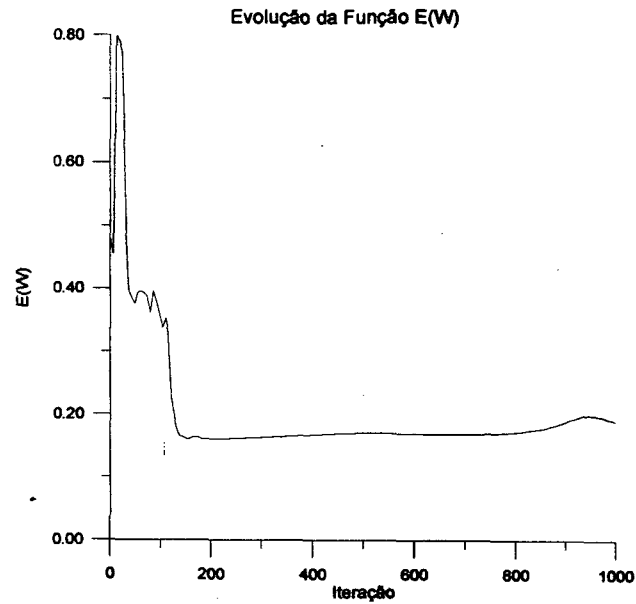


Figura 3. 17 - Evolução da Função E(W) durante o aprendizado

A estimativa das correlações cruzadas são mostradas na Figura 3. 18.

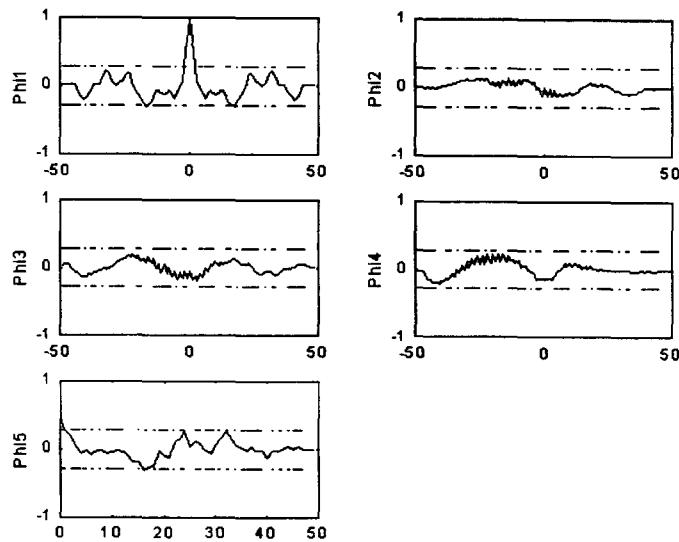


Figura 3. 18 - Estimativas das Funções Correlações Cruzadas : (a)  $\Phi_{\varepsilon\varepsilon}(\tau)$ ; (b)  $\Phi_{u\varepsilon}(\tau)$ ; (c)  $\Phi_{u^2\varepsilon}(\tau)$ ; (d)  $\Phi_{u^2\varepsilon^2}(\tau)$ ; (e)  $\Phi_{\varepsilon(au)}(\tau)$

A utilização do sinal  $\varepsilon(k)$  na entrada da rede permitiu o uso de apenas três neurônios dinâmicos na segunda camada do modelo neural. Aqui as particularidades do modelo neural influenciaram na obtenção das medidas de correlação, que só

foram conseguidas com o uso de uma matriz  $A$  cheia. Ou seja, a identificação é realizada com uma parametrização total<sup>3</sup>, através de nove parâmetros  $a_{ij}$ .

### 3.5.4. Simulação 03

Nesta simulação a equação de saída do processo é:

$$y(k) = [x_1(k) + d(k)]^2 \quad (3.35)$$

onde a influência do ruído em  $y(k)$  é não-linear. O modelo neural tem na sua entrada apenas o sinal  $u(k)$ . A RNA utilizada é um modelo formado por 4 camadas de neurônios, com :

- Primeira camada : um único neurônio com função ativação identidade, significando um neurônio estático, e função de saída do tipo tangente hiperbólica.
- Segunda camada : seis neurônios com função ativação do tipo equação a diferenças de primeira ordem, onde a entrada do neurônio é afetada pelos estados dos neurônios vizinhos mais próximos conforme equação (3.31) e a equação (3.32), e função de saída do tipo tangente hiperbólica,
- Terceira camada : doze neurônios com função ativação identidade, significando neurônios estáticos, e função de saída do tipo tangente hiperbólica.
- Última camada : um único neurônio com função ativação identidade, significando um neurônio estático, e função de saída do tipo tangente hiperbólica.

A identificação se deu em duas etapas : uma sessão de treinamento, onde a RNA teve os seus pesos ajustados em função dos dados coletados, e outra sessão onde foram calculadas as estimativas das funções correlações amostradas determinadas na equação (3.25).

---

<sup>3</sup> Para obter maiores detalhes sobre identificação de sistemas dinâmicos com parametrização total, o leitor pode ler o trabalho de McKelvey (McKelvey, 1995).

A terceira simulação feita, utiliza o sinal de erro  $d(t)$ , da equação (3.29), como sendo um ruído do tipo,  $d(k) = e(k)$ , com  $e(k)$  sendo uma sequência ruído branco gaussiano de média zero e variância 0.001.

A Figura 3. 19 mostra o comportamento da RNA no início do treinamento. Após 246 iterações, interrompe-se a sessão de treinamento com um valor de EMQ, determinado pela Equação (2.69), igual 0.156856, como mostra a Figura 3. 20.

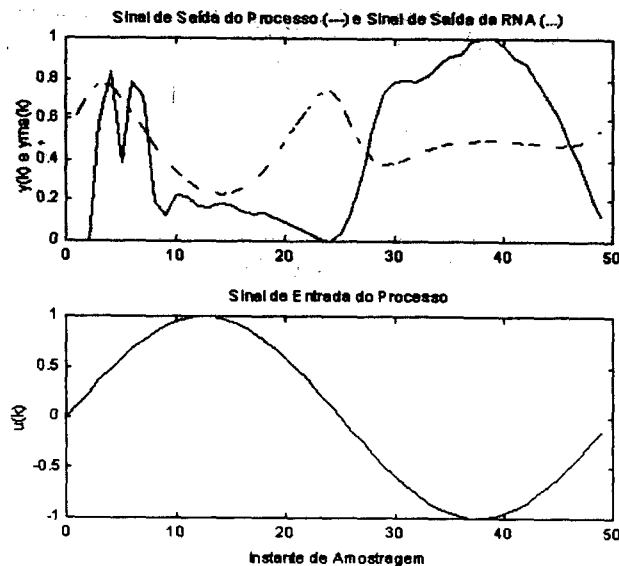


Figura 3. 19 - Comportamento do Modelo Neural no início do aprendizado

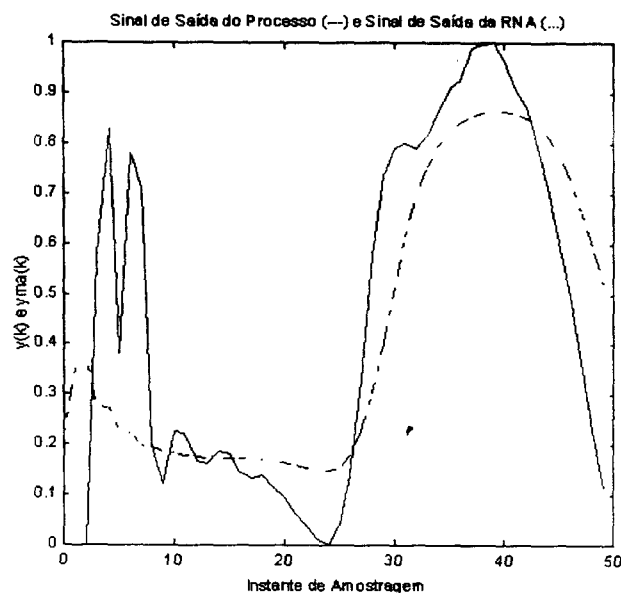


Figura 3. 20 - Comportamento do Modelo Neural após fase de aprendizado

A evolução da Função E(W), é mostrado na Figura 3. 21.

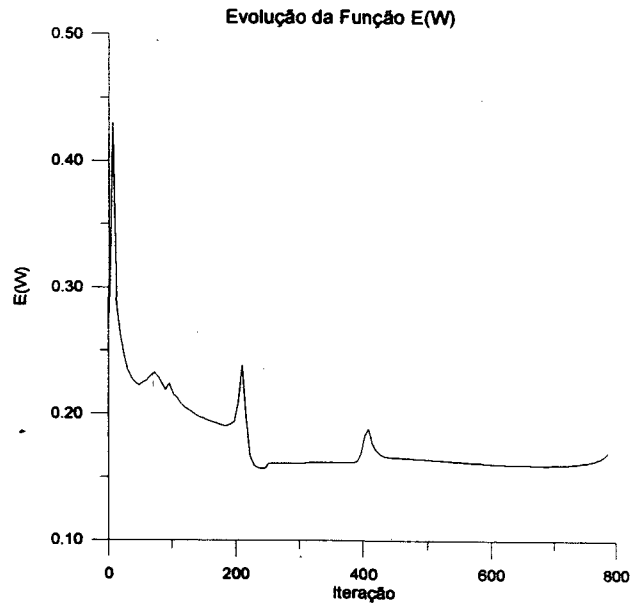


Figura 3. 21 - Evolução da Função E(W) durante o aprendizado

As estimativas das correlações cruzadas são mostradas na Figura 3. 22.

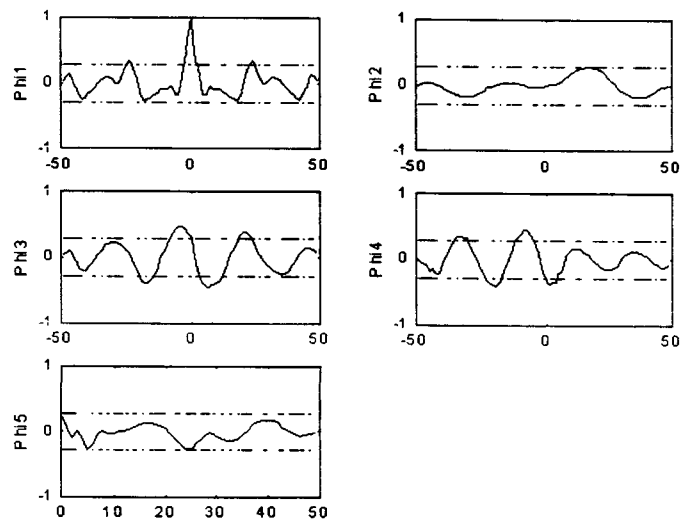


Figura 3. 22 - Estimativas das Funções Correlações Cruzadas : (a)  $\Phi_{\varepsilon\varepsilon}(\tau)$ ; (b)  $\Phi_{u\varepsilon}(\tau)$ ; (c)  $\Phi_{u^2\varepsilon}(\tau)$ ; (d)  $\Phi_{u^2\varepsilon^2}(\tau)$ ; (e)  $\Phi_{\varepsilon(\varepsilon u)}(\tau)$

Neste exemplo, as condições de não-linearidade do ruído de medida e de ausência do sinal  $\varepsilon(k)$  na entrada da rede, permitiu que o modelo neural

apresentasse um desempenho apenas aceitável, representado pelas funções de correlação cruzada com algumas falhas em  $\Phi_3$  e  $\Phi_4$ , para a identificação neural.

Esta falhas não chegam a invalidar o modelo porque são em pequeno número e não indicam uma tendência das funções em permanecerem fora do intervalo de confiança, mas mostram alguma deficiência no mesmo, quando comparado com as outras simulações feitas anteriormente para o mesmo processo dinâmico. Mesmo quando foi utilizado uma parametrização total na matriz  $A$ , a rede continuou a ter um desempenho apenas aceitável.

### 3.5.5. Simulação 04

Nesta simulação a equação de saída do processo é determinada pela equação (3.35) com a influência não-linear do ruído na medida de saída do processo. O modelo neural tem na sua entrada o sinal  $u(k)$  e o sinal  $\varepsilon(k)$ . A RNA utilizada é um modelo formado por 4 camadas de neurônios, com :

- Primeira camada : dois neurônios com função ativação identidade, significando neurônios estáticos, e função de saída do tipo tangente hiperbólica (um neurônio recebe o sinal  $u(k)$  e o outro neurônio recebe o sinal  $\varepsilon(k)$ ).
- Segunda camada : três neurônios com função ativação do tipo equação a diferenças de primeira ordem, onde a entrada do neurônio é afetada pelos estados de todos os outros neurônios conforme equação (3.33) e equação (3.34), e função de saída do tipo tangente hiperbólica.
- Terceira camada : doze neurônios com função ativação identidade, significando neurônios estáticos, e função de saída do tipo tangente hiperbólica.
- Última camada : um único neurônio com função ativação identidade, significando um neurônio estático, e função de saída do tipo tangente hiperbólica.

A identificação se deu em duas etapas : uma sessão de treinamento, onde a RNA teve os seus pesos ajustados em função dos dados coletados, e outra sessão onde foi calculada a estimativa das funções correlações amostradas determinadas na equação (3.25).

Aqui utiliza-se o sinal de erro  $d(t)$ , da equação (3.29), como sendo um ruído do tipo,  $d(k) = e(k)$ , com  $e(k)$  sendo uma sequência ruído branco gaussiano de média zero e variância 0.001.

A Figura 3. 23 mostra o comportamento da RNA no início do treinamento. Após 259 iterações, interrompe-se a sessão de treinamento com um valor de EMQ, determinado pela Equação (2.69), igual 0.144622, como mostra a Figura 3. 24.

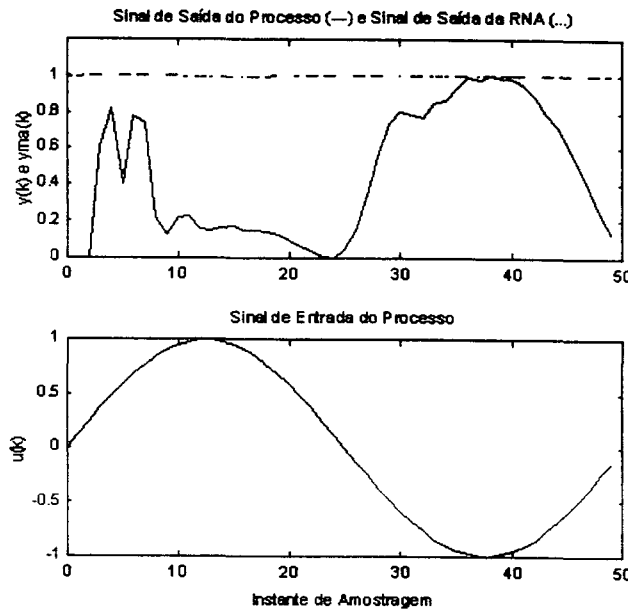


Figura 3. 23 - Comportamento do Modelo Neural no início do aprendizado



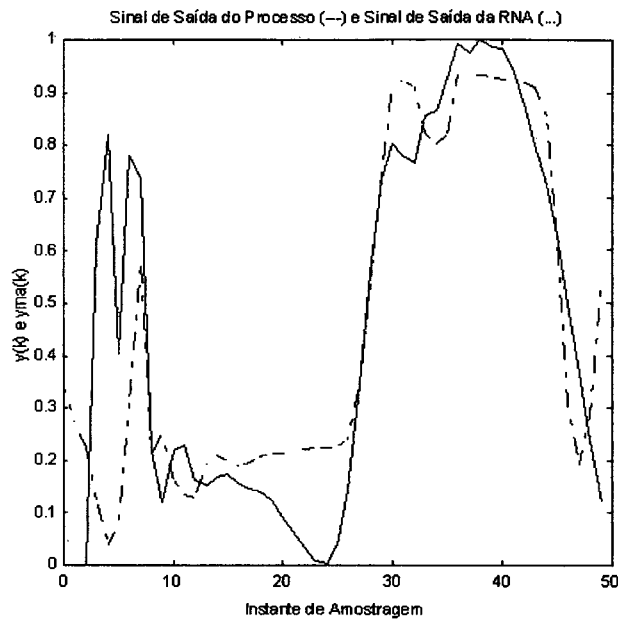


Figura 3. 24 - Comportamento do Modelo Neural após fase de aprendizado

A evolução da Função  $E(W)$ , é mostrado na Figura 3. 25.

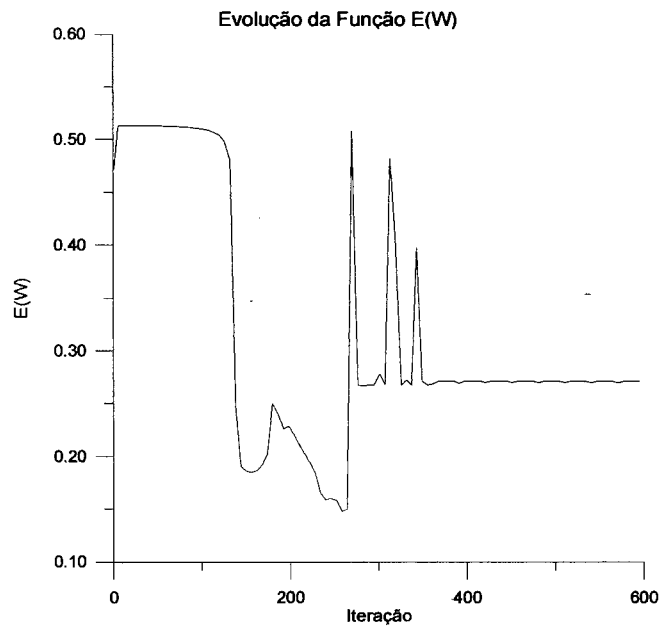


Figura 3. 25 - Evolução da Função  $E(W)$  durante o aprendizado

As estimativas das correlações cruzadas são mostradas na Figura 3. 26.

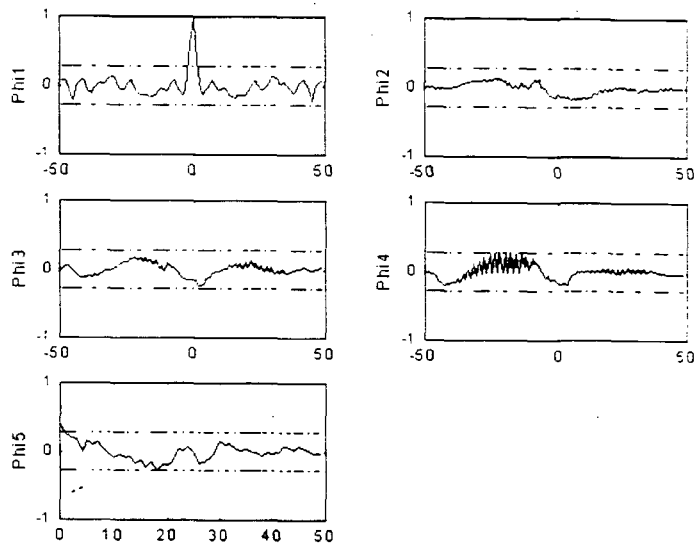


Figura 3. 26 - Estimativas das Funções Correlações Cruzadas : (a)  $\text{Phi1}=\Phi_{\varepsilon\varepsilon}(\tau)$ ; (b)  $\text{Phi2}=\Phi_{u\varepsilon}(\tau)$ ; (c)  $\text{Phi3}=\Phi_{u^2\varepsilon}(\tau)$ ; (d)  $\text{Phi4}=\Phi_{u^2\varepsilon^2}(\tau)$ ; (e)  $\text{Phi5}=\Phi_{\varepsilon(au)}(\tau)$

Com a inclusão do sinal  $\varepsilon(k)$  na entrada da rede, pode-se utilizar apenas três neurônios dinâmicos na segunda camada, com parametrização total, e conseguir que o modelo neural tenha um excelente desempenho na identificação do processo dinâmico, como visto na Figura 3.26.

### 3.6 Discussão

Neste capítulo, o modelo neural apresentado na seção (2.3) é utilizado para a identificação de um processo dinâmico não-linear. O modelo neural apresentou um desempenho aceitável. Como o modelo neural apresenta neurônios dinâmicos lineares, o mesmo tem dificuldade em tratar as não-linearidades da equação (3.28). Esta dificuldade é, em parte, compensada com a utilização adicional de neurônios dinâmicos e de parâmetros na matriz  $A$ .

Dos resultados mostrados na seção (3.5) observa-se que :

- com o ruído aditivo na saída do processo a identificação neural se comporta da mesma maneira, independente do ruído ser branco ou colorido. O ruído colorido apenas torna a tarefa de identificação mais difícil, representado pelo maior

número de iterações da sessão de aprendizado (treinamento com nove vezes mais iterações do que o treinamento com o ruído branco) e pelo comportamento 'nervoso' da função  $E(W)$ . Sem o uso do sinal de erro de predição na entrada da rede neural, a mesma necessita de seis estados para realizar a identificação, o dobro do número de estados do processo (para ambos os tipos de ruído).

- com o ruído de medida aditivo e o uso do sinal de erro de predição na entrada da rede neural, a mesma necessita apenas de três neurônios dinâmicos para identificar o processo dinâmico de terceira ordem. Apresentou-se apenas o caso com ruído de medida colorido por ser o mais complicado, como dito no item anterior. Neste caso a rede só apresentou um desempenho aceitável quando utilizou-se a matriz  $A$  cheia.
- com o ruído de medida influenciando de forma não-linear no sinal de saída do processo, o modelo neural só conseguiu ter um bom desempenho com a inclusão do sinal de erro de predição na entrada da rede. Neste caso, também só foram necessários três neurônios dinâmicos e a matriz  $A$  cheia. Sem o erro de predição na entrada da rede neural, a mesma não consegue ter o mesmo desempenho para identificar o processo dinâmico não-linear, independente do aumento do número de neurônios dinâmicos.

Como comentário final deste capítulo, pode-se dizer que o modelo neural com dinâmica interna proposto no Capítulo 2, a despeito de suas particularidades, consegue identificar processos dinâmicos não-lineares. Como na imensa maioria dos resultados de identificação neural existente na literatura, o modelo neural obtido tem uma pequena aproximação com as relações das variáveis de estado do processo identificado. Esta é uma diferença básica entre as abordagens de identificação não-linear neural e as que não utilizam redes neurais. As técnicas de identificação não-linear que não utilizam redes neurais sempre procuram reproduzir as relações existentes no processo identificado, já as técnicas de identificação neural não se preocupam com esta reprodução. A situação em que o modelo neural apresenta um número de estados maior do que o processo pode ser explicado pelo uso de

---

hiperestados, no modelo neural, que é mapeado para os estados do processo (Zbikowski e Gawthrop, 1995; Zbikowski, 1994; Tzirkel-Hancock, 1992). É conveniente lembrar que o algoritmo de treinamento não estima estados e sim parâmetros.

No próximo capítulo, se restringe a classe de processos não-lineares que são identificados por esta rede neural, conseguindo assim, um modelo que apresenta relações entre variáveis o mais próximo possível das existentes entre os estados do processo identificado.

## 4 - Sistemas Dinâmicos com Não-Linearidades Estáticas e com Dinâmica Linear

### 4.1 Introdução

A complexidade e diversidade dos sistemas não-lineares podem ser consideradas como as razões para a não existência de uma teoria geral de identificação não-linear (Sjöberg, 1995; Sandberg, 1982). Devido a este fato é importante estudar e desenvolver métodos para identificar classes de sistemas não-lineares.

Existem basicamente duas linhas mestras no desenvolvimento de métodos para identificar processos não-lineares. A primeira delas é o uso de uma descrição geral para representar o processo não-linear. Típicas representações não-lineares gerais são as séries de Volterra e expansão de Wiener (Schetzen, 1980), além das RNA's feedforward multicamadas (Narendra e Parthasarathy, 1990; Levin e Narendra, 1991; Chen e Billings, 1992; Feng, 1993; Saxén e Saxén, 1994; Sjöberg, 1995; Von Zuben, 1996; Ruiz-Vargas, 1997). O uso destes modelos gerais na prática envolve a decisão de quais termos da série de Volterra e da expansão de Wiener serão utilizados, ou então qual a topologia da rede neural utilizada, para uma dada aplicação. Não existem receitas para guiar esta seleção da estrutura do modelo não-linear quando há ausência de conhecimento sobre o processo identificado (Billings e Chen, 1995; Hwang e Shyu, 1988). Um problema adicional na estimação de modelos gerais é o surgimento de problemas numéricos para processos de ordem alta.

A segunda linha é considerar classes especiais de modelos não-lineares, tais como sistemas bilineares (Beghelli e Giudorzi, 1976; Dunoyer et al., 1997), sistemas affine (Liu e Celikovský, 1997; Tunay e Kaynak, 1995). Uma das classes de sistemas não-lineares mais estudadas é aquela que considera modelos não-lineares formados pela combinação de blocos não-lineares estáticos e blocos dinâmicos

lineares (Billings, 1980; Billings e Fakhouri, 1982; Norton, 1986; Åström e Wittenmark, 1995). Exemplos destes modelos são : modelo de Wiener (Greblicki, 1992; Sbárbaro e Johansen, 1997; Wigren, 1994), modelo de Hammerstein (Stoica e Söderström, 1982; Gallman, 1976; Greblicki e Pawlak, 1987; Åström e Wittenmark, 1995), modelo de Wiener-Hammerstein (Haber e Unbehauen, 1990; Billings e Fakhouri, 1978) e modelo de Hammerstein-Wiener (Hunter e Korenberg, 1986; Bai, 1998), como observados na figura a seguir.

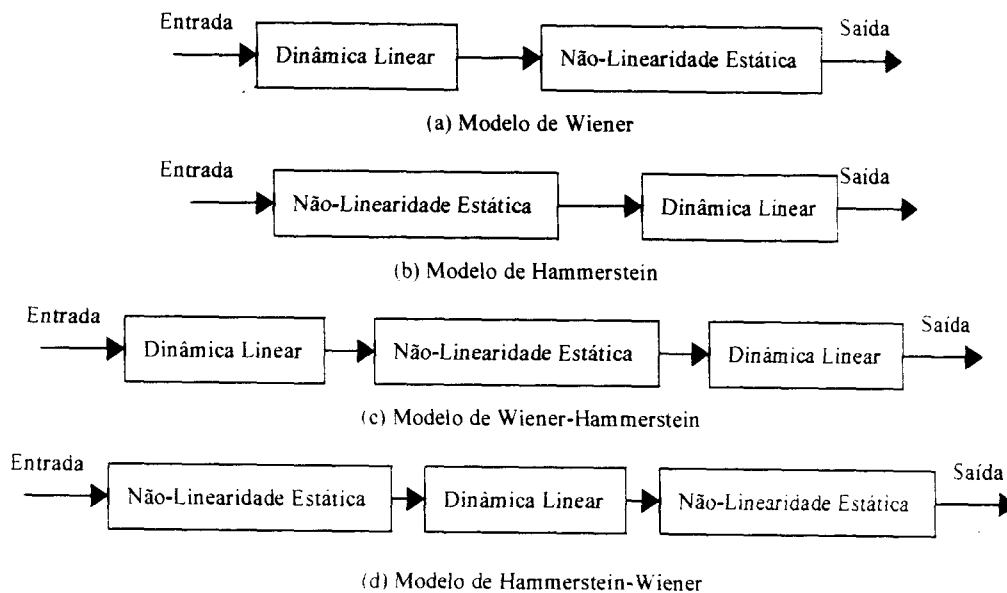


Figura 4. 1 - Modelos Não-Lineares orientados à blocos, com blocos de dinâmica linear e com blocos não-lineares estáticos

Nestes modelos apenas os sinais de entrada e de saída são disponíveis. Todos os outros sinais intermediários não são disponíveis. Estes modelos são do tipo caixa-cinza porque existe o conhecimento prévio da sua estrutura.

Estes tipos de modelos podem servir para aproximar processos que são encontrados em algumas aplicações importantes, tais como : controle de pH (Pajunen, 1994), fluxo de fluido (Singh et al. 1980; Zhang e Bai, 1996), identificação de sistemas biológicos (Hunter e Korenberg, 1986), identificação de sistemas lineares com sensores não-lineares (Doebelin, 1983), processamento de imagem (Swachuk e Strand, 1982), modelo para cancelamento de ruído (Stapleton e Bass,

1985), modelo para sistemas visuais biológicos (Brinker, 1989) e descrição do processo de uma coluna de destilação (Bars et al., 1990).

Estes modelos orientados à blocos são identificados por métodos paramétricos e não-paramétricos. Os métodos paramétricos tratam a não-linearidade como sendo do tipo polinomial finita, de ordem conhecida e o sistema com uma única entrada. Neste caso, o algoritmo de identificação das partes lineares e não-lineares são dependentes computacionalmente, ou seja, os parâmetros de uma parte são considerados constantes enquanto os parâmetros da outra parte são determinados (Billings e Fakhouri, 1978; Hwang e Shyu, 1988; Gallman, 1976; Åström e Wittenmark, 1995).

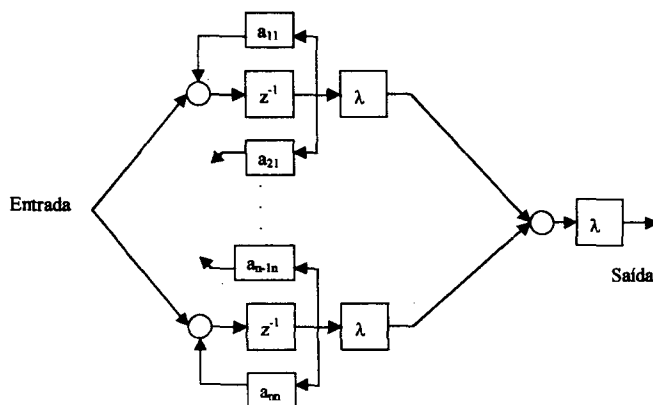
Os métodos não-paramétricos são baseados em algoritmos de estimação de funções de regressão, retirados da literatura de estatística (Rao, 1983), para identificar a seção não-linear e para a seção linear utiliza-se a identificação da função impulso. Outra abordagem não-paramétrica é o uso da série de Volterra (Billings, 1980; Billings e Fakhouri, 1979; Haber e Unbehauen, 1990). Nestes métodos não existem restrições quanto ao tipo de função não-linear e podem ser utilizados para sistemas com várias entradas (Greblicki e Pawlak, 1987; Greblicki, 1992). Na identificação não-paramétrica a maior dificuldade é a escolha dos *kernel's* da função regressão ou da série de Volterra.

Embora estes métodos de identificação, paramétricos ou não-paramétricos, tenham sido desenvolvidos para sistemas orientados à blocos, os mesmos não são aplicados indistintamente para os quatro tipos de modelos da Figura 4.1. Por exemplo, os procedimentos de identificação desenvolvidos para o modelo de Wiener-Hammerstein (Billings e Fakhouri, 1978), modelo de Hammerstein (Rangan et al., 1995) e modelo de Wiener (Sbárbaro e Johansen, 1997) não se aplicam aos modelos de Hammerstein-Wiener, devido a existência de duas não-linearidades. Já o método desenvolvido para o modelo de Hammerstein só se aplica ao modelo de Wiener se a não-linearidade deste for inversível (Greblicki, 1992).

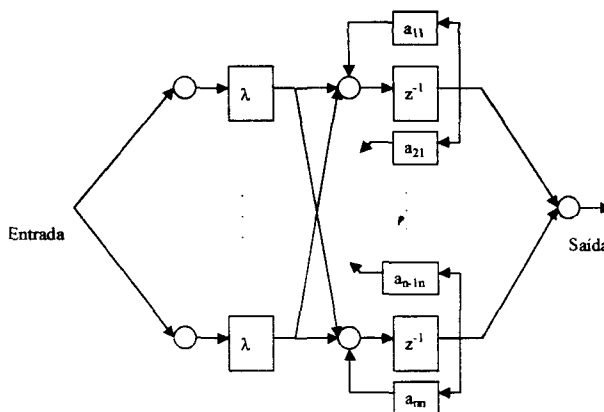
Na tentativa de superar alguns destes problemas e apresentar um método de identificação unificado para os quatro modelos não-lineares apresentados na Figura 4. 1, a próxima seção mostra o modelo neural com dinâmica interna, da seção (2.3), como uma alternativa para a modelagem de processos que contenham linearidade dinâmica e não-linearidade estática. Esta modelagem apresentada serve apenas para modelos discretos, devido a RNA desenvolvida ser do tipo discreta.

### 4.2 Identificação Neural de Sistemas Orientados à Blocos

Por conter neurônios com dinâmica linear e neurônios com não-linearidade estática, o modelo neural apresentado na seção (2.3) está completamente adequado para implementar modelos de processos que contenham blocos dinâmicos lineares e blocos não-lineares estáticos. A figura a seguir mostra como a rede neural implementa os quatro modelos da Figura 4. 1.

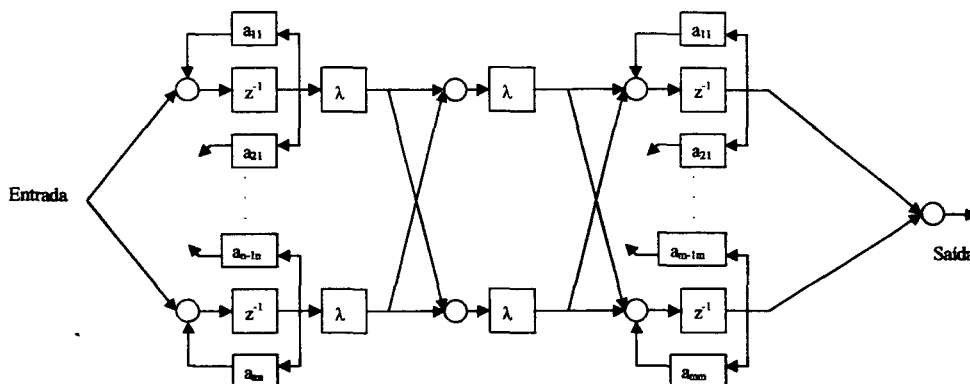


(a) RNA com dinâmica interna implementando um modelo de Wiener

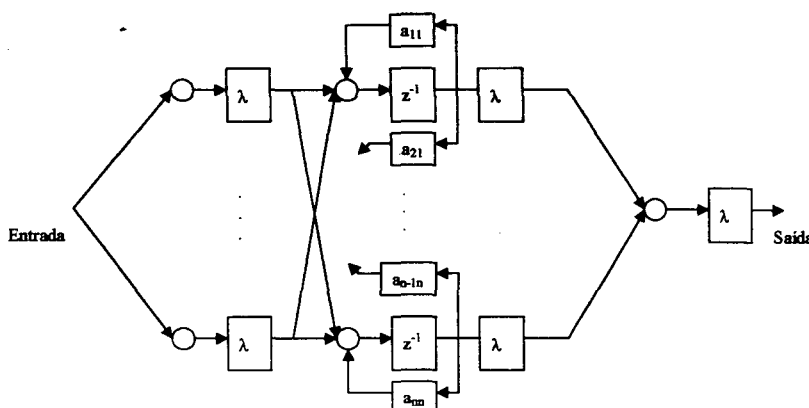


(b) RNA com dinâmica interna implementando um modelo de Hammerstein





(c) RNA com dinâmica interna implementando um modelo de Wiener-Hammerstein



(d) RNA com dinâmica interna implementando um modelo de Hammerstein-Wiener

Figura 4. 2 - RNA feedforward multicamadas discreta com dinâmica interna implementando modelos dinâmicos não-lineares orientados à blocos

Existem na literatura trabalhos que utilizam RNA's na identificação de processos orientados à blocos (Al-Duwaish et al., 1996; Abdulaziz e Farsi, 1993). Entretanto nestes trabalhos a RNA é do tipo multicamadas feedforward com dinâmica externa, e serve para identificar apenas a parte não-linear estática do processo. A parte dinâmica linear é identificada através de um modelo ARMA com o uso do algoritmo dos mínimos quadrados recursivo. O primeiro trabalho em que utiliza-se RNA's com dinâmica interna na identificação de processos orientados à blocos foi publicado por De Oliveira e co-autores (De Oliveira et al., 1997b).

O uso da rede neural com dinâmica interna na implementação de modelos para processos orientados à blocos tem as seguintes características :

- método de identificação unificado para modelos de Wiener, Hammerstein, Wiener-Hammerstein e Hammerstein-Wiener;
- método de identificação que devido a topologia das RNA's serve para identificar sistemas SISO e MIMO;
- a partir da capacidade de aproximação não-linear das RNA's, não existe restrições quanto ao tipo de não-linearidade estática presente no processo;
- permite a identificação simultânea dos blocos dinâmicos lineares e dos blocos não-lineares estáticos.

A aplicabilidade da identificação neural para esta classe de processos não-lineares é ilustrada através de vários exemplos simulados, que são apresentados na próxima seção.

### 4.3 Exemplos Simulados

#### 4.3.1 Modelo de Hammerstein-Wiener

Nesta seção é utilizada a rede neural para identificar, segundo a descrição da Figura (3.1), um processo não-linear do tipo Hammerstein-Wiener de segunda ordem. O processo é descrito pelas equações,

$$\begin{aligned} x_1(k+1) &= -0.7x_1(k) + 0.1 \tanh[u(k)] \\ x_2(k+1) &= -0.6x_2(k) + 0.2 \tanh[u(k)] \end{aligned} \quad (4.1)$$

$$y(k) = [x_1(k)]^2 + d(k) \quad (4.2)$$

onde  $k$  é o instante de amostragem,  $x(k)$  é o vetor de estados,  $u(k)$  é o sinal de entrada,  $y(k)$  é o sinal de saída e  $d(k)$  é a sequência de ruído aditivo na saída do processo. O ruído é simulado por uma sequência pseudo-aleatória de distribuição normal com média 0.0 e variância 0.0003. Este valor de variância foi escolhido para que o ruído não deformasse completamente o sinal de saída.

O conjunto de treino é formado por 200 amostras coletadas para um sinal de entrada do tipo senoidal como mostrado na equação,

$$u(k) = \begin{cases} 2 \left[ \text{sen} \left( \frac{2 k \pi}{NPT / 4} \right) \right] + 0.2, & k < 100 \\ 2 \left[ \text{sen} \left( \frac{2 k \pi}{NPT / 2} \right) \right] + 0.2, & k \geq 100 \end{cases} \quad (4.3)$$

onde  $NPT$  é o número de pontos de treino, ou número de amostras.

A RNA utilizada é formada por 4 camadas de neurônios, com todas as camadas contendo um *bias*, exceto na primeira camada, onde :

- A primeira camada apresenta um único neurônio com função ativação identidade, significando um neurônio estático, e função de saída do tipo tangente hiperbólica.
- A segunda camada tem dois neurônios com função ativação do tipo equação a diferenças de primeira ordem, onde a entrada do neurônio é afetada pelo estado do próprio neurônio, conforme equação (4.4), e função de saída do tipo tangente hiperbólica. Neste caso a matriz  $A$  é diagonal, como mostra a equação (4.5).

$$x_i(k+1) = a_{ii}x_i(k) + net_i(k) \quad (4.4)$$

$$A = \begin{bmatrix} a_{11} & 0 \\ 0 & a_{22} \end{bmatrix} \quad (4.5)$$

- A terceira camada contém seis neurônios com função ativação identidade, significando neurônios estáticos, e função de saída do tipo tangente hiperbólica.
- A última camada tem um único neurônio com função ativação identidade, significando um neurônio estático, e função de saída do tipo tangente hiperbólica.

A identificação se deu em duas etapas : uma sessão de treinamento, onde a RNA teve os seus pesos ajustados em função dos dados coletados e outra sessão onde foram calculadas as estimativas das funções correlações amostradas determinadas na Equação (3.25). Os resultados são mostrados a seguir. A Figura 4.3 mostra o comportamento da RNA no início do treinamento e a Figura 4.4 o sinal de entrada  $u(k)$ . Após 195077 iterações, interrompe-se a sessão de treinamento com o comportamento do modelo neural sendo mostrado na Figura 4.5.

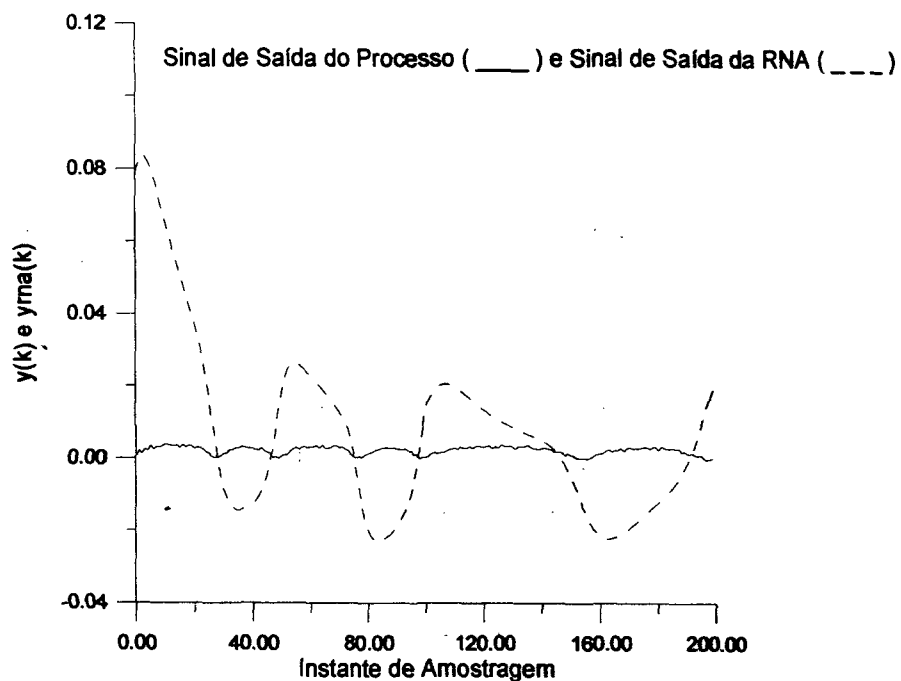


Figura 4. 3 - Comportamento do modelo neural no início do aprendizado

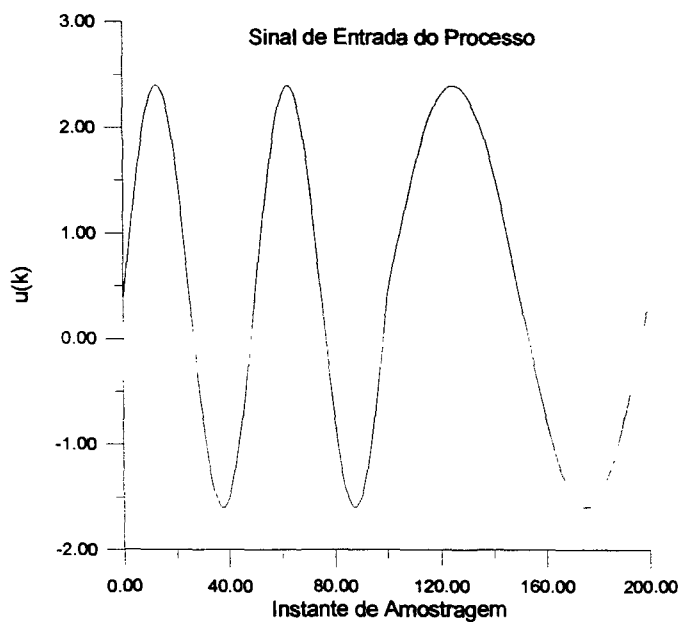


Figura 4. 4 - Sinal de entrada do processo

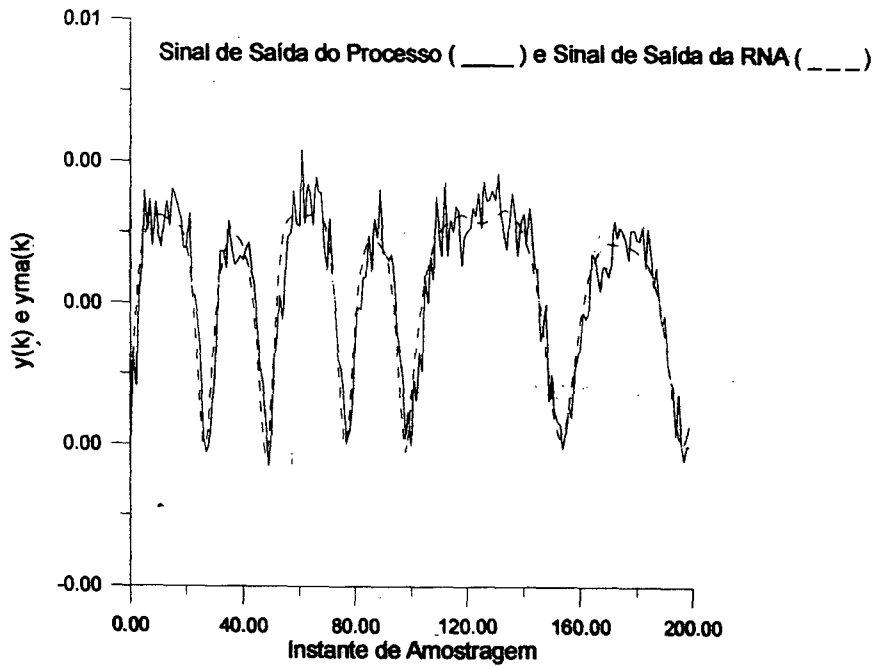


Figura 4. 5 - Comportamento do modelo neural após fase de aprendizado

As estimativas das correlações cruzadas são mostradas na Figura 4. 6.

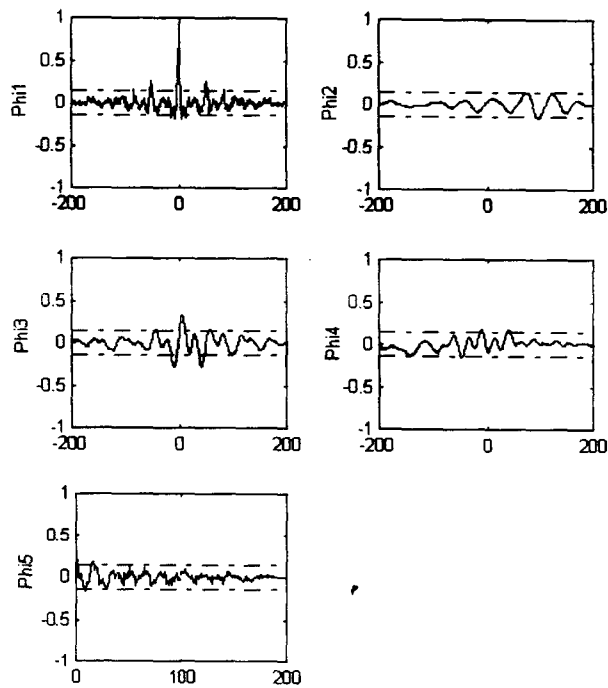


Figura 4. 6 - Estimativas das Funções Correlações Cruzadas :  $\text{Phi1} = \Phi_{\varepsilon\varepsilon}(\tau)$ ;  $\text{Phi2} = \Phi_{u\varepsilon}(\tau)$ ;

$$\text{Phi3} = \Phi_{u^2\varepsilon}(\tau); \text{Phi4} = \Phi_{u^2\varepsilon^2}(\tau); \text{Phi5} = \Phi_{\varepsilon(uu)}(\tau)$$

A evolução da função  $E(.)$ , que é o valor de EMQ (Equação (2.69)) em função do número de iterações do aprendizado, é mostrada na Figura 4.7, onde  $k$  é o número de iterações e indica a apresentação das  $NPT$  amostras para treinamento (lembrando que cada amostra, de uma iteração, proporciona um ajuste nos parâmetros da rede neural).

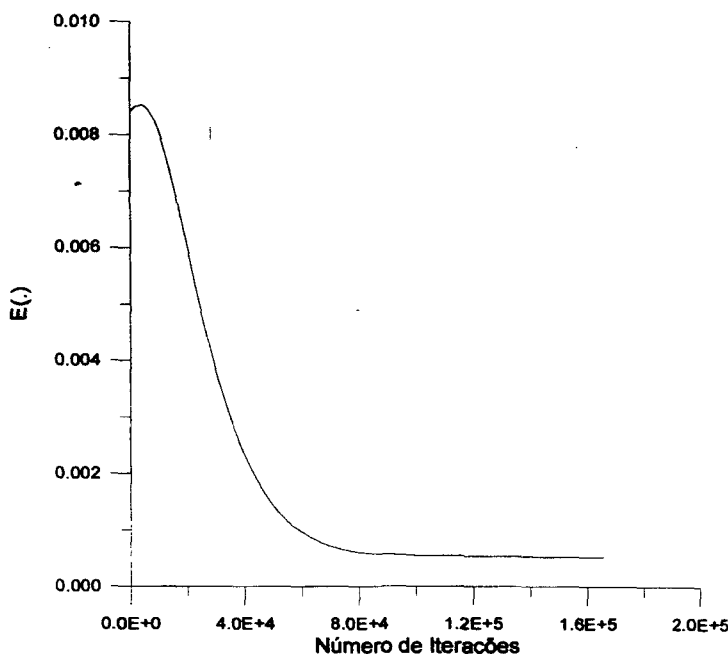


Figura 4. 7 – Evolução da Função Custo  $E(.)$  durante o aprendizado

### 4.3.2 Modelo de Wiener

Outro exemplo de utilização do modelo neural é a identificação, segundo a descrição da Figura (3.1), de um processo não-linear do tipo Wiener de segunda ordem. O processo é descrito pelas equações,

$$x_1(k+1) = -0.7x_1(k) + 0.2x_2(k) + u(k) \tag{4.6}$$

$$x_2(k+1) = 0.3x_1(k) - 0.6x_2(k)$$

$$y(k) = \text{sen}\left((x_1(k))^2\right) + d(k) \tag{4.7}$$

O ruído é simulado por uma sequência pseudo-aleatória de distribuição normal com média 0.0 e variância 0.05. Este valor de variância foi escolhido para que o ruído não deformasse completamente o sinal de saída.

O conjunto de treino é formado por 200 amostras coletadas para um sinal de entrada do tipo senoidal como mostrado na equação,

$$u(k) = \begin{cases} 1.3 \operatorname{sen}\left(\frac{2k\pi}{NPT/1}\right) + 0.2, & k \leq 100 \\ 1.3 \operatorname{sen}\left(\frac{2k\pi}{NPT/2}\right) + 0.2, & k > 100 \end{cases} \quad (4.8)$$

onde  $NPT$  é o número de amostras.

A RNA utilizada é formada por 4 camadas de neurônios, com todas as camadas contendo um *bias*, exceto na primeira camada, onde :

- A primeira camada apresenta um único neurônio com função ativação identidade, significando um neurônio estático, e função de saída do tipo linear com saturação (nível da saturação igual a 100).
- A segunda camada tem dois neurônios com função ativação do tipo equação a diferenças de primeira ordem, onde a entrada do neurônio é afetada pelos estados de todos os outros neurônios dinâmicos, conforme equação (4.9), e função de saída do tipo tangente hiperbólica. Neste caso a matriz  $A$  é cheia, como mostra a equação (4.10).

$$\begin{aligned} x_1(k+1) &= a_{11}x_1(k) + a_{12}x_2(k) + net_1(k) \\ x_2(k+1) &= a_{21}x_1(k) + a_{22}x_2(k) + net_2(k) \end{aligned} \quad (4.9)$$

$$A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \quad (4.10)$$

- A terceira camada contém dez neurônios com função ativação identidade, significando neurônios estáticos, e função de saída do tipo tangente hiperbólica.
- A última camada tem um único neurônio com função ativação identidade, significando um neurônio estático, e função de saída do tipo linear com saturação (nível da saturação igual a 100).

A identificação se deu em duas etapas : uma sessão de treinamento, onde a RNA teve os seus pesos ajustados em função dos dados coletados e outra sessão onde foram calculadas as estimativas das funções correlações amostradas determinadas na Equação (3.25). Os resultados são mostrados a seguir. A Figura 4.

8 mostra o comportamento da RNA no início do treinamento e a Figura 4. 9 o sinal de entrada  $u(k)$ . Após 26852 iterações, interrompe-se a sessão de treinamento com o comportamento do modelo neural sendo mostrado na Figura 4. 10.

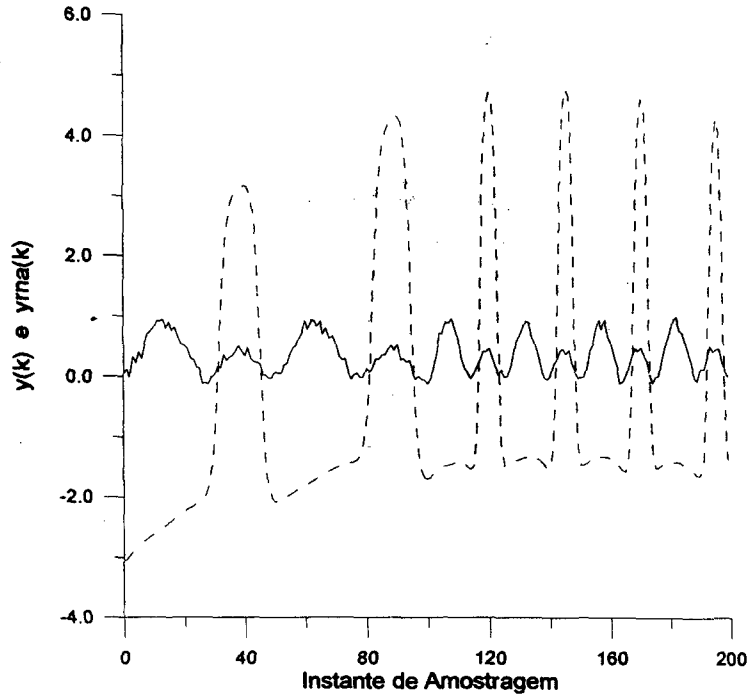


Figura 4. 8 - Comportamento do modelo neural no início do aprendizado

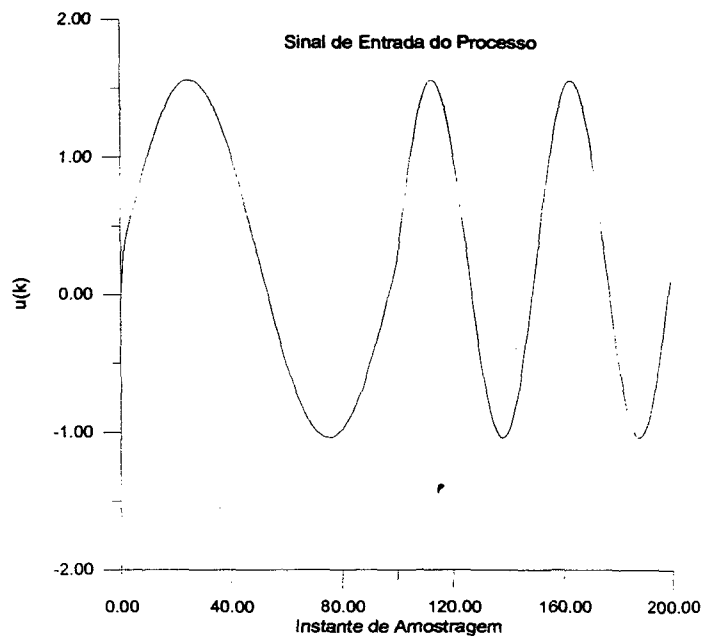


Figura 4. 9 - Sinal de entrada do processo



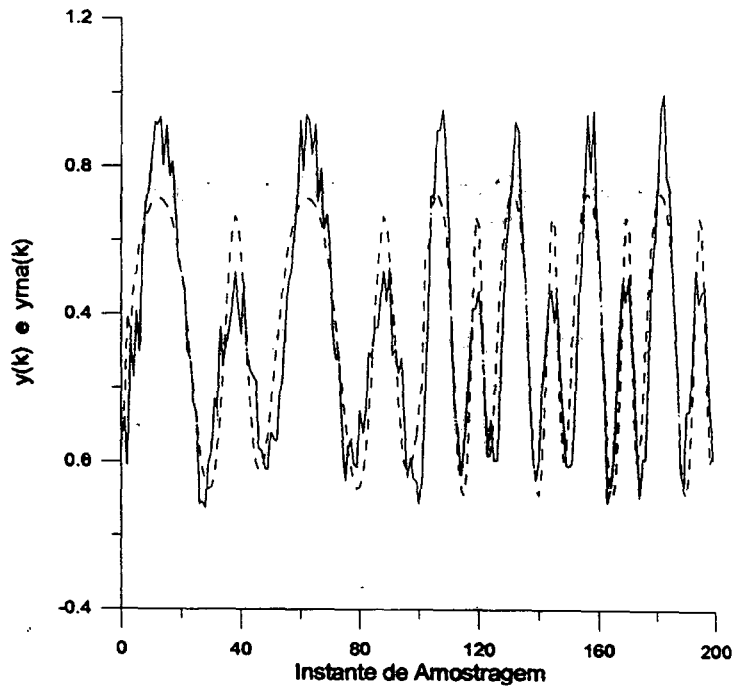


Figura 4. 10 - Comportamento do modelo neural após fase de aprendizado

As estimativas das correlações cruzadas são mostradas na Figura 4. 11.

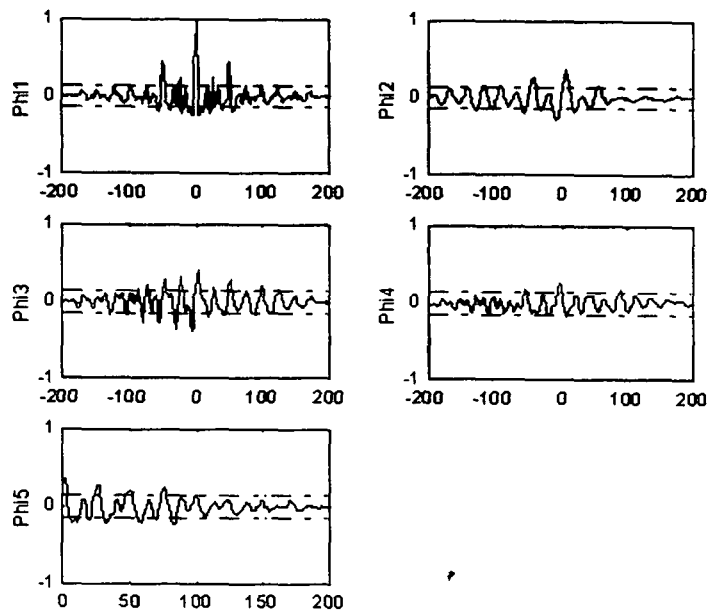


Figura 4. 11 - Estimativas das Funções Correlações Cruzadas :

$$\text{Phi1} = \Phi_{\varepsilon\varepsilon}(\tau); \text{Phi2} = \Phi_{u\varepsilon}(\tau); \text{Phi3} = \Phi_{u^2\varepsilon}(\tau); \text{Phi4} = \Phi_{u^2\varepsilon^2}(\tau); \text{Phi5} = \Phi_{\varepsilon(au)}(\tau)$$

Após as 26852 iterações, que determinou o término do treinamento, continuou-se a adaptação dos pesos e dos elementos da matriz  $A$ . A evolução da função custo  $E(\cdot)$ , que é o valor EMQ (Equação (2.69)) em cada iteração, para este procedimento, é mostrada na Figura 4.12, onde  $k$  é o número de iterações e indica a apresentação das  $NPT$  amostras para treinamento.

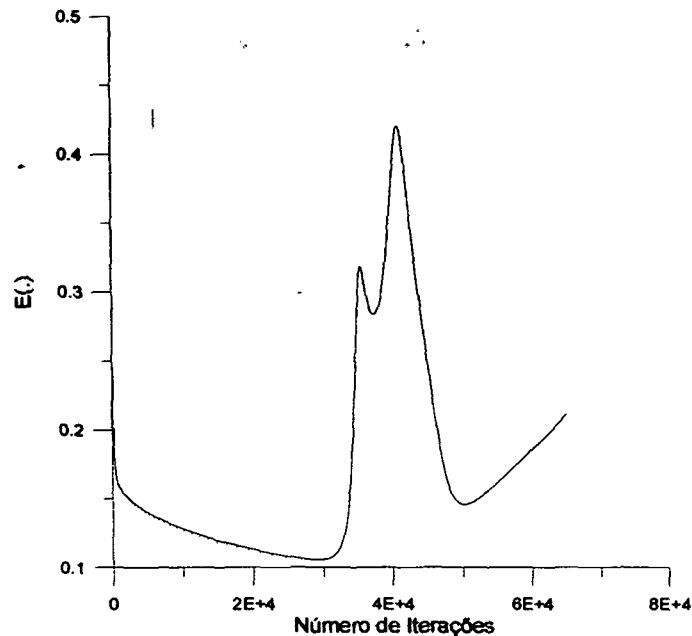


Figura 4. 12 – Evolução da Função Custo  $E(\cdot)$  durante a sessão de aprendizado

### 4.3.3 Modelo de Hammerstein

Outro exemplo de utilização do modelo neural é a identificação, segundo a descrição da Figura (3.1), de um processo não-linear do tipo Hammerstein de segunda ordem. O processo é descrito pelas equações,

$$\begin{aligned} x_1(k+1) &= 0.8x_1(k) - x_2(k) + f[u(k)] \\ x_2(k+1) &= 0.15x_1(k) + 0.3f[u(k)] \end{aligned} \quad (4.11)$$

$$y(k) = x_1(k) + d(k), \quad (4.12)$$

$$f[u] = u - 0.9u^2 + 0.35u^3 \quad (4.13)$$

O ruído é simulado por uma sequência pseudo-aleatória de distribuição normal com média 0.0 e variância 0.3. Este valor de variância foi escolhido para que o ruído não deformasse completamente o sinal de saída.

O conjunto de treino é formado por 200 amostras coletadas para um sinal de entrada do tipo senoidal como mostrado na equação,

$$u(k) = \begin{cases} 1.3 \operatorname{sen}\left(\frac{2k\pi}{NPT/1}\right) + 0.2, & k \leq 100 \\ 1.3 \operatorname{sen}\left(\frac{2k\pi}{NPT/2}\right) + 0.2, & k > 100 \end{cases} \quad (4.14)$$

onde  $NPT$  é o número de amostras. A RNA utilizada é formada por 5 camadas de neurônios, com todas as camadas contendo um *bias*, exceto na primeira camada, onde :

- A primeira camada apresenta um único neurônio com função ativação identidade, significando um neurônio estático, e função de saída do tipo linear com saturação (nível da saturação igual a 10).
- A segunda camada contém nove neurônios com função ativação identidade, significando neurônios estáticos, e função de saída do tipo tangente hiperbólica.
- A terceira camada apresenta um único neurônio com função ativação identidade, significando um neurônio estático, e função de saída do tipo tangente hiperbólica.
- A quarta camada tem dois neurônios com função ativação do tipo equação a diferenças de primeira ordem, onde a entrada do neurônio é afetada pelos estados de todos os outros neurônios dinâmicos, conforme equação (4.15), e função de saída do tipo tangente hiperbólica. Neste caso a matriz  $A$  é cheia, como mostrada na equação (4.16).

$$\begin{aligned} x_1(k+1) &= a_{11}x_1(k) + a_{12}x_2(k) + net_1(k) \\ x_2(k+1) &= a_{21}x_1(k) + a_{22}x_2(k) + net_2(k) \end{aligned} \quad (4.15)$$

$$A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \quad (4.16)$$

- A última camada tem um único neurônio com função ativação identidade, significando um neurônio estático, e função de saída do tipo linear com saturação (nível da saturação igual a 10).

A identificação se deu em duas etapas : uma sessão de treinamento, onde a RNA teve os seus pesos ajustados em função dos dados coletados e outra sessão

onde foram calculadas as estimativas das funções correlações amostradas determinadas na Equação (3.25). Os resultados são mostrados a seguir. A Figura 4. 13 mostra o comportamento da RNA no início do treinamento e a Figura 4. 14 o sinal de entrada  $u(k)$ . Após 7074 iterações, interrompe-se a sessão de treinamento com o comportamento do modelo neural sendo mostrado na Figura 4. 15.

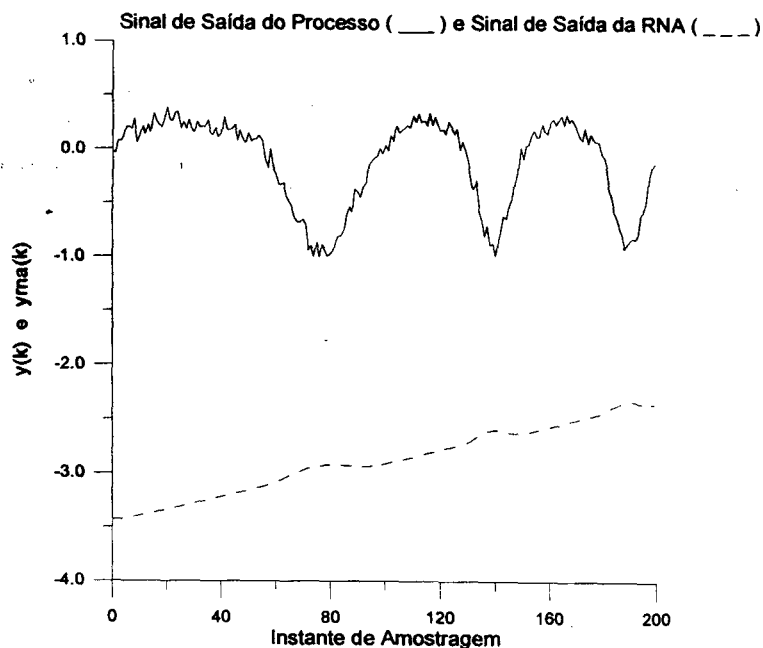


Figura 4. 13 - Comportamento do modelo neural no início do aprendizado

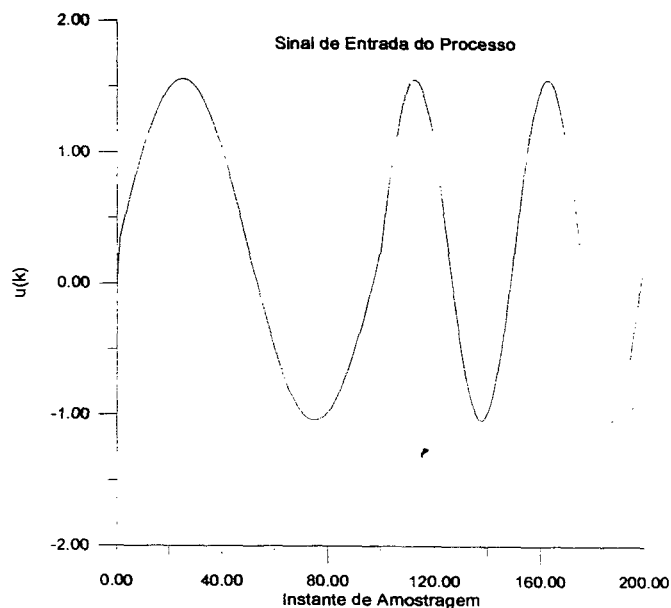


Figura 4. 14 - Sinal de entrada do processo

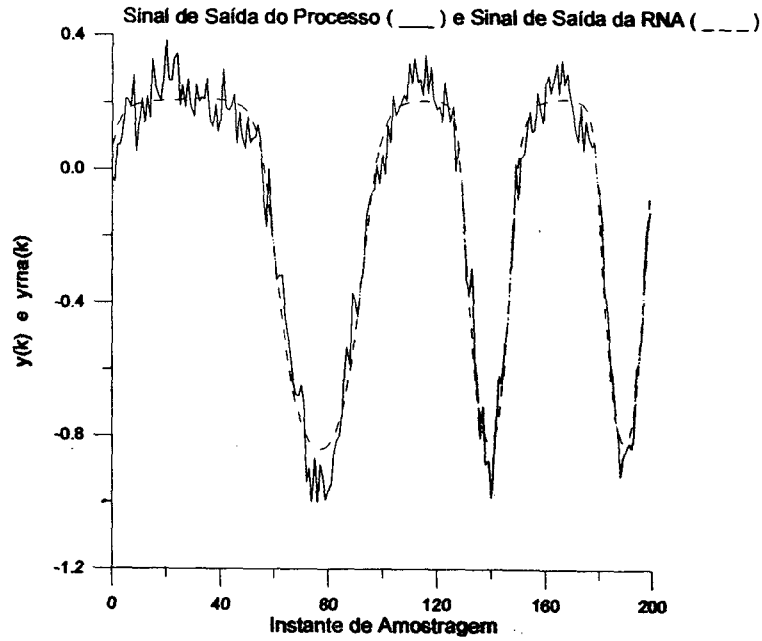


Figura 4. 15 - Comportamento do modelo neural após fase de aprendizado

As estimativas das correlações cruzadas são mostradas na Figura 4.16.

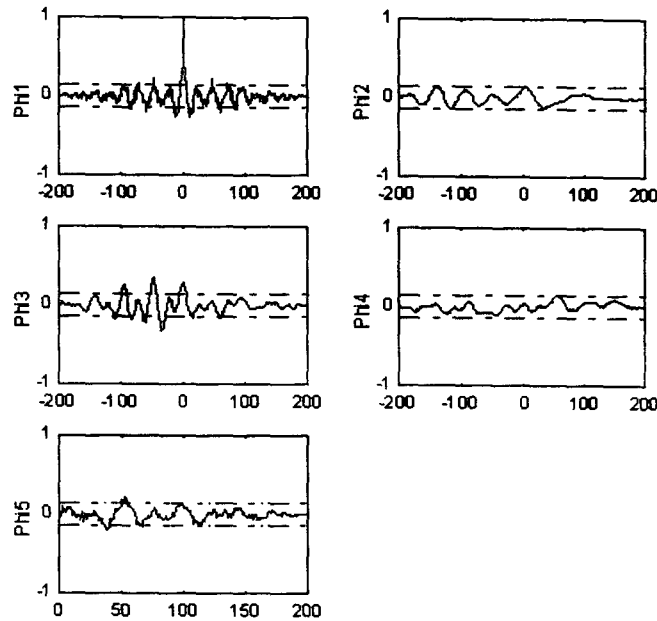


Figura 4.16 - Estimativas das Funções Correlações Cruzadas :

$$\text{Phi1} = \Phi_{\varepsilon\varepsilon}(\tau); \text{Phi2} = \Phi_{u\varepsilon}(\tau); \text{Phi3} = \Phi_{u^2\varepsilon}(\tau); \text{Phi4} = \Phi_{u^2\varepsilon^2}(\tau); \text{Phi5} = \Phi_{\varepsilon(\varepsilon u)}(\tau)$$

A evolução da função custo  $E(.)$  para este procedimento, é mostrada na Figura 4.17, onde  $k$  é o número de iterações e indica a apresentação das  $NPT$  amostras para treinamento.

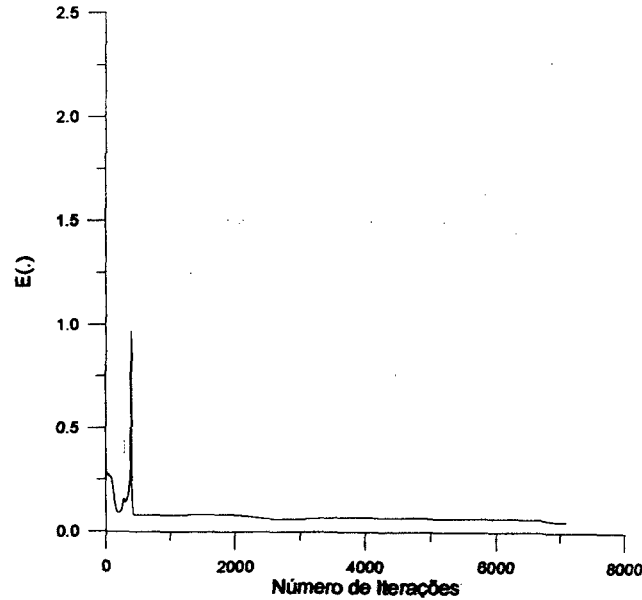


Figura 4. 17 – Evolução da Função Custo E(.) durante a sessão de aprendizado

### 4.3.4 Modelo de Wiener-Hammerstein

Outro exemplo de utilização do modelo neural é a identificação, segundo a descrição da Figura (3.1), de um processo não-linear do tipo Wiener-Hammerstein de segunda ordem. O processo é descrito pelas equações,

$$x_1^1(k+1) = 0.8x_1^1(k) - x_2^1(k) + u(k) \tag{4. 17}$$

$$x_2^1(k+1) = 0.15x_1^2(k) + 0.3u(k)$$

$$y^1(k) = f(x_1^1) \tag{4. 18}$$

$$f(x_1^1) = x_1^1 - 0.9 [x_1^1]^2 + 0.35 [x_1^1]^3 \tag{4. 19}$$

$$x_1^2(k+1) = -0.7x_1^2(k) + 0.2x_2^2(k) + y^1(k) \tag{4. 20}$$

$$x_2^2(k+1) = 0.3x_1^2(k) - 0.6x_2^2(k)$$

$$y(k) = x_1^2(k) + d(k) \tag{4. 21}$$

O ruído é simulado por uma sequência pseudo-aleatória de distribuição normal com média 0.0 e variância 0.3. Este valor de variância foi escolhido para que o ruído não deformasse completamente o sinal de saída.

O conjunto de treino é formado por 200 amostras coletadas para um sinal de entrada do tipo senoidal como mostrado na equação,

$$u(k) = \begin{cases} 1.3 \operatorname{sen}\left(\frac{2k\pi}{NPT/1}\right) + 0.2, & k \leq 100 \\ 1.3 \operatorname{sen}\left(\frac{2k\pi}{NPT/2}\right) + 0.2, & k > 100 \end{cases} \quad (4.22)$$

onde  $NPT$  é o número de amostras.

A RNA utilizada é formada por 7 camadas de neurônios, com todas as camadas contendo um *bias*, exceto na primeira camada, onde :

- A primeira camada apresenta um único neurônio com função ativação identidade, significando um neurônio estático, e função de saída do tipo linear com saturação (nível da saturação igual a 10).
- A segunda camada tem dois neurônios com função ativação do tipo equação a diferenças de primeira ordem, onde a entrada do neurônio é afetada pelos estados de todos os outros neurônios dinâmicos, conforme equação (4.23), e função de saída do tipo tangente hiperbólica. Neste caso a matriz  $A$  é cheia, como mostra a equação (4.24).

$$\begin{aligned} x_1^1(k+1) &= a_{11}^1 x_1^1(k) + a_{12}^1 x_2^1(k) + net_1^1(k) \\ x_2^1(k+1) &= a_{21}^1 x_1^1(k) + a_{22}^1 x_2^1(k) + net_2^1(k) \end{aligned} \quad (4.23)$$

$$A_1 = \begin{bmatrix} a_{11}^1 & a_{12}^1 \\ a_{21}^1 & a_{22}^1 \end{bmatrix} \quad (4.24)$$

- A terceira camada apresenta um único neurônio com função ativação identidade, significando um neurônio estático, e função de saída do tipo tangente hiperbólica.
- A quarta camada contém seis neurônios com função ativação identidade, significando neurônios estáticos, e função de saída do tipo tangente hiperbólica.
- A quinta camada apresenta um único neurônio com função ativação identidade, significando um neurônio estático, e função de saída do tipo tangente hiperbólica.
- A sexta camada tem dois neurônios com função ativação do tipo equação a diferenças de primeira ordem, onde a entrada do neurônio é afetada pelos estados de todos os outros neurônios dinâmicos, conforme equação (4.25), e

função de saída do tipo tangente hiperbólica. Neste caso a matriz  $A$  é cheia, como mostrada na equação (4.26).

$$\begin{aligned} x_1^2(k+1) &= a_{11}^2 x_1^2(k) + a_{12}^2 x_2^2(k) + net_1^2(k) \\ x_2^2(k+1) &= a_{21}^2 x_1^2(k) + a_{22}^2 x_2^2(k) + net_2^2(k) \end{aligned} \tag{4.25}$$

$$A_2 = \begin{bmatrix} a_{11}^2 & a_{12}^2 \\ a_{21}^2 & a_{22}^2 \end{bmatrix} \tag{4.26}$$

- A última camada tem um único neurônio com função ativação identidade, significando um neurônio estático, e função de saída do tipo linear com saturação (nível da saturação igual a 10).

A identificação se deu em duas etapas : uma sessão de treinamento, onde a RNA teve os seus pesos ajustados em função dos dados coletados e outra sessão onde foram calculadas as estimativas das funções correlações amostradas determinadas na Equação (3.25). Os resultados são mostrados a seguir. A Figura 4. 138 mostra o comportamento da RNA no início do treinamento e a Figura 4. 149 o sinal de entrada  $u(k)$ . Após 2 iterações, interrompe-se a sessão de treinamento com o comportamento do modelo neural sendo mostrado na Figura 4. 1520.

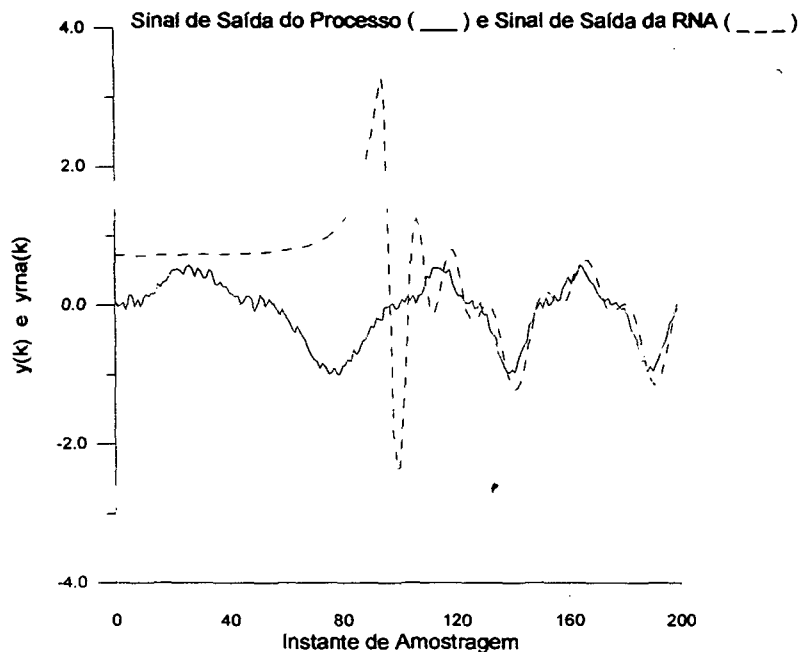


Figura 4. 18 - Comportamento do modelo neural no início do aprendizado



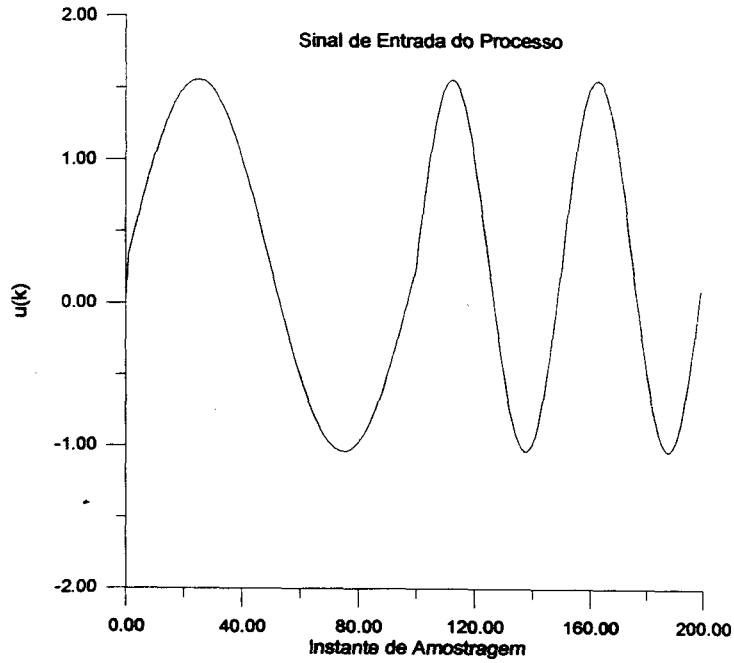


Figura 4. 19 - Sinal de entrada do processo

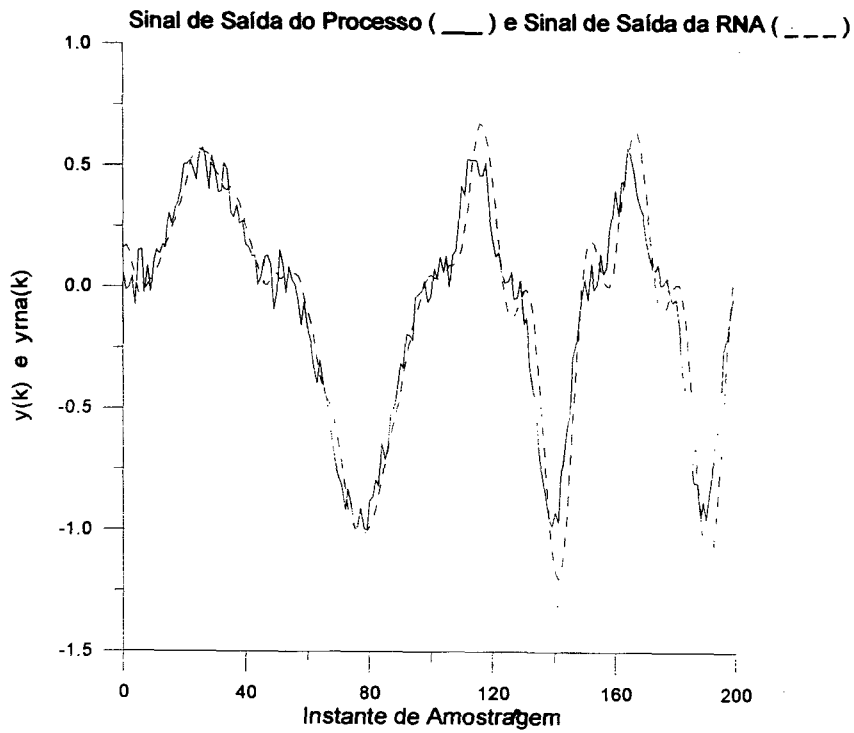


Figura 4. 20 - Comportamento do modelo neural após fase de aprendizado

As estimativas das correlações cruzadas são mostradas na Figura 4.21.

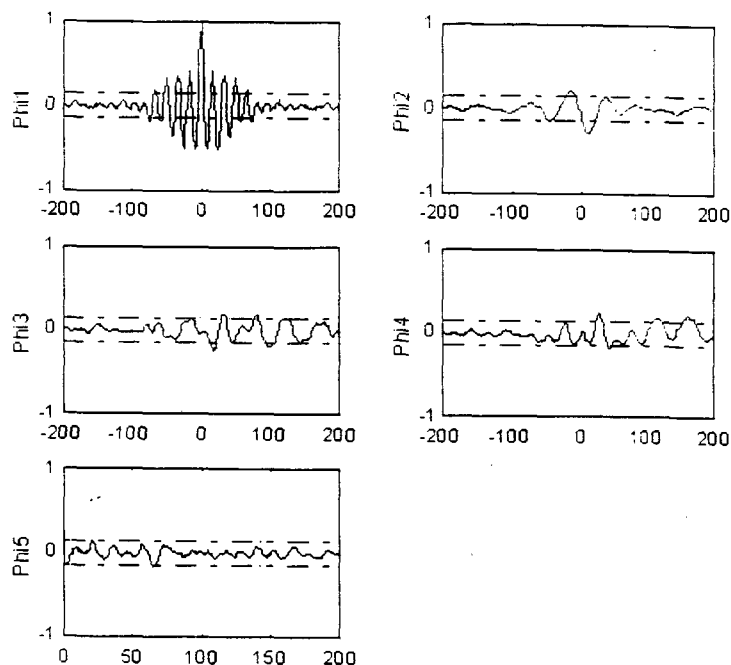


Figura 4.21 - Estimativas das Funções Correlações Cruzadas :

$$\text{Phi1} = \Phi_{\varepsilon\varepsilon}(\tau); \text{Phi2} = \Phi_{u\varepsilon}(\tau); \text{Phi3} = \Phi_{u^2\varepsilon}(\tau); \text{Phi4} = \Phi_{u^2\varepsilon^2}(\tau); \text{Phi5} = \Phi_{\varepsilon(uu)}(\tau)$$

Não mostra-se a evolução da função  $E(\cdot)$  porque o treinamento se realizou em um pequeno número de iterações.

#### 4.4 Discussões

Os resultados mostrados na seção 4.3 confirmam a proposição de uso do modelo neural com dinâmica interna, desenvolvido na seção (2.3), para identificação de processos orientados à blocos. Embora este modelo reproduza a estrutura do processo, devido as suas características não-lineares e a distribuição da informação, a estimação dos parâmetros do modelo, na sua parte dinâmica linear, fica distante dos valores reais encontrados no processo. Este é um fato decorrente do uso de camadas de neurônios estáticos para reproduzir a não-linearidade estática do processo que é essencial para o bom desempenho global do modelo, mas prejudica a estimação local dos parâmetros referentes as camadas de neurônios dinâmicos. A princípio este fenômeno, apresentado pela rede, não parece adequado a uma tarefa de identificação paramétrica, entretanto apesar desta diferença entre os parâmetros

do bloco dinâmico linear do processo com a camada dinâmica linear do modelo neural, a estimativa das funções correlações cruzadas indica um desempenho aceitável para a identificação. Nas sub-seções a seguir, mostra-se algumas estimativas dos parâmetros  $a_{ij}$ 's realçando as diferenças já citadas. Embora o uso do modelo neural desenvolvido para identificar processos descritos por blocos lineares dinâmicos e por blocos não-lineares estáticos tenha se mostrado um sucesso, o algoritmo de treinamento utilizado não prevê a situação de instabilidade na parte linear dinâmica da rede. Estes dois assuntos são exemplificados também nas sub-seções seguintes.

#### 4.4.1 Resultados das Simulações

Os resultados da seção 4.3.1 poderiam ser coletados em apenas 80000 iterações já que, como mostra a Figura 4. 7, o nível da função custo  $E(.)$  não varia significativamente a partir deste ponto. A evolução de  $E(.)$  é suave como as curvas padrões do uso do algoritmo *backpropagation* em modelos neurais com dinâmica externa. Este comportamento não é o padrão para o uso do algoritmo *backpropagation* em modelos neurais com dinâmica interna, devido as variações abruptas que ocorrem nos valores de autovalores da matriz  $A$  das camadas dinâmicas do modelo neural. A pequena discrepância encontrada na função correlação cruzada  $\Phi_{i3}$  não é significativa porque não indica uma tendência e ocorre em poucos pontos.

Para o processo descrito nesta mesma seção utilizou-se também uma entrada  $u(k)$  do tipo PRBS (Pseudo-Random Binary Sequence) com 50 amostras e amplitude igual a  $\pm 0.5$  no procedimento de identificação neural. Aqui o objetivo é verificar o possível casamento entre os parâmetros do modelo neural com os parâmetros do processo, além é claro do mapeamento entrada-saída conseguido pela RNA. O uso da sequência PRBS como sinal de entrada é para observar o efeito do sinal  $u(.)$  na evolução dos parâmetros da rede. A topologia da rede é parecida com a da seção 4.3.1, a única diferença é que a terceira camada de neurônios da rede contém oito neurônios estáticos. Os resultados obtidos são mostrados nas figuras a seguir, após

493925 iterações de treinamento. Na estimativa das funções correlações cruzadas as medidas de  $\Phi_{i3}$  e  $\Phi_{i4}$  são ignoradas porque para este tipo de sinal de entrada o vetor  $u^2$  é nulo.

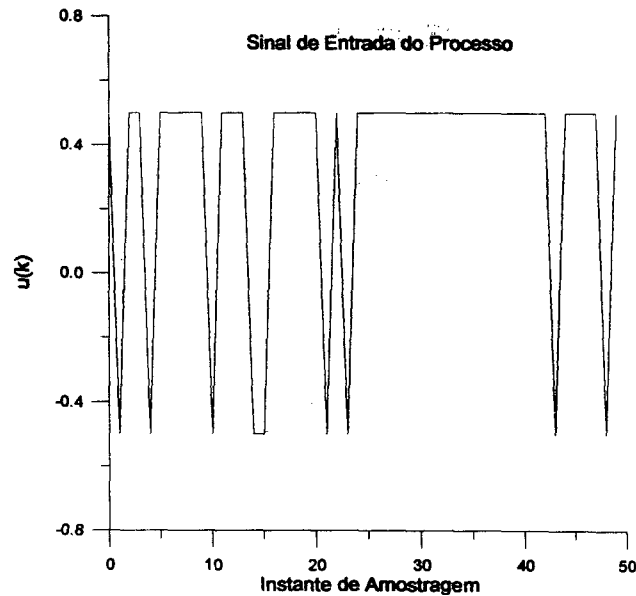


Figura 4. 22 - - Sinal do tipo PRBS usado como entrada do processo para o modelo de Hammerstein-Wiener da seção 4.3.1

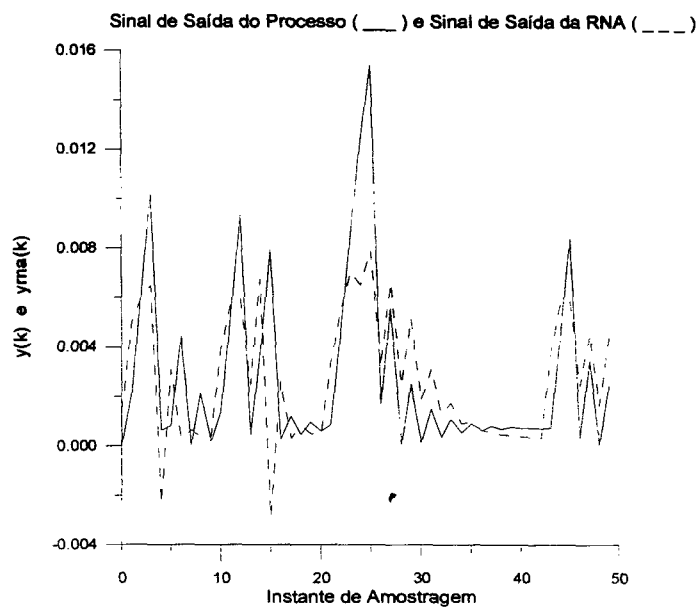


Figura 4. 23 - Comportamento do modelo neural após fase de aprendizado com  $u(k)$  do tipo PRBS, para o modelo de Hammerstein-Wiener da seção 4.3.1

493925 iterações de treinamento. Na estimativa das funções correlações cruzadas as medidas de  $\Phi_3$  e  $\Phi_4$  são ignoradas porque para este tipo de sinal de entrada o vetor  $u^2$  é nulo.

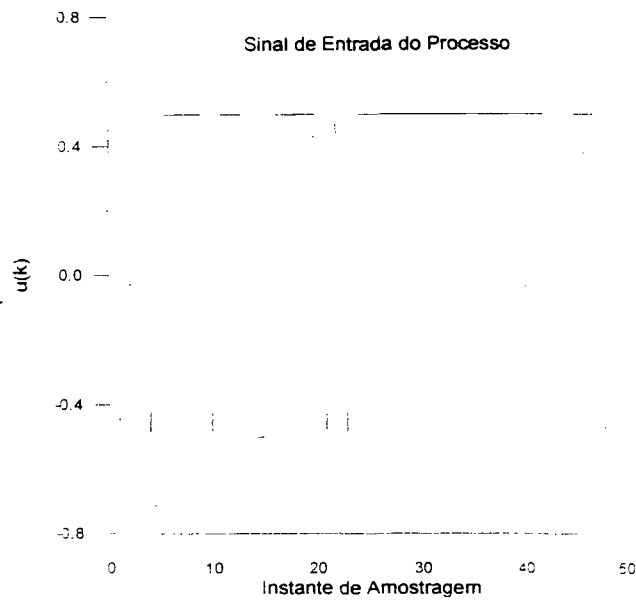


Figura 4. 22 - - Sinal do tipo PRBS usado como entrada do processo para o modelo de Hammerstein-Wiener da seção 4.3.1

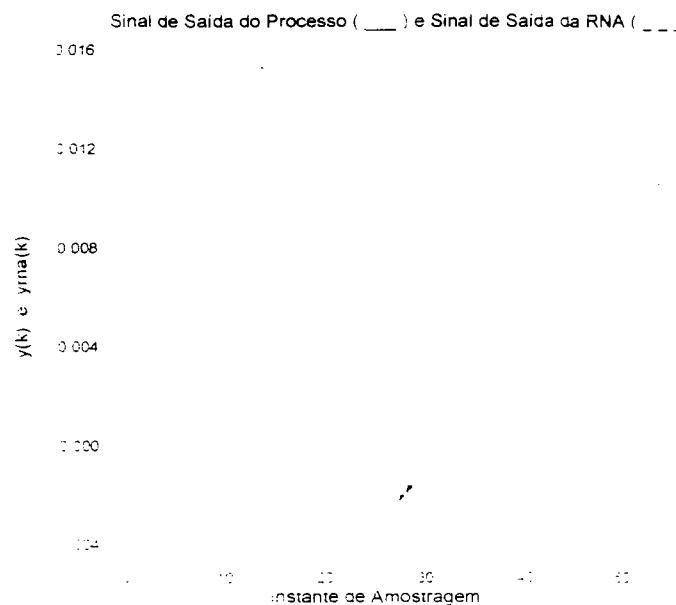


Figura 4. 23 - Comportamento do modelo neural após fase de aprendizado com  $u(k)$  do tipo PRBS, para o modelo de Hammerstein-Wiener da seção 4.3.1

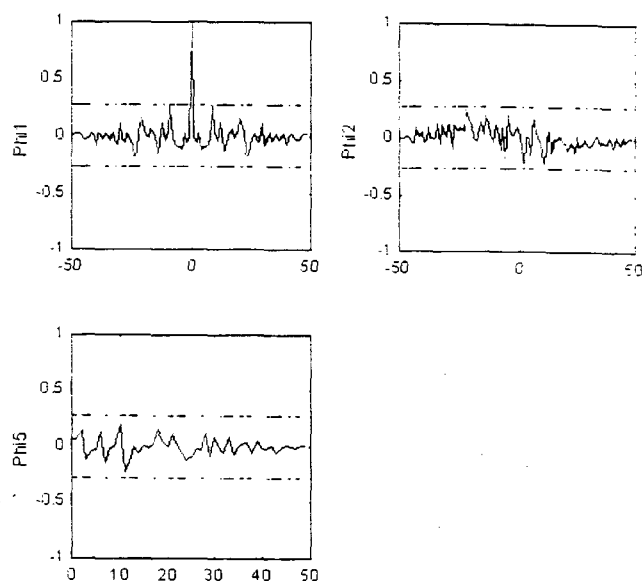


Figura 4. 24 -- Estimativa das Funções Correlações Cruzadas :  $\Phi_{11} = \Phi_{\varepsilon\varepsilon}(\tau)$ ;  $\Phi_{22} = \Phi_{u\varepsilon}(\tau)$ ;  $\Phi_{55} = \Phi_{\varepsilon(au)}(\tau)$  com  $u(k)$  do tipo PRBS, para o modelo de Hammerstein-Wiener da seção 4.3.1

Como pode-se observar a partir da Figura 4. 24, o procedimento de identificação é um sucesso. Um detalhe interessante é que o número de iterações necessárias para a conclusão do treinamento é elevado, se comparado com o caso original da seção citada.

Como já descrito anteriormente, este caso apresenta grandes diferenças entre os autovalores estimados da camada dinâmica linear do modelo neural com os autovalores do processo, mesmo modificando-se o tipo de sinal de entrada. Estes são -0.7 e -0.6 enquanto que aqueles têm valores de -0.2844 e -0.8420. Convém notar que embora a rede não encontre os valores exatos para os pólos do modelo, a mesma mantém a dominância de um frente o outro. Ou seja, o processo tem um pólo mais dominante em -0.7, e o modelo neural também tem um pólo dominante, neste caso em -0.842.

A evolução dos autovalores estimados na camada dinâmica linear do modelo neural é mostrada a seguir.

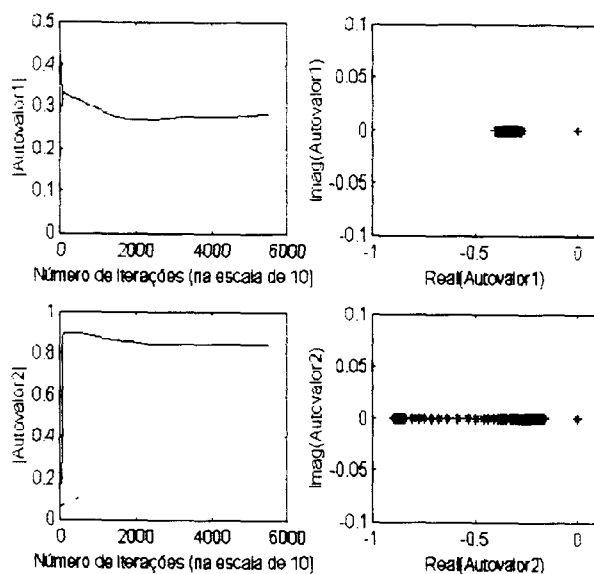


Figura 4. 25 - Comportamento dos autovalores referentes à camada dinâmica linear do modelo neural, com  $u(k)$  do tipo PRBS, durante o treinamento para identificar o modelo de Hammerstein-Wiener da seção 4.3.1

Os trabalhos existentes com redes neurais recorrentes em que existem o casamento perfeito entre os elementos da matriz  $A$  e os elementos do modelo neural só existe para os casos em que o processo é linear (Raol, 1995). Para o caso de processos não-lineares este assunto não é levado em consideração, só importando o desempenho do sinal de saída da rede em relação ao sinal de saída do processo (Delgado et al., 1995; Karakasoglu et al., 1993; Horne e Giles, 1995; Ku e Lee, 1995; Pham e Liu, 1992, 1996; Saxén e Saxén, 1994; Tsung, 1994).

Os resultados da seção 4.3.2, observados nas estimativas das funções correlações cruzadas da Figura 4. 11, indicam um desempenho aceitável para o procedimento de identificação neural. Embora todas as funções  $\Phi_i$ 's apresentem alguns pontos fora do intervalo de confiança, mais acentuados em  $\Phi_3$ , os mesmos são em pequeno número e não indicam tendência de ficarem fora deste intervalo. Por este motivo estas discrepâncias são irrelevantes.

A evolução da função custo  $E(.)$  observada na Figura 4. 12 mostra oscilações depois da mesma atingir o seu valor de  $E(.)$  mínimo. Esta figura também mostra

minúsculas oscilações antes da rede atingir o mínimo. Oscilações na evolução da função custo  $E(.)$  é um comportamento normal para este modelo neural proposto, apesar do algoritmo de treinamento ser do tipo *backpropagation*. Nesta simulação o número de iterações necessárias, para se alcançar valores razoáveis de  $E(.)$ , é baixo se leva-se em consideração o número apresentado na seção 4.3.1. Estes números são sempre relativos porque dependem em grande parte dos valores iniciais dos parâmetros da rede neural.

Os resultados da seção 4.3.3, observados nas estimativas das funções correlações cruzadas da Figura 4.16, indicam um bom desempenho para o procedimento de identificação neural. Novamente, como nos dois casos anteriores, a estimativa da função correlação cruzada  $\Phi_{i3}$  apresenta alguns poucos pontos fora do intervalo de confiança que pelos mesmos motivos anteriores, são insignificantes em relação ao contexto da identificação neural. A evolução da função custo  $E(.)$ , Figura 4.17, mostra um transitório no início do procedimento de aprendizagem e depois o custo decai suavemente de forma muito lenta (comportamento típico do *backpropagation*). Novamente, como na seção 4.3.1, o treinamento poderia ser interrompido por volta da iteração de número 500 sem que houvesse mudanças significativas na medida de  $E(.)$ .

Na seção 4.3.3 o tipo de não-linearidade utilizada é do tipo polinomial finita. Também foi simulado um caso em que a não-linearidade não é do tipo polinomial finita. Os resultados para este caso, que repete a seção 4.3.3, exceto a não-linearidade que agora é igual a  $f[u] = \tanh(u^2)$  e a RNA que tem a seguinte topologia:

- Primeira camada apresenta um único neurônio com função ativação identidade, significando um neurônio estático, e função de saída do tipo linear com saturação (nível da saturação igual a 10).
- Segunda camada contém cinco neurônios com função ativação identidade, significando neurônios estáticos, e função de saída do tipo tangente hiperbólica.
- Terceira camada contém três neurônios com função ativação identidade, significando um neurônio estático, e função de saída do tipo tangente hiperbólica.



- Quarta camada apresenta um único neurônio com função ativação identidade, significando um neurônio estático, e função de saída do tipo linear com saturação (nível da saturação igual a 10).
- Quinta camada tem dois neurônios com função ativação do tipo equação a diferenças de primeira ordem, onde a entrada do neurônio é afetada pelos estados de todos os outros neurônios dinâmicos, conforme equação (4.27), e função de saída do tipo tangente hiperbólica. Neste caso a matriz A é cheia, como mostrada na equação (4.28).

$$\begin{aligned} x_1(k+1) &= a_{11}x_1(k) + a_{12}x_2(k) + net_1(k) \\ x_2(k+1) &= a_{21}x_1(k) + a_{22}x_2(k) + net_2(k) \end{aligned} \tag{4.27}$$

$$A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \tag{4.28}$$

- A última camada tem um único neurônio com função ativação identidade, significando um neurônio estático, e função de saída do tipo linear com saturação (nível da saturação igual a 10).

Os resultados obtidos são mostrados nas figuras a seguir.

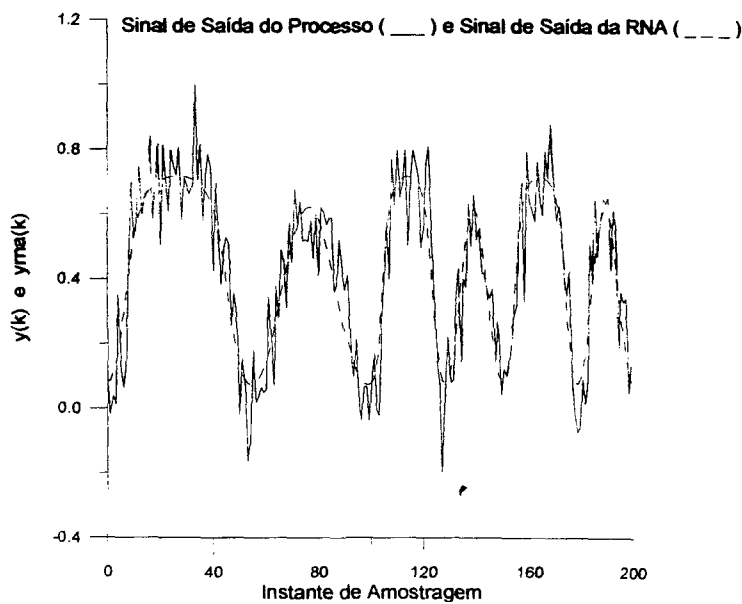


Figura 4. 26 - - Comportamento do modelo neural após fase de aprendizado com não-linearidade do tipo não-polinomial (no caso,  $\tanh(.)$ ), para o modelo de Hammerstein da seção 4.3.3

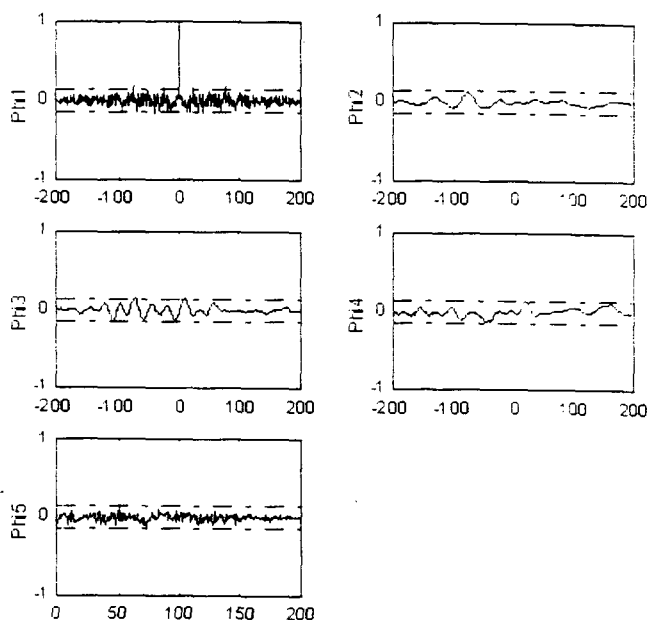


Figura 4. 27 - Estimativa das Funções Correlações Cruzadas :

$\Phi_{11} = \Phi_{\varepsilon\varepsilon}(\tau)$ ;  $\Phi_{12} = \Phi_{u\varepsilon}(\tau)$ ;  $\Phi_{13} = \Phi_{u^2\varepsilon}(\tau)$ ;  $\Phi_{14} = \Phi_{u^2\varepsilon^2}(\tau)$ ;  $\Phi_{15} = \Phi_{\varepsilon(\varepsilon u)}(\tau)$ , com não-linearidade do tipo não-polinomial, para o modelo de Hammerstein da seção 4.3.3

A Figura 4. 27 mostra o bom desempenho da identificação neural, agora para uma não-linearidade não-polinomial, que é um caso onde os procedimentos de identificação paramétricos existentes não abordam (Billings e Fakhouri, 1978; Gallman, 1976; Hwang e Shyu, 1988; Åström e Wittenmark, 1995).

Os resultados da seção 4.3.4, observados nas estimativas das funções correlações cruzadas da Figura 4.21, indicam um desempenho aceitável para o procedimento de identificação neural. Como já notado nas simulações anteriores, as discrepâncias existentes em algumas funções  $\Phi_{ii}$ s não são relevantes, o que chama a atenção é o fato de que  $\Phi_{11}$ , que mede a quantidade de informação do sinal de ruído na saída do modelo, tenha mais pontos do que as outras  $\Phi_{ii}$ s fora do intervalo de confiança. Este comportamento de  $\Phi_{11}$  é explicado pelo baixíssimo número de iterações envolvidas no treinamento que ficou apenas em 2 iterações.

A efetivação do treinamento para este tipo de modelo foi o mais complicado dos quatro modelos utilizados. Em 95% das simulações realizadas o modelo neural

ficou instável em menos de 100 iterações de treinamento. Este comportamento instável para o modelo é devido a um dos pólos do segundo bloco dinâmico linear do processo estar próximo do círculo unitário, valor igual a -0.9, levando o ajuste dos parâmetros da segunda camada dinâmica do modelo neural a colocar os seus autovalores fora do círculo unitário do plano-Z muito rapidamente. O algoritmo de aprendizado, embora não convirja para os autovalores reais do processo, sempre tenta encontrá-los. Como um deles está próximo da instabilidade estas tentativas levam facilmente a instabilidade do modelo.

#### 4.4.2 Uso do erro de predição

O uso do erro de predição  $\varepsilon(k)$  como mais um sinal de entrada da rede neural, além do  $u(k)$ , é efetivado para os casos em que existe a representação não-linear para o ruído na saída do processo. O ruído de medida não-linear é utilizado em dois dos quatro processos simulados da seção anterior, em duas situações : presença do erro de predição na entrada da rede e sem o mesmo como sinal de entrada do modelo neural. Os resultados apresentados foram conseguidos para os processos representados por modelos de Wiener e por modelos de Hammerstein-Wiener.

- Modelo de Wiener : ruído de medida não-linear do tipo  $y(k) = \text{sen}[x_1(k) + d(k)]$  com  $d(k)$  sendo uma sequência ruído branco gaussiano. O resultado final do treinamento foi alcançado em 2744 iterações com um valor de função custo igual à 0.1089. Para a mesma situação de ruído não-linear na saída do processo, mas sem o uso do sinal  $\varepsilon(k)$ , chega-se ao final do treinamento em 240 iterações com uma função custo igual à 0.073. A Figura 4. 28 e a Figura 4. 29 mostram o comportamento da rede neural no final do treinamento para o caso em que não se utiliza o erro de predição e as correlações cruzadas para este caso, respectivamente. A Figura 4. 30 e a Figura 4. 31 mostram o comportamento da rede neural no final do treinamento para o caso em que se utiliza o erro de predição e as correlações cruzadas para este caso, respectivamente.

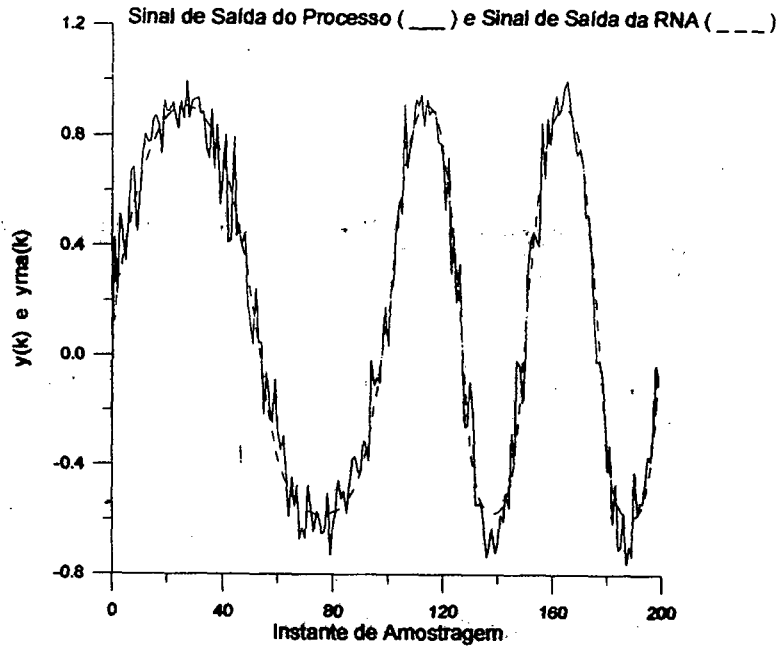


Figura 4. 28 - Comportamento do modelo neural após fase de treinamento para o caso de ruído de medida não-linear e sem utilizar o sinal de erro de predição na entrada da rede neural com o modelo de Wiener

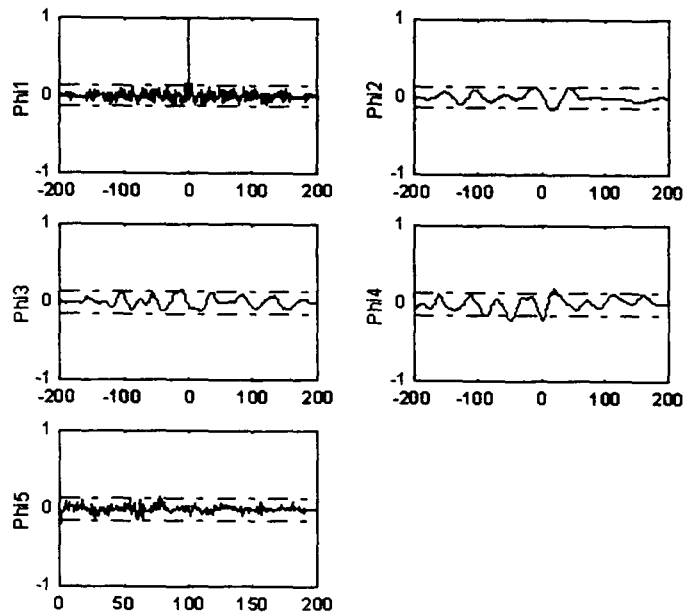


Figura 4. 29 - Estimativa das Funções Correlações Cruzadas :

$\Phi_{11} = \Phi_{\varepsilon\varepsilon}(\tau)$ ;  $\Phi_{12} = \Phi_{u\varepsilon}(\tau)$ ;  $\Phi_{13} = \Phi_{u^2\varepsilon}(\tau)$ ;  $\Phi_{14} = \Phi_{u^2\varepsilon^2}(\tau)$ ;  $\Phi_{15} = \Phi_{\varepsilon(au)}(\tau)$ , para o caso de ruído de medida não-linear e sem utilizar o sinal de erro de predição na entrada da rede neural com o modelo de Wiener

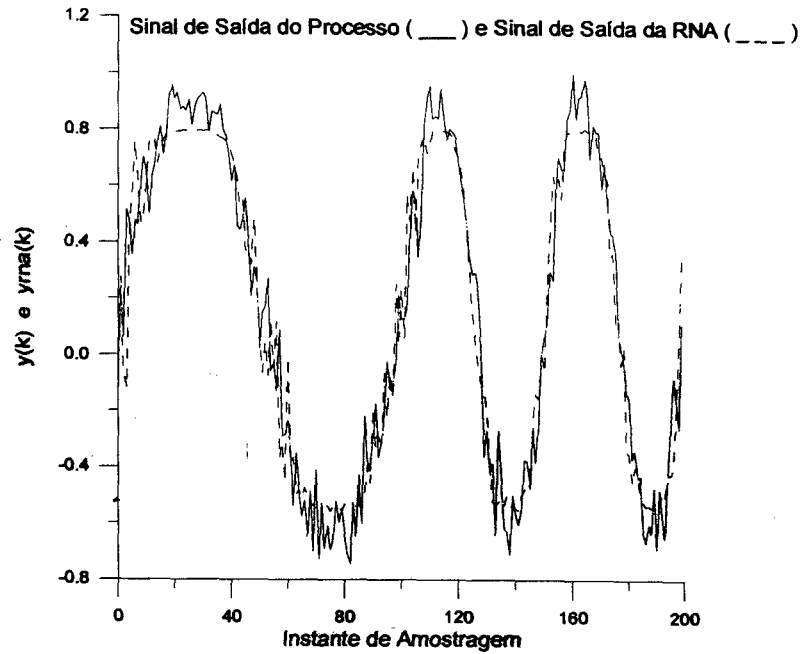


Figura 4. 30 - Comportamento do modelo neural após fase de treinamento para o caso de ruído de medida não-linear e com a utilização do sinal de erro de predição na entrada da rede neural com o modelo de Wiener

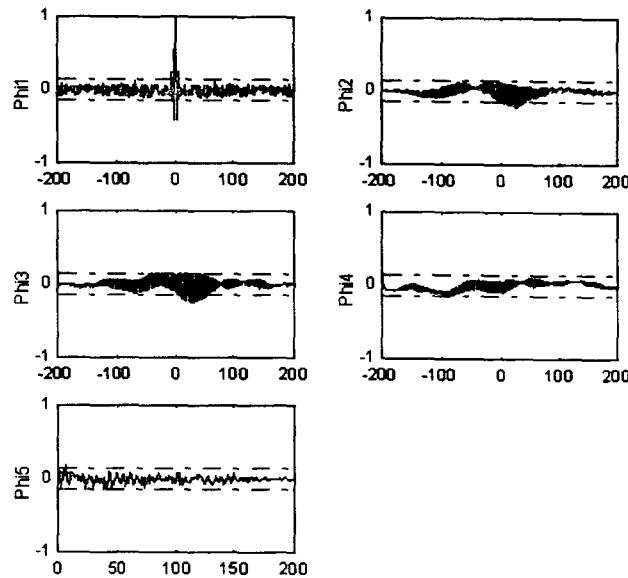


Figura 4. 31 - Estimativa das Funções Correlações Cruzadas :  
 $\Phi_1 = \Phi_{\epsilon\epsilon}(\tau)$ ;  $\Phi_2 = \Phi_{u\epsilon}(\tau)$ ;  $\Phi_3 = \Phi_{u^2\epsilon}(\tau)$ ;  $\Phi_4 = \Phi_{u^2\epsilon^2}(\tau)$ ;  $\Phi_5 = \Phi_{\epsilon(au)}(\tau)$ , para o caso de ruído de medida não-linear e com utilização do sinal de erro de predição na entrada da rede neural com o modelo de Wiener

Como nota-se nos resultados mostrados nas figuras acima, o uso do sinal  $\varepsilon(k)$  não acrescentou benefícios significantes à identificação. Inclusive, o uso de  $\varepsilon(k)$  exigiu um maior número de iterações para o treinamento. As estimativas das funções correlações cruzadas tiveram pequenas mudanças significativas. Quando não se utiliza o sinal  $\varepsilon(k)$  existem pequenas discrepâncias nas estimativas de  $\Phi_2$ ,  $\Phi_3$  e  $\Phi_4$  (Figura 4. 29). Quando se utiliza o sinal  $\varepsilon(k)$  continuam existindo pequenas discrepâncias nas estimativas das  $\Phi_i$ 's, só que agora elas ocorrem apenas em  $\Phi_2$  e  $\Phi_3$  (Figura 4. 31). Neste último caso, os pontos fora da banda de confiança que aparecem em  $\Phi_1$  são irrelevantes porque o formato geral da estimativa continua sendo a de um impulso. Nestas duas simulações a topologia do modelo neural, número de camadas e o número e tipo de neurônios por camadas, é idêntica. Esta topologia utilizada foi a que apresentou melhores resultados para ambas as situações, e é representada por uma RNA com 4 camadas, onde :

- Primeira camada apresenta um único neurônio com função ativação identidade, significando um neurônio estático, e função de saída do tipo linear com saturação (nível da saturação igual a 10). No caso do uso de  $\varepsilon(k)$ , esta camada apresenta dois neurônios (um para receber  $u(k)$  e outro para receber  $\varepsilon(k)$ ), permanecendo o resto da configuração.
- Segunda camada tem dois neurônios com função ativação do tipo equação a diferenças de primeira ordem, onde a entrada do neurônio é afetada pelos estados de todos os outros neurônios dinâmicos, conforme equação (4.9), e função de saída do tipo tangente hiperbólica. Neste caso a matriz  $A$  é cheia equação (4.10).
- Terceira camada contem seis neurônios com função ativação identidade, significando um neurônio estático, e função de saída do tipo tangente hiperbólica.
- A última camada tem um único neurônio com função ativação identidade, significando um neurônio estático, e função de saída do tipo linear com saturação (nível da saturação igual a 10).

- Modelo de Hammerstein-Wiener : neste caso, as equações do processo são as mesmas das utilizadas na seção (4.3.3) (modelo de Hammerstein), só que ao incluir-se o ruído de medida não-linear do tipo  $y(k) = \text{sen}[x_1(k) + d(k)]$  com  $d(k)$  sendo uma sequência ruído branco gaussiano, o processo passa a ser do tipo Hammerstein-Wiener. O resultado final do treinamento, com o uso do sinal de erro de predição na entrada do modelo neural, foi alcançado em 26 iterações com um valor de função custo igual à 0.4507. Para a mesma situação de ruído não-linear na saída do processo, mas sem o uso do sinal  $\varepsilon(k)$ , chega-se ao final do treinamento em 373 iterações com uma função custo igual à 0.4098. A Figura 4. 32 e a Figura 4. 33 mostram o comportamento da rede neural no final do treinamento para o caso em que se utiliza o erro de predição e as correlações cruzadas para este caso, respectivamente. A Figura 4. 34 e a Figura 4. 35 mostram o comportamento da rede neural no final do treinamento para o caso em que não se utiliza o erro de predição e as correlações cruzadas para este caso, respectivamente.

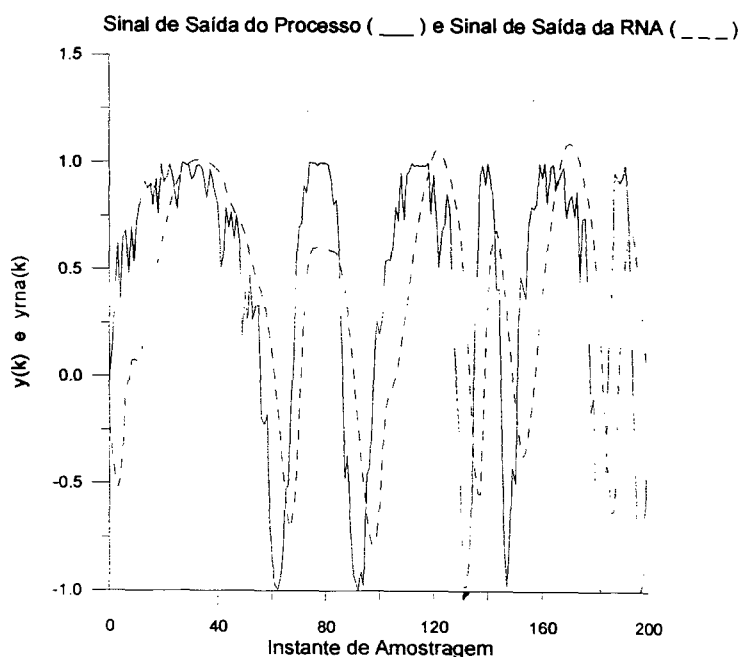


Figura 4. 32 - Comportamento do modelo neural após fase de treinamento para o caso de ruído de medida não-linear com a utilização do sinal de erro de predição na entrada da rede neural com o modelo de Hammerstein-Wiener

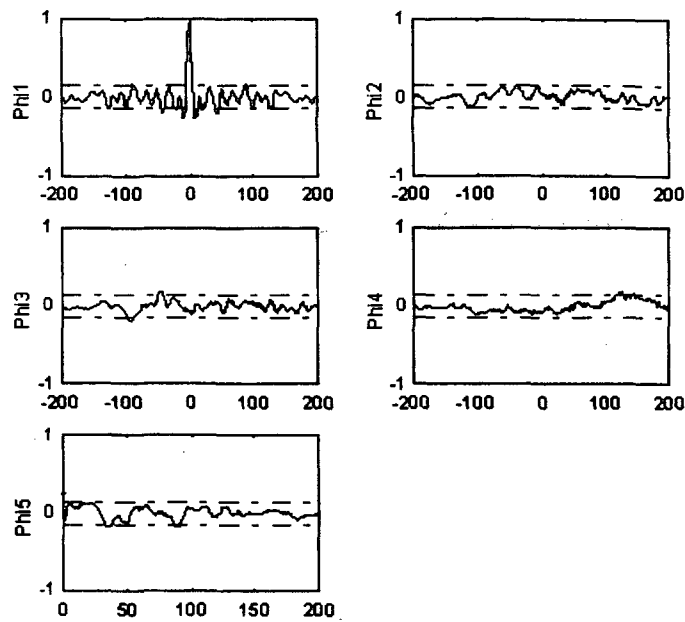


Figura 4. 33 - Estimativa das Funções Correlações Cruzadas :

$\Phi_{11} = \Phi_{\varepsilon\varepsilon}(\tau)$ ;  $\Phi_{12} = \Phi_{u\varepsilon}(\tau)$ ;  $\Phi_{13} = \Phi_{u^2\varepsilon}(\tau)$ ;  $\Phi_{14} = \Phi_{u^2\varepsilon^2}(\tau)$ ;  $\Phi_{15} = \Phi_{\varepsilon(2u)}(\tau)$ , para o caso de ruído de medida não-linear com a utilização do sinal de erro de predição na entrada da rede neural com o modelo de Hammerstein-Wiener

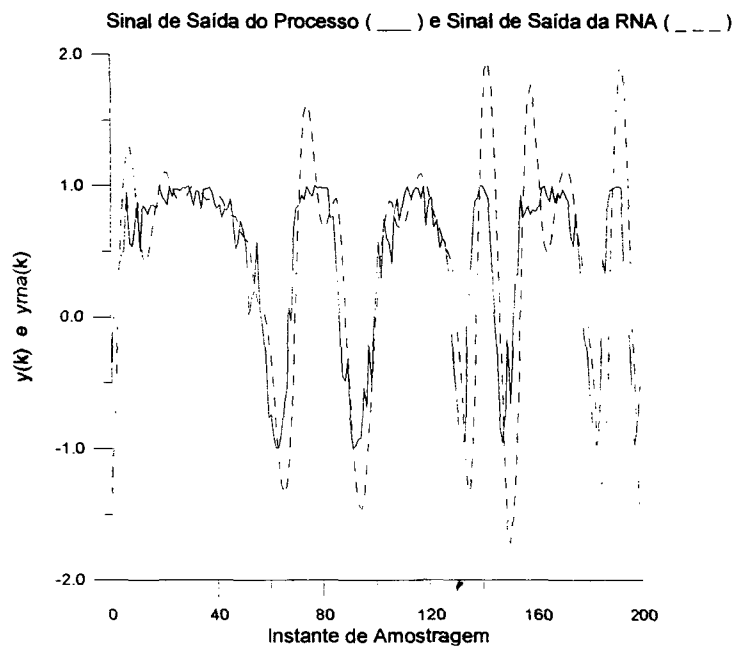


Figura 4. 34 - Comportamento do modelo neural após fase de treinamento para o caso de ruído de medida não-linear sem a utilização do sinal de erro de predição na entrada da rede neural com o modelo de Hammerstein-Wiener



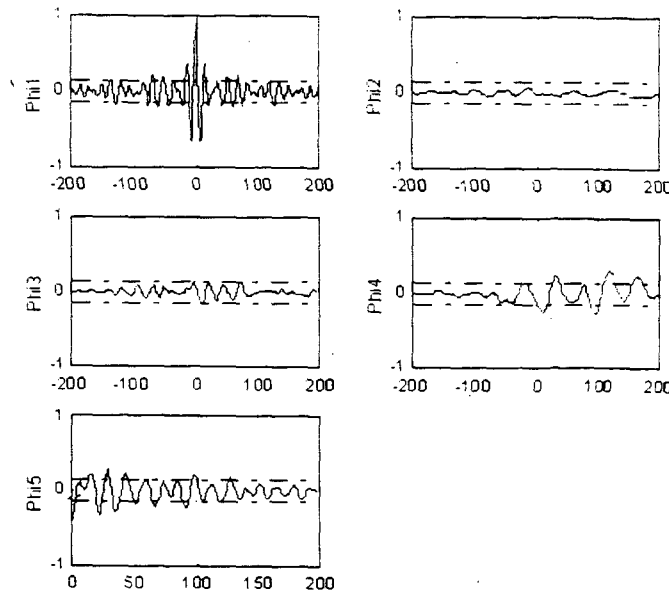


Figura 4. 35 - Estimativa das Funções Correlações Cruzadas :

$\text{Phi1} = \Phi_{\varepsilon\varepsilon}(\tau)$ ;  $\text{Phi2} = \Phi_{u\varepsilon}(\tau)$ ;  $\text{Phi3} = \Phi_{u^2\varepsilon}(\tau)$ ;  $\text{Phi4} = \Phi_{u^2\varepsilon^2}(\tau)$ ;  $\text{Phi5} = \Phi_{\varepsilon(a_1)}(\tau)$ , para o caso de ruído de medida não-linear sem utilização do sinal de erro de predição na entrada da rede neural com o modelo de Hammerstein-Wiener

Como nota-se nos resultados mostrados nas figuras acima, o uso do sinal  $\varepsilon(k)$  acrescentou benefícios à identificação, aumentando o desempenho do modelo neural. Este maior desempenho é representado pelo menor número de iterações realizado quando se utiliza o erro de predição e a adequação de todas as estimativas das funções correlações cruzadas, o que não ocorre com as funções *Phi3* e *Phi4* quando não se utiliza o sinal de erro de predição. As discrepâncias encontradas em *Phi3* e *Phi4*, quando não se tem o sinal  $\varepsilon(k)$  na entrada da rede são irrelevantes para o processo de identificação, mas importante quando se compara com o caso de utilização do erro de predição. Nestes dois casos simulados a RNA tem 5 camadas, onde

- Primeira camada apresenta um único neurônio com função ativação identidade, significando um neurônio estático, e função de saída do tipo linear com saturação (nível da saturação igual a 10). No caso do uso de

$\varepsilon(k)$ , esta camada apresenta dois neurônios (um para receber  $u(k)$  e outro para receber  $\varepsilon(k)$ ), permanecendo o resto da configuração.

- Segunda camada contem nove neurônios com função ativação identidade, significando um neurônio estático, e função de saída do tipo tangente hiperbólica.
- Terceira camada contem um único neurônio com função ativação identidade, significando um neurônio estático, e função de saída do tipo tangente hiperbólica.
- Quarta camada tem dois neurônios com função ativação do tipo equação a diferenças de primeira ordem, onde a entrada do neurônio é afetada pelos estados de todos os outros neurônios dinâmicos, conforme equação (4.15), e função de saída do tipo linear com saturação (nível da saturação igual a 10). Neste caso a matriz  $A$  é cheia equação (4.16).
- A última camada tem um único neurônio com função ativação identidade, significando um neurônio estático, e função de saída do tipo linear com saturação (nível da saturação igual a 10).

O que se conclui sobre a utilização do sinal de erro de predição  $\varepsilon(k)$ , quando se tem a situação de erro de medida não-linear na saída do processo orientado a blocos, é que o mesmo tem uma influência positiva no processo de identificação.

#### 4.4.3 Estabilidade do Modelo

A instabilidade do modelo neural com dinâmica interna proposto neste trabalho é devido aos ajustes dos parâmetros  $a_{ij}$ s das camadas dinâmicas da rede que levam os autovalores da matriz  $A$  para fora do círculo unitário do plano-Z.

O algoritmo de aprendizado desenvolvido não prevê a possibilidade de rejeição a ajustes paramétricos que torne a rede neural instável. Para os casos em que se utilizou camadas dinâmicas em que os neurônios dinâmicos lineares não são

influenciados pelos outros neurônios da mesma camada, ou seja a matriz  $A$  é diagonal, tentou-se evitar a instabilidade do modelo. Esta tentativa resumiu-se em ignorar todo ajuste que tornasse o módulo de  $a_{ii}$  maior que a unidade. Quando isto ocorresse o valor da constante de aprendizado era diminuído de um fator igual a 0.5 (testou-se vários valores entre 0.1 e 0.9). Mas a tentativa não teve sucesso, o algoritmo de aprendizado não conseguia deslocar os parâmetros para fora da região de instabilidade. A possível causa do fracasso deste procedimento foi a falta de um embasamento teórico que justificasse tal procedimento. Este tipo de procedimento é bem conhecido quando se utiliza modelos lineares, chamado de *algoritmo de projeção*. Na literatura o uso deste tipo de procedimento só existe para os modelos neurais em que o vetor de regressão é linear com os parâmetros, como por exemplos as RBF's (Feng, 1993) e RHONN's (Ruiz-Vargas, 1997), o que não acontece com o modelo aqui utilizado.

Um exemplo da situação de instabilidade do modelo neural com dinâmica interna é mostrado nas figuras a seguir, onde observa-se o comportamento dos autovalores da matriz  $A$  da camada dinâmica linear da rede neural. Esta simulação é resultado da tentativa de identificar o modelo de Wiener da seção 4.3.2, com o sinal de saída igual a  $y(k) = \text{sen}[x_1(k) + d(k)]$ , com  $d(k)$  sendo uma sequência ruído branco gaussiano. Neste exemplo não foi utilizado o sinal de erro de predição na entrada da rede neural e o sinal de entrada  $u(k)$  é uma sequência PRBS de amplitude  $\pm 1.0$  com 50 amostras coletadas. A topologia da rede é idêntica a utilizada na seção 4.3.2.

Como pode-se observar, o Autovalor1 da matriz  $A$ , da segunda camada de neurônios da rede utilizada, sai do círculo unitário do plano-Z gerando a instabilidade no modelo neural, isto ocorre na iteração de número 750 (visto na Figura 4. 36). Após o Autovalor1 sair do círculo unitário o Autovalor2 também apresenta módulo maior do que a unidade, isto ocorre na iteração de número 753 (não visto na Figura 4. 36 por problemas de escala). Como existem saturadores nas funções de saída, tanto faz ser tanh, sigmóide ou linear com saturação, a saída da rede fica confinada a valores pequenos. O mesmo ocorre com a saída de todos os neurônios da rede, causando o

fenômeno de paralisia no modelo neural, com todos os neurônios tendo seus sinais de saídas saturados.

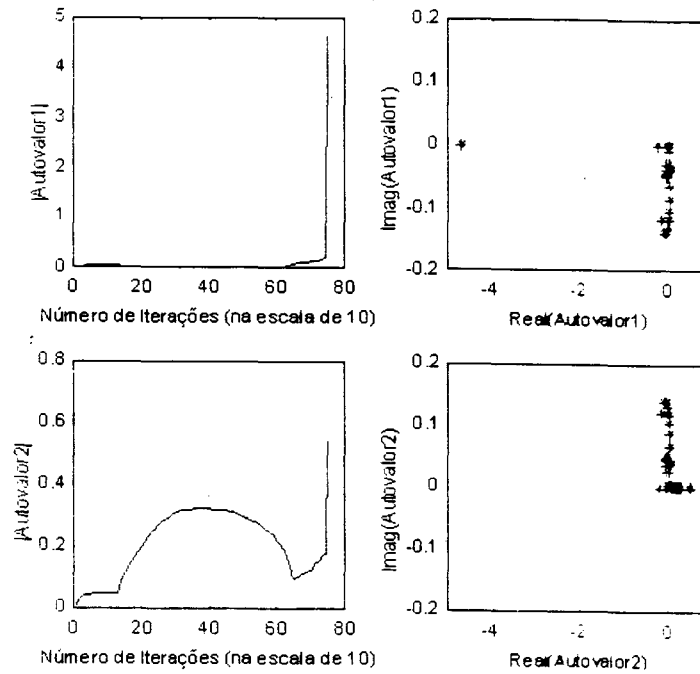


Figura 4. 36 - Comportamento dos autovalores referentes à camada dinâmica linear do modelo neural durante o treinamento para identificar o modelo de Wiener da seção 4.3.2 com um sinal de entrada  $u(k)$  do tipo PRBS e ruído não-linear na saída do processo

## 5 - Controle Feedforward Neural

### 5.1 Introdução

A teoria de controle é bem desenvolvida na área de análise e síntese para sistemas dinâmicos lineares invariantes no tempo. Entretanto no controle de sistemas dinâmicos não-lineares tem-se poucos resultados gerais disponíveis, aqui os estudos são feitos "caso-a-caso".

Na tentativa de desenvolver uma Teoria Geral de Controle vários pesquisadores procuraram utilizar as RNA's como controladores e/ou identificadores não-lineares (Antsaklis e Passino, 1993; Chen e Khalil, 1992; Elman, 1990; Jordan e Jacobs, 1990; Pao, 1990). A justificativa, para este uso, sempre foi a habilidade das RNA's em aproximar mapeamentos não-lineares arbitrários. Embora as RNA's estabeleçam uma nova abordagem para os difíceis e árduos problemas de controle dinâmico não-linear, as mesmas ainda estão longe de serem uma solução geral para o controle não-linear.

Mesmo não sendo uma solução geral, as RNA's são utilizadas com sucesso em várias aplicações específicas em que procura-se explorar, ao máximo, a sua capacidade de modelar processos dinâmicos não-lineares. Exemplos destas aplicações específicas são : controle por modelo interno (De Azevedo e Barreto, 1994; Hunt e Sbarbaro, 1991); controle feedforward (Kawato et al, 1987); controle adaptativo indireto (Narendra e Parthasarathy, 1991); etc.

Neste capítulo mostra-se a aplicação desenvolvida por Kawato e co-autores (Kawato et al, 1987) para o controle de um sistema dinâmico contínuo não-linear e para o controle de um sistema dinâmico discreto com blocos lineares dinâmicos e blocos não-lineares estáticos. A inovação desta aplicação está na utilização do novo modelo neural com dinâmica interna, desenvolvido no capítulo 2, como controlador feedforward. Antes de mostrar a aplicação, faz-se uma pequena revisão do que se convencionou chamar de controle neural.

## 5.2 Controle Neural

### 5.2.1. Motivação Biológica

Tem sido um desejo dos cientistas de sistemas criar uma máquina que possa operar com o nível de independência do sistema de controle humano em um ambiente não-estruturado e incerto. O sucesso da operação de uma máquina autônoma depende da sua habilidade em tratar com uma variedade de eventos inesperados no seu ambiente operacional. Sendo uma máquina autônoma, a mesma deve receber apenas um objetivo desejado e realizar este objetivo através de contínuas interações com o seu ambiente e com contínuas realimentações sobre suas ações. Sendo autônoma, ou inteligente, a máquina deve ser apta à aprender tarefas cognitivas de auto-nível. Além disso, deve continuamente se adaptar e realizar tarefas com aumento da eficiência, mesmo quando houver modificações imprevistas no ambiente.

Este tipo de máquina, autônoma, inteligente, ou cognitiva, deve ser utilizada onde a interação humana direta possa ser perigosa, tediosa ou impossível.

Sistemas biológicos podem ser considerados como uma fonte de motivação plausível para o desenvolvimento de máquinas autônomas. A biologia pode dar não apenas a motivação, mas também indicar caminhos para o desenvolvimento de aprendizado robusto e algoritmos de adaptação em máquinas (Rosenberg, 1998). De posse da tecnologia de controle atual, observa-se que a ausência destas habilidades de robustez e adaptação está relacionada ao fato de que o processamento da informação nos sistemas biológicos é completamente diferente do utilizado nas técnicas convencionais de controle.

Enquanto as técnicas convencionais de controle são "baseadas em modelos" (Kalman et al., 1969), no sentido de que os métodos de projeto envolvem a construção de um modelo matemático explícito do sistema dinâmico a ser controlado, os mecanismos de controle biológicos não são baseados em modelos. Estes trabalham com sucesso em condições de grande incertezas e complexidades,

e podem coordenar muitos graus de liberdade durante a execução de tarefas de movimento em um ambiente não-estruturado.

Os mecanismos de controle neural são normalmente muito complexos e de difícil, ou quase impossível, formulação matemática. Estes mecanismos solucionam tarefas complexas sem ter de desenvolver um modelo matemático da tarefa e do ambiente onde vai ser realizada, e sem resolver qualquer equação diferencial, integral ou outras operações complexas (Gupta e Rao, 1994).

Gupta e Rao (Gupta e Rao, 1994) afirmam que,

*“Se os princípios fundamentais da computação neural utilizados pelos sistemas de controle biológico forem bem entendidos, parece provável que uma geração de metodologias de controle, totalmente nova, possa ser desenvolvida, sendo mais robustas e inteligentes, muito além da capacidade das tecnologias atuais, que são baseadas na modelagem matemática explícita.”*

Sendo os modelos de RNA's uma classe de sistemas que servem para resolver problemas de controle muito complexos, o maior desafio é entender as complexas funções dos sistemas neurais biológicos e usar este conhecimento para construir uma RNA.

Como a idéia principal é utilizar as RNA's para controlar sistemas complexos, que são difíceis de serem modelados, espera-se que os mecanismos biológicos sejam simples o suficiente para que não surja a questão de se procurar uma solução bem mais complexa que o próprio problema a ser solucionado. Por serem sistemas criados pela natureza, as redes neurais biológicas, até o nível de compreensão atual, parecem apresentar mecanismos básicos de funcionamento razoavelmente simples, supondo-se que toda sua força computacional esteja nas interações entre os seus  $10^{11}$  neurônios.

### 5.2.2. Controle Convencional

Os métodos de projeto convencional para sistemas de controle, envolvem a construção de um modelo matemático que descreva o comportamento dinâmico do sistema a ser controlado, e a aplicação de técnicas analíticas, neste modelo, de forma a gerar uma lei de controle. Normalmente, o modelo matemático consiste de um conjunto de equações diferenciais, ou a diferenças, lineares ou não-lineares, muitas das quais são encontradas a partir de algumas formas de aproximação e simplificação. Estas técnicas convencionais fracassam quando o sistema em questão é de difícil representação, dificuldades estas geradas por incertezas ou por pura complexidade. Diferentemente, o operador humano nem sempre trata o problema de controle com um modelo matemático detalhado do sistema, mas sim com um sentimento qualitativo do processo, e com conhecimento e raciocínio aproximados do procedimento de controle. Apesar da dificuldade de controlar processos, os quais não se conhece adequadamente, o controle convencional apresenta técnicas que buscam uma solução para esta situação.

Existem duas abordagens de controle, normalmente descritas na literatura, para tratar a questão do desempenho do controlador, quando a dinâmica do processo é desconhecida (Gupta e Rao, 1994) :

- Controle Robusto
- Controle Adaptativo

No controle robusto, se o sistema físico é um membro de uma classe de sistemas que está próximo de um dado processo nominal, um controlador robusto garante que o controla e o estabiliza. Uma vez que é esperado que um controlador fixo controle e estabilize todos os sistemas pertencentes a um conjunto, o preço a pagar pode ser que o controlador projetado seja mais complexo quando comparado com a complexidade necessária para estabilizar um único processo desta classe. O controle adaptativo apresenta parâmetros do controlador variáveis que são ajustados, a cada iteração, no sentido de respeitar um dado critério de desempenho. Em geral, a abordagem adaptativa é aplicada a uma ampla faixa de incertezas, mas



os controladores robustos são mais simples de serem implementados e não apresentam ajustes temporais nos parâmetros do controlador quando ocorrem variações no processo. Descrições detalhadas das técnicas de controle robusto e controle adaptativo podem ser encontradas em Gupta (Gupta, 1986), Åström (Åström e Wittenmark, 1995), Narendra (Narendra e Anaswamy, 1989), Dorato (Dorato e Yedavalli, 1990) e Abdallah (Abdallah et al., 1991). Dentre as duas técnicas citadas, a que mais se aproxima do procedimento de aprendizado em RNA's é o controle adaptativo.

Uma representação esquemática de um sistema de controle adaptativo geral é mostrado na Figura 5.1. Um sistema de controle adaptativo mede certos índices de desempenho utilizando as entradas, os estados e as saídas do sistema dinâmico sobre controle. A partir da comparação dos valores do índice de desempenho medido com o desejado, o mecanismo de adaptação (regra adaptativa) modifica os parâmetros do controlador de forma a manter a resposta do processo próxima da resposta desejada.

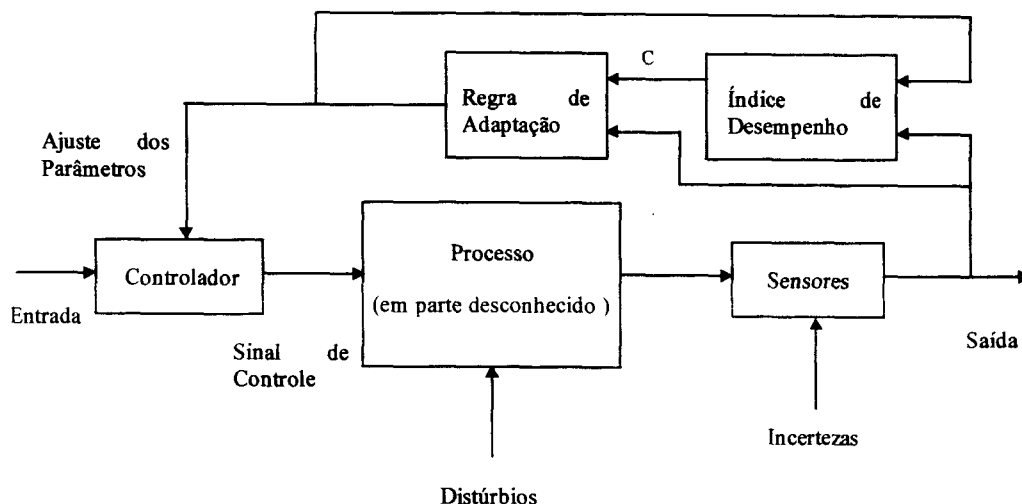


Figura 5. 1 - Configuração básica de um sistema adaptativo

Para garantir a estabilidade do sistema de controle adaptativo é necessário algum conhecimento "a priori" sobre o processo a ser controlado. Este conhecimento "a priori", também é necessário para implementar os identificadores ou observadores que têm a mesma ordem do processo (Gupta e Rao, 1994). Uma vez que para sistemas complexos do mundo real, a ordem do mesmo pode ser desconhecida, ou

então grande, as implementações de controle adaptativo para estes casos podem se tornar difíceis ou então impossíveis, no sentido de estabilidade absoluta e relativa (Ortega e Tang, 1989).

A necessidade de controlar sistemas complexos que apresentem significantes incertezas, tem levado a uma reavaliação das metodologias de controle existentes. A evolução no paradigma de controle tem sido direcionada por duas grandes preocupações (Gupta e Rao, 1994) : a necessidade de trabalhar com sistemas cada vez mais complexos, e o requisito de alcançar as, cada vez, mais exigentes necessidades de projeto com menos conhecimento preciso, sobre o processo e o seu ambiente. Nestas situações, é quase obrigatório que o esquema de controle apresente características de aprendizado e/ou adaptação.

### 5.2.3. Controlador Neural

Ao trabalhar com processos dinâmicos que apresentam incertezas, o controlador tem que estimar as informações desconhecidas durante a sua operação. Se esta estimativa da informação, gradualmente aproxima-se da informação verdadeira a medida que o tempo aumenta, então o controlador assim projetado, aproxima-se de um desempenho ótimo. Por causa do aumento gradual de desempenho, devido a melhoria da estimativa da informação, este controlador pode ser visto como um controlador adaptativo com aprendizado. O controlador aprende as informações desconhecidas durante sua operação, e esta informação, por sua vez, é utilizada como uma experiência para futuras decisões e controles (Fu, 1970).

A Figura 5.2 representa um tipo de esquema de controle e aprendizado neural. Um sistema de controle é denominado de um "Sistema de Controle com Aprendizado" se as informações relativas às, características desconhecidas do processo ou do ambiente, são adquiridas durante a operação, e a informação obtida é utilizada em futuras estimações, reconhecimentos, classificações, controles ou decisões, tal que o desempenho global do sistema melhore.

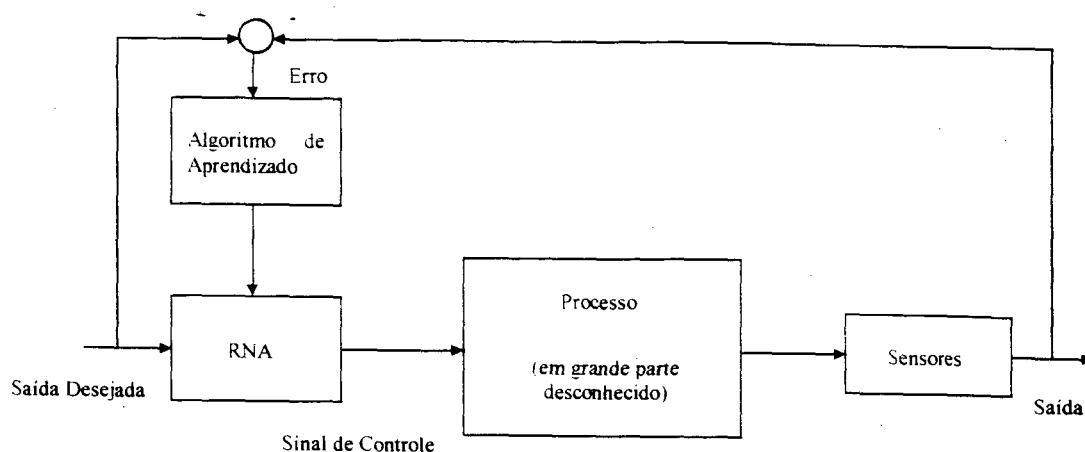


Figura 5. 2 - Um típico esquema de controle neural com aprendizado

Com o uso do aprendizado no controlador, expande-se de maneira eficaz a região de operação do controlador, podendo-se criar controladores mais robustos que possam levar a criação de uma classe de controladores verdadeiramente autônomos. Estas leis de controle não necessitam ser diretamente explicitadas, já que pode ser feito um aprendizado baseado em exemplos. Devido ao aprendizado por exemplos, o sistema de controle pode compensar um grande número de mudanças nas condições do processo e de seu ambiente (desde que estas situações tenham sido treinadas, ou então, escolhidas de tal forma que o controlador possa generalizar para um conjunto de situações, onde algumas foram treinadas e outras não). Estas mudanças de ponto de operação podem exceder as tolerâncias de projeto de um sistema adaptativo convencional. Isto ocorre porque não existe uma solução analítica geral para sistemas que são inerentemente complexos, não-lineares e incertos. O controlador com aprendizado determina os valores de seus parâmetros de forma a ter um desempenho ótimo para as condições de operação treinadas, independente das mesmas acarretarem ou não modelos matemáticos explícitos.

Ambos os sistemas de controle adaptativo e com aprendizado podem ser implementados utilizando algoritmos de ajuste de parâmetros, e os dois fazem uso de informações de desempenho realimentadas (Farrell et al., 1993). A diferença

entre sistema adaptativo e com aprendizado, reside no fato que este trata cada ponto de operação distinto como novo, enquanto aquele correlata a experiência passada com a situação presente e, por conseguinte, modifica seu comportamento (Gupta e Rao, 1994). Uma vez que um sistema com aprendizado é capaz de modificar suas ações, ele é também um sistema adaptativo.

A necessidade da característica de aprendizado no controle de sistemas complexos, operando na presença de incertezas significantes, tem levado ao desenvolvimento de novas técnicas de controle, bastante diferentes das usuais. O uso de RNA's em sistemas de controle, ou simplesmente controle neural, pode ser visto como um passo natural na evolução das metodologias de controle (Kawato et al., 1987; Nguyen e Widrow, 1990). A RNA, com seu maciço paralelismo e sua habilidade de aprender, oferece várias e excitantes possibilidades para a criação de técnicas de controle superiores as atuais, em situações que envolvam sistemas complexos. Entretanto, deve-se notar que sistemas de controle baseados em RNA's não são uma panacéia, e as mesmas não apresentam soluções para todos os problemas.

#### 5.2.4. Controle Não-Linear

As RNA's têm mostrado grande potencial de uso no domínio dos problemas de controle não-linear. Enquanto muitos avanços têm sido feitos no projeto de controladores adaptativos para sistemas lineares com parâmetros desconhecidos, estes mesmos controladores não podem ser utilizados para o controle global de sistemas não-lineares, exceto nos casos onde existe a linearização em torno de um ponto de operação (Coelho, 1991; Barreiros, 1995). Para controle não-linear, existe um conjunto de métodos convencionais, tais como plano de fase, função descritiva e linearização por realimentação, para a análise e síntese de controladores, sendo que os mesmos foram desenvolvidos para classes específicas de sistemas não-lineares. Os métodos de controle não-linear existentes são para sistemas específicos, em outras palavras, uma metodologia de controle designada para uma classe de sistemas não-lineares pode ser completamente inaceitável para uma outra classe.

A mais significativa característica das RNA's é a sua habilidade em aproximar funções não-lineares arbitrarias. Esta habilidade das RNA's tem feito das mesmas um modelo para sistemas não-lineares, mesmo que não preserve nenhuma estrutura usual (seção 3.6), que é de primordial importância na síntese de controladores não-lineares (Hunt et al., 1992). Um controlador neural, em geral, realiza uma forma específica de controle adaptativo, com o controlador tendo a forma de uma RNA multicamadas e os parâmetros adaptáveis sendo definidos como pesos ajustáveis. Em geral, RNA's representam estruturas paralelas de processamento de informações, que fazem das mesmas candidatas, em primeira mão, para o uso em sistemas de controle multivariáveis. A abordagem por RNA's define o problema de controle como o mapeamento de sinais medidos por "mudanças" em ações de controle, calculadas como mostrado na Figura 5.3.

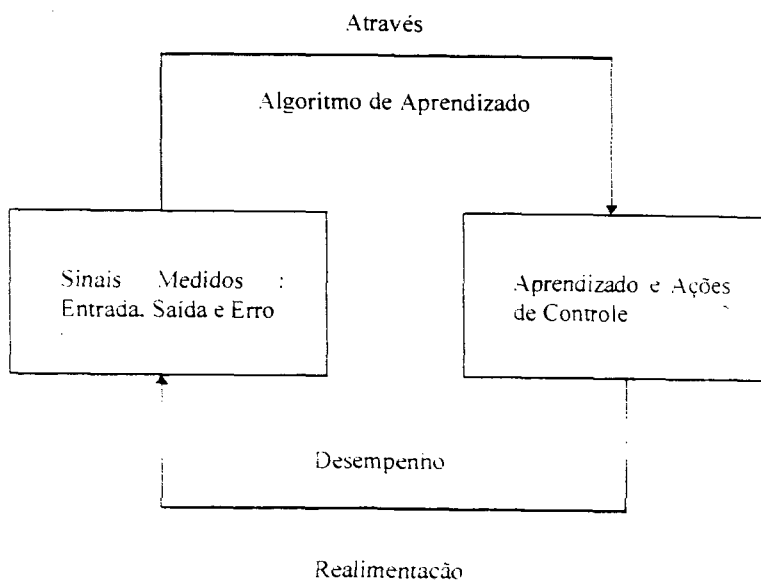


Figura 5.3 - Representação do aprendizado e ações de controle em uma abordagem por RNA's com o mapeamento dos sinais medidos para o espaço de controle feito pelo algoritmo de aprendizado

Hunt e co-autores (Hunt et al., 1992) apontaram as seguintes características das RNA's, como sendo as mais importantes para as aplicações em controle não-linear :

1. RNA's têm, através de teoremas matemáticos (Capítulo 3), a habilidade de aproximar mapeamentos não-lineares arbitrários. Existe também a possibilidade de uma aproximação por RNA's ser mais parcimoniosa, ou seja, esta aproximação requer menos parâmetros, que outras técnicas, tais como polinômios ortogonais, série de Fourier, etc. Entretanto, isto ainda tem que ser provado teoricamente (Sontag, 1993).
2. Uma vez que RNA's podem ter múltiplas entradas e saídas, elas podem ser utilizadas, naturalmente, para controle de sistemas multivariáveis.
3. RNA's podem ser treinadas *off-line*, utilizando a coleta de dados passados do sistema a ser controlado, ou podem ser adaptadas *on-line* de forma a compensar possíveis mudanças que ocorram no sistema.
4. Uma vez que RNA's são dispositivos de processamento paralelo de informações, podem ser implementadas em hardware paralelo. Deste modo, como uma consequência da grande rapidez do processamento de sinais no hardware atual, RNA's podem ser utilizadas em controle de tempo-real. Também devido a sua organização distribuída, RNA's têm a possibilidade de oferecer, quando treinada apropriadamente, um bom nível de tolerância a falha, mesmo com perda interna de neurônios ou pesos (Nascimento Jr., 1994).
5. Utilizando RNA's, é possível realizar uma eficaz fusão de dados recebidos de vários sensores, que são informações simbólicas e numéricas, integrando-os de uma forma natural. De maneira similar a fusão de dados de sensores, a RNA pode realizar uma integração de atuadores, onde vários atuadores agem em diferentes sistemas, sem uma correspondência um-a-um entre atuador e sistema (isto é, cada atuador produz sinais para vários sistemas e cada sistema recebe sinais de vários atuadores). Com esta característica, um controlador neural pode ser robusto à perda de sensores e atuadores.

Se for possível obter todas estas características simultaneamente, o resultado será impressionante : um controlador em tempo real não-linear multivariável adaptativo (Nascimento Jr., 1994). A natureza, através da evolução, produziu controladores deste tipo nos sistemas biológicos. Obviamente, os modelos de RNA's e de sistemas de controle não-neural estão longe de apresentar este nível de refinamento.

### 5.3 Controle Feedforward Neural

Basicamente as RNA's são utilizadas no controle não-linear como modelos diretos ou modelos indiretos do processo a ser controlado. A estratégia de controle utilizada neste capítulo apresenta a RNA com dinâmica interna, desenvolvida no Capítulo 2, funcionando como um modelo inverso do processo não-linear.

Uma das mais simples e elegantes arquiteturas de controle neural utilizando modelo inverso é a arquitetura proposta por Kawato e co-autores (Kawato et al., 1987), chamada originalmente de *Feedback-Error-Learning* (ou método de aprendizado por erro de realimentação). Em 1987, Kawato e co-autores (Kawato et al., 1987; Miyamoto et al., 1988) propuseram o método de aprendizado através do erro de realimentação para treinar uma RNA de forma a realizar o controle dinâmico de um robô. A maior motivação destes pesquisadores foi propor um modelo computacional próximo ao utilizado no sistema nervoso central para o aprendizado motor. A idéia básica é combinar um controlador de realimentação de erro de saída convencional, sempre disponível, sintonizado precariamente com uma RNA atuando como um controlador *feedforward*. O controlador de realimentação deve, pelo menos, ser suficiente para estabilizar o sistema em malha-fechada, quando utilizado sozinho, sem a RNA, mas o mesmo não precisa ser sintonizado de maneira ótima.

Embora, originalmente, Kawato e co-autores imaginassem a ação conjunta do controlador de realimentação do sinal de erro da saída do processo com o controlador *feedforward* neural, mostrou-se mais tarde que esta estratégia tenta eliminar a influência daquele em favor deste.

O objetivo é adaptar a RNA de forma a minimizar o erro de rastreamento,  $\|e\|$ , definido como a diferença entre um sinal de referência,  $y_d$ , e a saída medida do processo,  $y$ , que normalmente é um subconjunto do vetor de estado  $X$ . Para alcançar este objetivo, Kawato (Miyamoto et al., 1988) propôs o uso da saída do controlador de realimentação como o sinal de erro que ajusta os pesos da rede. Diferentemente do imaginado no trabalho original de Kawato e co-autores, este método pode ser interpretado como uma forma de minimizar o valor médio quadrático do sinal de saída do controlador de realimentação e não como a minimização do sinal  $\|e\|$  (Nascimento Jr., 1994).

O algoritmo de aprendizado minimiza a função custo

$$E[u_f(t)] = \sqrt{\frac{1}{NPT} \sum_{t=1}^{NPT} u_f^2(t)} \quad (5.1)$$

onde NPT é o número de pontos treinados ou número de amostras. Esta função equivale a função EMQ do Capítulo 2 (Equação 2.69).

É importante notar que ao se utilizar o sinal de erro de realimentação,  $e = y_d - y$ , como o sinal de ajuste dos pesos da RNA, o problema de retro-propagar o erro medido na saída do processo através do mesmo, até a saída da RNA (ou através de um modelo do processo) é evitado (Miyamoto et al., 1988). Além disto a RNA pode ser treinada *on-line*, e o método de treinamento é orientado à meta visto que, quando o erro de realimentação é zero, a saída do controlador de realimentação irá também ser zero (na realidade, se existir uma componente integral no controlador de realimentação, a sua saída pode ser uma constante diferente de zero, neste caso um termo de *bias*, de forma linear, deve ser utilizado no sinal de saída da RNA, para cancelar este termo constante na saída do controlador de realimentação (Nascimento Jr., 1994)).

Espera-se que a medida que transcorra o treinamento da rede, a RNA irá, suavemente, retirar a ação de controle do controlador de realimentação e ao mesmo



tempo melhorar o desempenho do controle global (De Oliveira, 1991). Desta maneira a RNA está sendo treinada para representar o modelo dinâmico inverso do processo.

A seguir será apresentada esta arquitetura de controle, com as modificações feitas, no esquema original de Kawato, por De Oliveira (De Oliveira, 1991; De Oliveira et al., 1991) e Nascimento Jr. (Nascimento Jr., 1994).

### 5.3.1. Estrutura do Controlador Neural Feedforward

A arquitetura de controle neural com aprendizado através do erro de realimentação é mostrada na Figura 5. 4. As modificações feitas na arquitetura original de Kawato (Kawato et al., 1987) foram :

1. Utilização de um conjunto de sinais atrasados na entrada da RNA, no lugar dos sinais  $\frac{dy_d(t)}{dt}, \frac{d^2y_d(t)}{dt^2}, \dots$  (De Oliveira, 1991).
2. Utilização de uma RNA Direta Multicamadas, no lugar da RNA de uma única camada (De Oliveira, 1991).
3. Utilização do modelo de referência para gerar o sinal de saída desejado (De Oliveira, 1991).
4. Utilização do sinal  $y_d(t-q)$  para gerar o sinal  $e(t)$ , no lugar do sinal  $y_d(t)$  (Nascimento Jr., 1994).
5. Utilização de um controlador PID (ou PD, ou PI) como controlador de realimentação, no lugar do controlador proporcional (Nascimento Jr., 1994).

As modificações introduzidas tornaram esta arquitetura mais geral, possibilitando o seu uso em tempo-real com pouco conhecimento sobre o processo a ser controlado.

A primeira modificação implica na necessidade do processo controlado ser observável isto é, a partir da trajetória de saída se possa determinar seus estados, além de facilitar a sua utilização em tempo real. Neste trabalho não será necessário

a utilização deste banco de atrasos na entrada da rede neural porque utiliza-se um modelo neural com dinâmica interna.

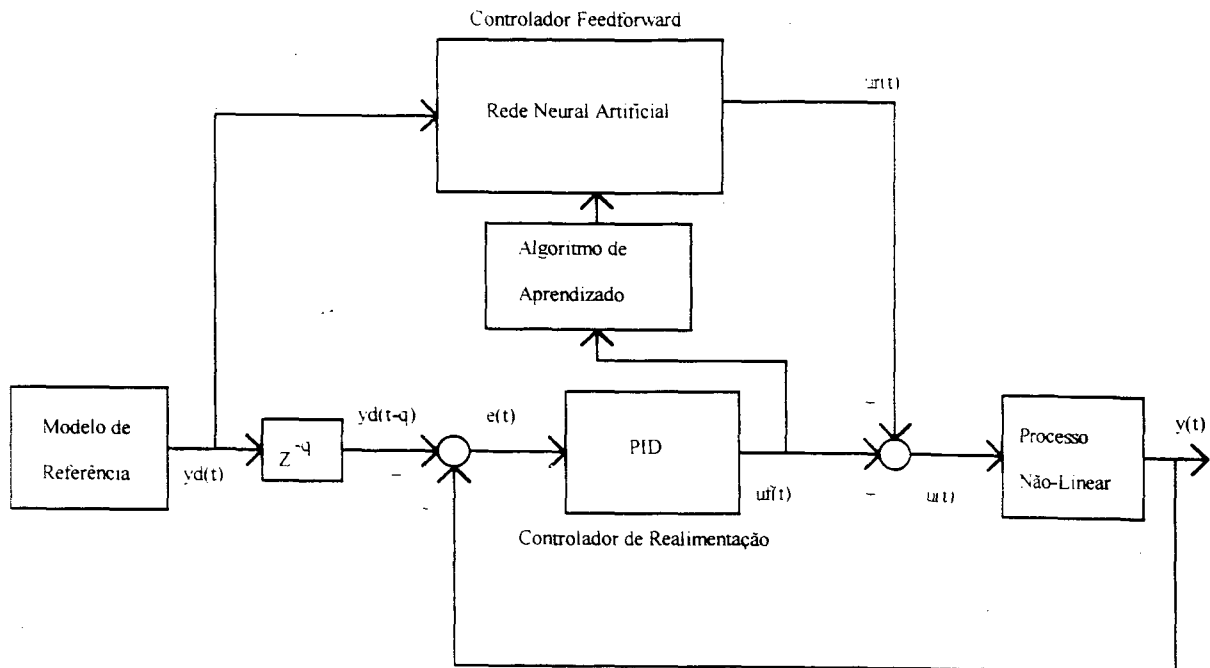


Figura 5. 4 - Arquitetura de controle neural com aprendizado através do erro de realimentação

A segunda modificação torna o controlador neural um controlador geral, uma vez que o mesmo pode implementar um mapeamento não-linear arbitrário, sempre tendo em vista a exigência de existir um modelo inverso para o processo.

A terceira modificação torna mais fácil a medida de desempenho do controlador neural, já que com o uso do modelo de referência pode-se antecipadamente definir os valores de tempo de subida, sobresinal, tempo de acomodação, etc, desejáveis na saída do processo.

A quarta modificação torna mais fácil a tarefa do controlador neural, já que o mesmo necessita implementar uma aproximação do modelo inverso atrasado do processo. Sem o atraso  $q$ , a RNA deverá implementar o modelo inverso do processo, que é uma tarefa mais difícil, uma vez que a rede deverá atuar como um preditor (Nascimento Jr., 1994). O uso de um atraso no sinal de referência foi proposto

primeiramente por Widrow e Stearns (Widrow e Stearns, 1985) para uma arquitetura de controle não-linear. Como em muitas aplicações, um modelo inverso atrasado é aceitável, o uso do sinal de referência atrasado não é uma grande limitação (Nascimento Jr., 1994; Widrow e Stearns, 1985).

A quinta modificação dá ao controlador de realimentação uma maior flexibilidade para estabilizar o sistema em malha-fechada, permitindo assim o treinamento da rede. O uso de um controlador com dinâmica, diferente do ganho proporcional utilizado no trabalho original de Kawato, é que ajudou a mostrar o fato de que o aprendizado da RNA minimiza o sinal  $u_k(t)$ .

Uma análise matemática desta estrutura de controle encontra-se no trabalho de Nascimento Jr. (Nascimento Jr., 1994), onde é mostrado que, para um sistema linear invariante no tempo e estável em malha-aberta, é possível, para a minimização do sinal de saída do controlador de realimentação, encontrar analiticamente os valores dos pesos de uma RNA linear com apenas um neurônio em uma única camada. Este trabalho também mostra o desenvolvimento matemático necessário para encontrar a equação de atualização dos pesos da RNA linear de uma única camada. Este desenvolvimento não é apresentado nos trabalhos de Kawato (Kawato et al., 1987; Miyamoto, 1988), mas apenas sugerido com base em informações fisiológicas sobre a plasticidade dos neurônios biológicos.

Um ponto de maior interesse em aplicações de redes neurais em controle é o tipo de sinal de entrada para a rede neural. Esta questão é abordada em alguns trabalhos (Neto et al., 1998; Narendra e Levin, 1995; Kurdila et al., 1995), mas os mesmos ainda são incipientes, permanecendo este estudo como uma área do controle neural ainda em aberto.

A seguir apresenta-se algumas simulações que mostram a utilidade do novo modelo neural com dinâmica interna desenvolvido em capítulos anteriores, na tarefa de controlar, segundo a estratégia de Kawato, processos não-lineares contínuos ou discretos, orientados à blocos ou não.

## 5.4 Exemplos Simulados

### 5.4.1. Sistema Não-Linear Contínuo

Nesta seção é utilizado a rede neural para controlar, segundo a descrição da Figura 5. 4, um processo não-linear contínuo de terceira ordem. O processo é descrito pelas equações,

$$\frac{dx_1(t)}{dt} = x_2(t) \quad (5. 2)$$

$$\frac{dx_2(t)}{dt} = x_3(t)$$

$$\frac{dx_3(t)}{dt} = -1.25\text{sen}(x_1(t)) - 0.15\text{cos}(x_1(t))x_2(t) - 10x_3(t) + 200u(t)$$

$$y(t) = 10x_1(t) \quad (5. 3)$$

O sinal de saída desejado,  $y_d(t)$ , é gerado pela equação (5.4), com  $a=2$ ,  $b=-2$  e  $c=4$ .

$$y_d(t) = \frac{\pi}{4} \left[ a + b \cos\left(\frac{2\pi t}{c}\right) \right] \quad (5. 4)$$

A implementação do controlador de realimentação PID digital foi conseguida através da equação (5.5), com  $k_p = 0.001$ ,  $k_i = 0.0$  e  $k_d = 0.02$ , sintonizados através de tentativa e erro, com a única preocupação de tornar o sistema em malha-fechada estável, permitindo, assim, o treinamento do controlador neural feedforward. Na equação (5.5) a variável  $e_s$  representa o valor acumulado do sinal de erro, necessário para efetuar a ação de controle integral do controlador PID, e a variável  $e_d$  equivale ao sinal de erro anterior, o qual é utilizado para realizar a ação de controle derivativa do PID.

$$e_s = e_s + e$$

$$u = k_p e + (k_i T) e_s + \left(\frac{k_d}{T}\right)(e - e_d) \quad (5. 5)$$

$$e_d = e$$

O conjunto de treino é formado por 100 amostras, para um período de amostragem igual a 0.08 segundos. O atraso na referência,  $q$ , é igual a 2. O

processo contínuo foi simulado com a utilização do algoritmo de Runge-Kutta de quarta ordem.

A RNA utilizada é formada por 4 camadas de neurônios, com todas as camadas contendo um *bias*, exceto na primeira camada, onde :

- A primeira camada apresenta um único neurônio com função ativação identidade, significando um neurônio estático, e função de saída do tipo tangente hiperbólica.
- A segunda camada contém quatro neurônios com função ativação identidade, significando neurônios estáticos, e função de saída do tipo tangente hiperbólica.
- A terceira camada tem três neurônios com função ativação do tipo equação a diferenças de primeira ordem, onde a entrada do neurônio é afetada pelos estados de todos os outros neurônios, inclusive dele próprio, conforme equação (5.6), e função de saída do tipo tangente hiperbólica. Neste caso a matriz  $A$  é cheia, como mostra a equação (5.7).

$$x_1(k+1) = a_{11}x_1(k) + a_{12}x_2(k) + a_{13}x_3(k) + net_1(k) \quad (5.6)$$

$$x_2(k+1) = a_{21}x_1(k) + a_{22}x_2(k) + a_{23}x_3(k) + net_2(k)$$

$$x_3(k+1) = a_{31}x_1(k) + a_{32}x_2(k) + a_{33}x_3(k) + net_3(k)$$

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \quad (5.7)$$

- A última camada tem um único neurônio com função ativação identidade, significando um neurônio estático, e função de saída do tipo linear com saturação (nível de saturação igual a 100).

A Figura 5. 5 mostra o comportamento da saída do processo no início do treinamento e a Figura 5. 6 os sinais de controle  $u_1(k)$  e  $u_2(k)$ .

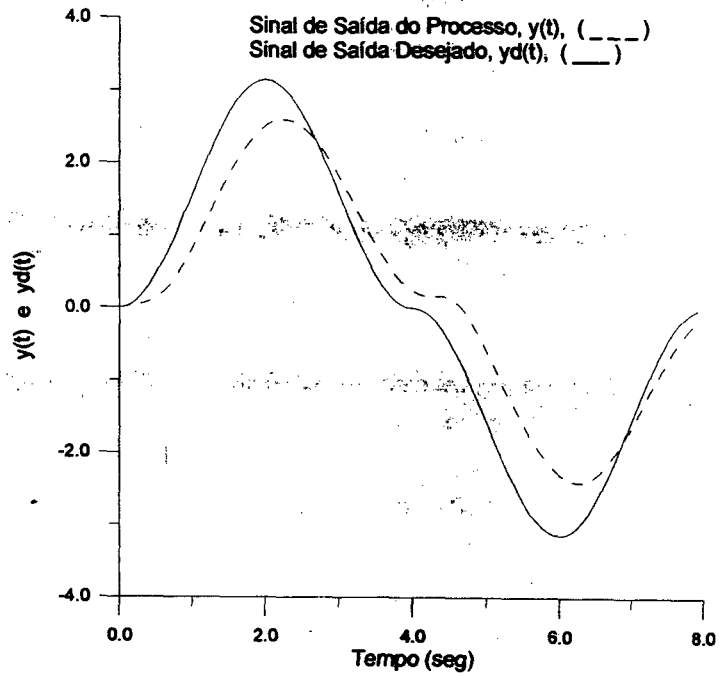


Figura 5. 5 - Comportamento da saída do processo no início do treinamento

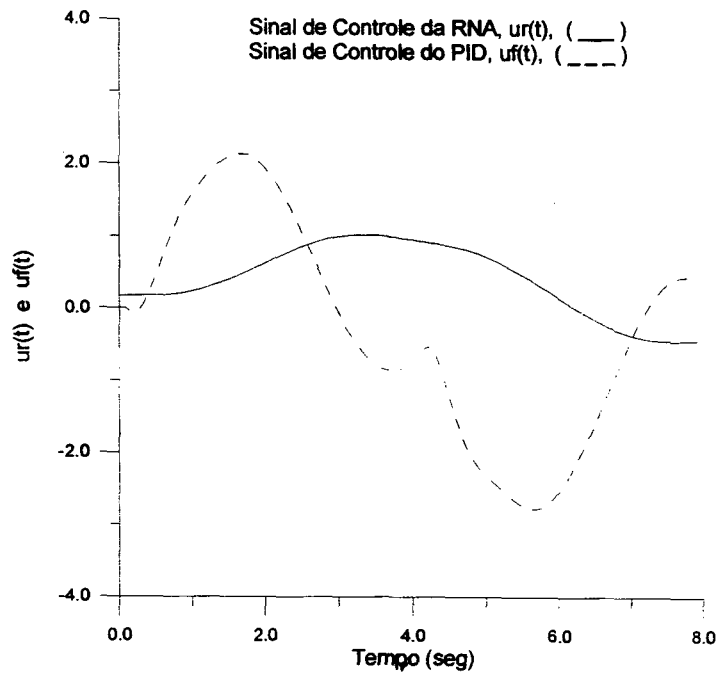


Figura 5. 6 - Sinais de controle no início do aprendizado

Após 72582 iterações, interrompe-se a sessão de treinamento com o comportamento do sinal de saída do processo sendo mostrado na Figura 5.7. A Figura 5.8 mostra os sinais de controle  $u_r(k)$  e  $u_f(k)$  após o término do aprendizado.

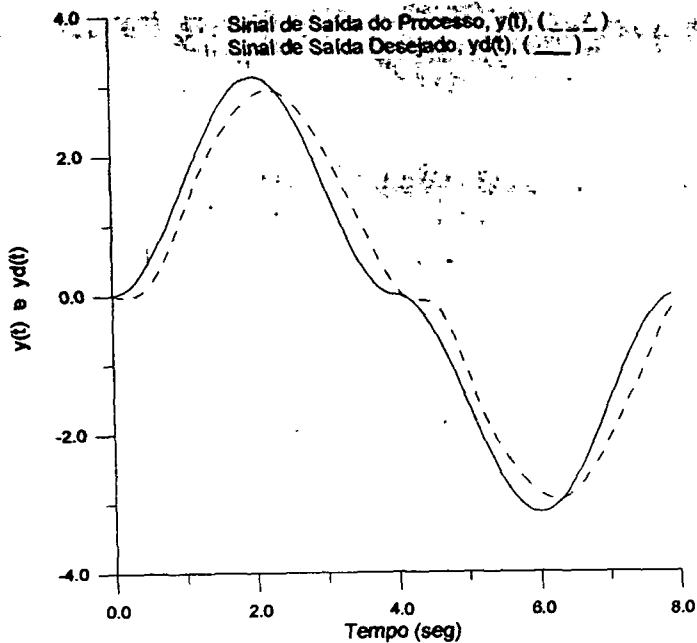


Figura 5.7 - Comportamento da saída do processo após o treinamento

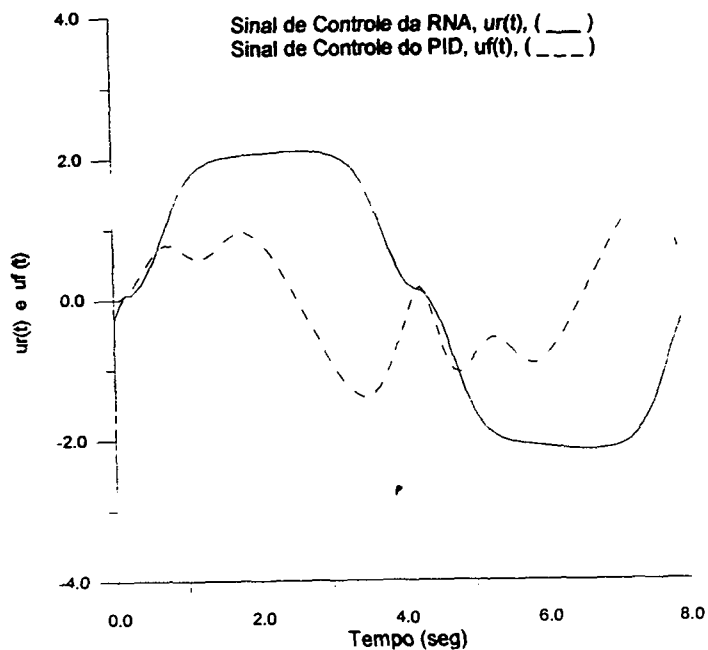


Figura 5.8 - Sinais de controle após o treinamento

A Figura 5. 7 mostra o sinal de saída do processo atrasado de dois períodos de amostragem. Este atraso é o valor de  $q$  escolhido para esta simulação, que gera o sinal  $y_d(t-q)$  a ser seguido pelo processo. Existe alguma diferença entre o sinal de saída desejado para o processo e o sinal de saída da planta, mas precisamente na amplitude dos sinais. Este fato pode ser consequência do uso da banda integral do controlador PID igual a zero ( $k_i=0$ ).

Como já citado, e mostrado, nos capítulos anteriores o modelo neural utilizado apresenta situações onde ocorre a instabilidade do mesmo. As figuras a seguir mostram uma destas situações. Neste caso fez-se uma outra simulação para o processo descrito anteriormente, com novos pesos iniciais no controlador neural, permanecendo a mesma topologia da rede.

A Figura 5. 9 mostra a evolução da função custo  $E[e(t)]$  em quase 200000 iterações.

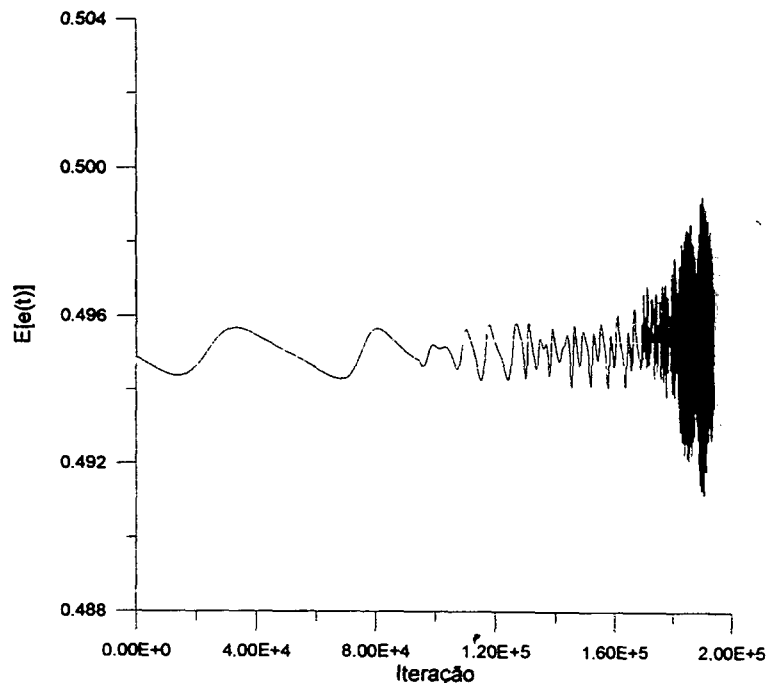


Figura 5. 9 - Evolução da função  $E[e(t)]$  com instabilidade do modelo neural



Embora o algoritmo de aprendizado minimize a função  $E[u_A(t)]$  preferiu-se medir a função  $E[e(t)]$  porque a mesma informa o quanto a saída do processo está perto da saída desejada. A Figura 5. 10 mostra o comportamento dos autovalores da matriz  $A$ , correspondente aos neurônios dinâmicos da segunda camada de neurônios do controlador *feedforward*.

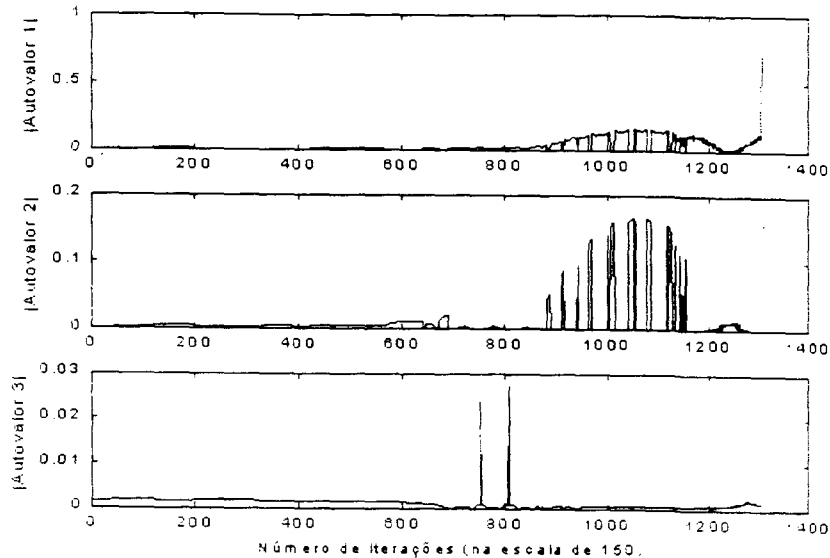


Figura 5. 10 - Comportamento dos autovalores da matriz  $A$  quando ocorre a instabilidade do modelo neural

A instabilidade da rede é causada pelo *autovalor1* que sai do interior do círculo unitário do plano- $Z$ , em torno de 195000 iterações. Este fato não aparece na Figura 5. 10 por problemas de escala, mas pode-se observar a tendência do módulo do *autovalor1* de ficar maior que a unidade.

#### 5.4.2. Sistema Não-Linear Discreto Orientado à Blocos

Nesta seção é utilizada a rede neural para controlar, segundo a descrição da Figura 5. 4, um processo não-linear discreto orientado à blocos de segunda ordem, que segue a descrição dos modelos de Hammerstein apresentados no capítulo anterior. O processo é descrito pelas equações.

$$x(k+1) = \begin{bmatrix} 0.8 & -0.15 \\ 1 & 0 \end{bmatrix} x(k) + \begin{bmatrix} 1 \\ 0 \end{bmatrix} f[u(k)] \quad (5.8)$$

$$y(t) = [1.4 \quad -0.07]x(t) + f[u(k)] \quad (5.9)$$

$$f[u(k)] = \text{sen}(u^2(k)) \quad (5.10)$$

O sinal de saída desejado,  $y_d(t)$ , é idêntico ao caso simulado na seção 5.4.1 (equação (5.4)). A implementação do controlador de realimentação PID digital foi conseguida através da equação (5.5), com  $k_p = 0.2$ ,  $k_i = 0.0$  e  $k_d = 0.001$ , sintonizados através de tentativa e erro, com a única preocupação de tornar o sistema em malha-fechada estável, permitindo assim o treinamento do controlador neural feedforward.

O conjunto de treino é formado por 50 amostras. O atraso na referência,  $q$ , é igual a 2.

A RNA utilizada é formada por 5 camadas de neurônios, com todas as camadas contendo um *bias*, exceto na primeira camada, onde :

- A primeira camada apresenta um único neurônio com função ativação identidade, significando um neurônio estático, e função de saída do tipo linear com saturação (saturação igual a 100).
- A segunda camada tem dois neurônios com função ativação do tipo equação a diferenças de primeira ordem, onde a entrada do neurônio é afetada pelos estados de todos os outros neurônios, inclusive dele próprio, conforme equação (5.11), e função de saída do tipo linear com saturação (saturação igual a 100).

Neste caso a matriz  $A$  é cheia, como mostra a equação (5.12).

$$\begin{aligned} x_1(k+1) &= a_{11}x_1(k) + a_{12}x_2(k) + ne_{1}(k) \\ x_2(k+1) &= a_{21}x_1(k) + a_{22}x_2(k) + ne_{2}(k) \end{aligned} \quad (5.11)$$

$$A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \quad (5.12)$$

- A terceira camada contém um neurônio com função ativação identidade, significando neurônio estático, e função de saída do tipo tangente hiperbólica.

- A quarta camada contém quinze neurônios com função ativação identidade, significando neurônios estáticos, e função de saída do tipo tangente hiperbólica.
- A última camada tem um único neurônio com função ativação identidade, significando um neurônio estático, e função de saída do tipo linear com saturação (nível de saturação igual a 100).

A Figura 5. 11 mostra o comportamento da saída do processo no início do treinamento e a Figura 5. 12 os sinais de controle  $u_r(k)$  e  $u_i(k)$ .

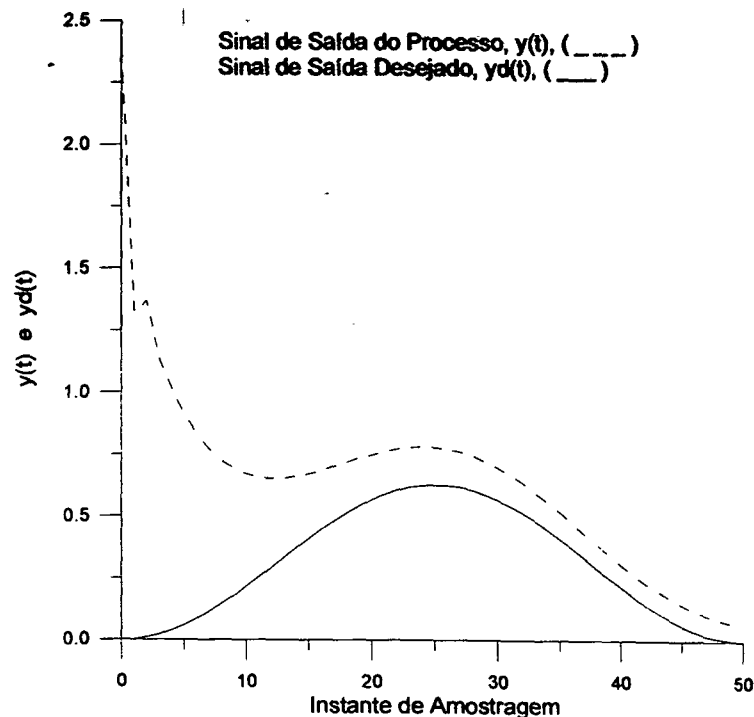


Figura 5. 11 - Comportamento da saída do processo no início do treinamento --

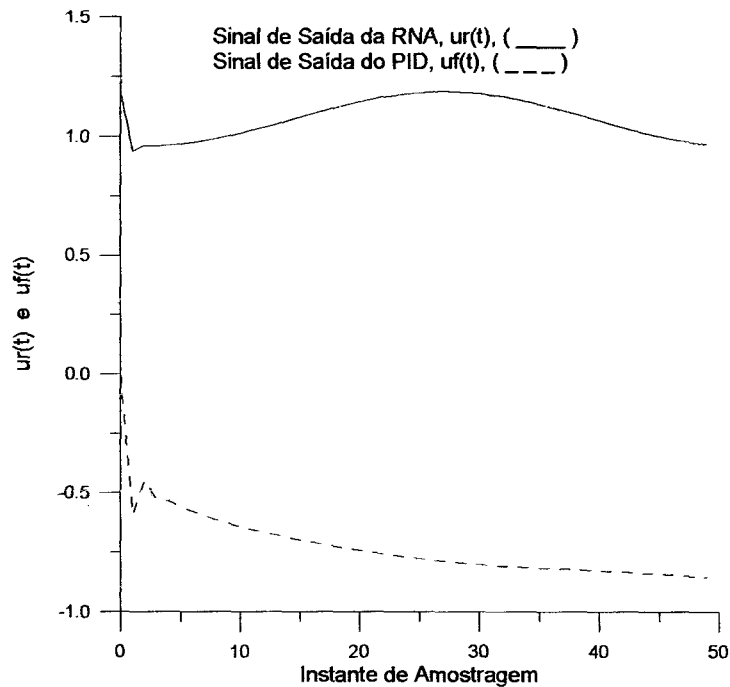


Figura 5.12 - Sinais de controle no início do aprendizado

Após 75285 iterações, interrompe-se a sessão de treinamento com o comportamento do sinal de saída do processo sendo mostrado na Figura 5.13. A Figura 5.14 mostra os sinais de controle  $u_r(k)$  e  $u_f(k)$  após o término do aprendizado.

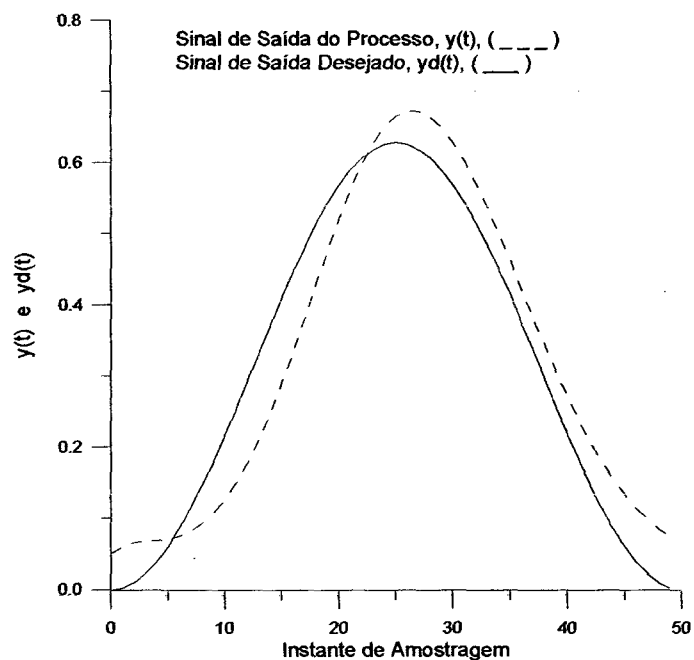


Figura 5. 13 - Comportamento da saída do processo após o treinamento

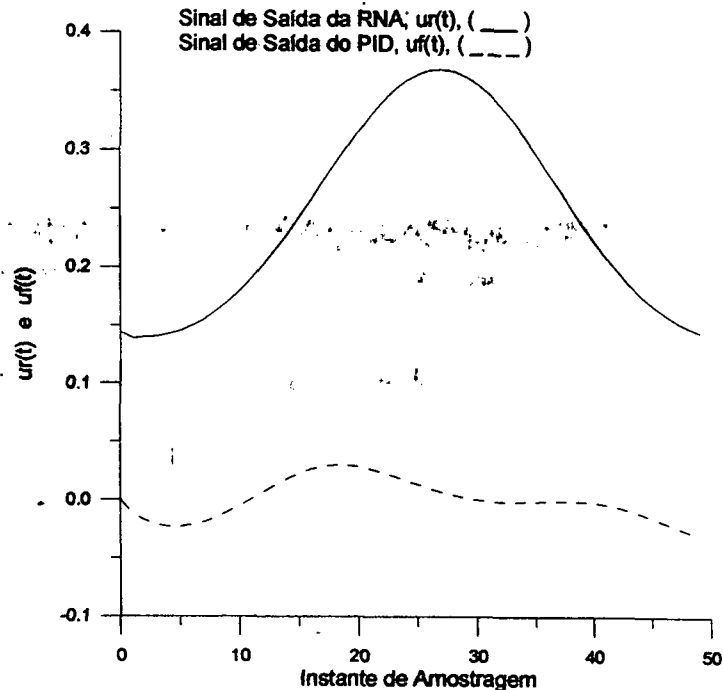


Figura 5. 14 - Sinais de controle após o treinamento

A Figura 5. 13 mostra o sinal de saída do processo atrasado de dois períodos de amostragem. Este atraso é o valor de  $q$  escolhido para esta simulação, que gera o sinal  $y_d(t-q)$  a ser seguido pelo processo. Neste exemplo o controlador neural apresenta um problema de nível dc na saída do processo (este efeito pode ser causado pelo uso do  $k_i$  igual a zero).

Nesta estratégia de controle, o controlador neural *feedforward* implementa o modelo inverso do processo. Para situações onde o processo não tem modelo inverso, este tipo de controle não funciona adequadamente. Esta situação é apresentada a seguir, quando utiliza-se a arquitetura de Kawato para controlar o processo abaixo, com o mesmo sinal de saída desejado da simulação anterior e valores de  $k_p$ ,  $k_i$  e  $k_d$  respectivamente iguais a 2.0, 0.0 e 0.01.

$$x(k+1) = \begin{bmatrix} 0.7 & 0 \\ 0 & -0.6 \end{bmatrix} x(k) + \begin{bmatrix} 1 \\ 1 \end{bmatrix} f[u(k)] \quad (5.13)$$

$$y(t) = [x_1(k)]^2 \quad (5.14)$$

$$f[u(k)] = \tanh[u(k)] \quad (5.15)$$

O controlador neural tem a seguinte topologia :

- A primeira camada apresenta um único neurônio com função ativação identidade, significando um neurônio estático, e função de saída do tipo tangente hiperbólica.
- A segunda camada contém dez neurônios com função ativação identidade, significando neurônios estáticos, e função de saída do tipo tangente hiperbólica.
- A terceira camada tem dois neurônios com função ativação do tipo equação a diferenças de primeira ordem, onde a entrada do neurônio é afetada apenas pelo estado do próprio neurônio, conforme equação (5.16), e função de saída do tipo linear com saturação (saturação igual a 100). Neste caso a matriz  $A$  é diagonal, como mostra a equação (5.17).

$$\begin{aligned} x_1(k+1) &= a_1 x_1(k) + net_1(k) \\ x_2(k+1) &= a_2 x_1(k) + net_2(k) \end{aligned} \quad (5.16)$$

$$A = \begin{bmatrix} a_{11} & 0 \\ 0 & a_{22} \end{bmatrix} \quad (5.17)$$

- A quarta camada contém dez neurônios com função ativação identidade, significando neurônios estáticos, e função de saída do tipo tangente hiperbólica.
- A última camada tem um único neurônio com função ativação identidade, significando um neurônio estático, e função de saída do tipo linear com saturação (nível de saturação igual a 100).

A Figura 5. 15 mostra o comportamento da saída do processo após 10000 iterações de treinamento. Foram realizadas mais de 50 simulações para tentar

controlar este processo, variando-se a topologia da rede, o número de neurônios nas camadas além dos parâmetros do algoritmo de treinamento, sendo que o melhor resultado conseguido foi o apresentado na Figura 5. .

A causa do péssimo comportamento do controlador neural, é que o processo a ser controlado apresenta problemas quanto ao seu modelo inverso.

Por exemplo, o bloco linear determinado pelas matrizes  $\begin{bmatrix} -0.7 & 0 \\ 0 & -0.6 \end{bmatrix}$  e  $\begin{bmatrix} 1 \\ 1 \end{bmatrix}$  da equação (5.13), indica que o sistema tem dois pólos e um zero (o número de zeros é independente dos valores assumidos para os componentes de uma possível matriz  $C$  na saída do processo,  $y(k) = C \bar{x}(k)$ , da mesma forma que tem-se  $B\bar{u}(k)$  na equação (5.13)). O inverso deste bloco linear apresenta um pólo e dois zeros, o que indica um sistema não-causal, ou antecipativo, o qual a rede neural não consegue implementar. Além deste problema, o inverso da não-linearidade  $f[u(k)]$  apresenta uma descontinuidade na origem, que também a rede neural não consegue implementar.

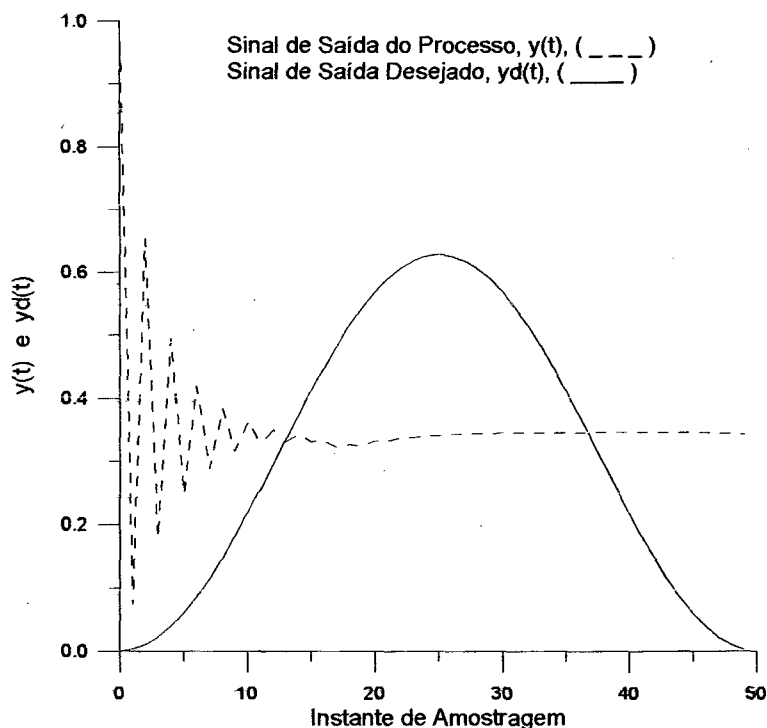


Figura 5. 15 - Comportamento da saída do processo

## 5.5 Discussões

O comportamento do modelo neural proposto como controlador *feedforward* mostrou-se adequado para controlar processos não-lineares. Nesta arquitetura de controle o controlador neural implementa o modelo inverso do processo em questão. Existe dificuldade no controle de processos que não apresentam modelos inversos, o que é inerente aos métodos que utilizam a modelagem inversa.

Como já visto nos dois capítulos anteriores, o modelo neural proposto serve como um aproximador não-linear universal porque implementa modelos não-lineares dos mais diversos tipos : contínuos, discretos, polinomiais, etc. Devido as suas particularidades, este novo modelo neural é mais adequado para tratar sistemas que apresentem modelos orientados à blocos. Neste caso, fica fácil verificar se o processo não-linear a ser controlado apresenta ou não modelo inverso.

Para os dois casos simulados em que o processo controlado é orientado à blocos, a rede neural apresenta os autovalores da matriz  $A$  variando de forma bem diferente. Para o primeiro caso, em que existe o modelo inverso do processo os autovalores de  $A$  são sempre reais, durante toda a sessão de treinamento. A Figura 5.16 mostra a evolução destes autovalores para 300000 iterações de treinamento. Ou seja, continua-se a treinar a rede após as 72000 iterações que geraram o comportamento da Figura 5.13. Observa-se que um dos autovalores aproxima-se do limite de instabilidade da rede ao ter seu módulo próximo da unidade.

Para o outro processo orientado à blocos, onde não é possível encontrar um modelo inverso, os autovalores da matriz  $A$  da rede não apresentam problemas de estabilidade e nem chegam perto do limite de estabilidade. Aqui já aparecem pares complexos conjugados de autovalores, como observado na Figura 5.17.



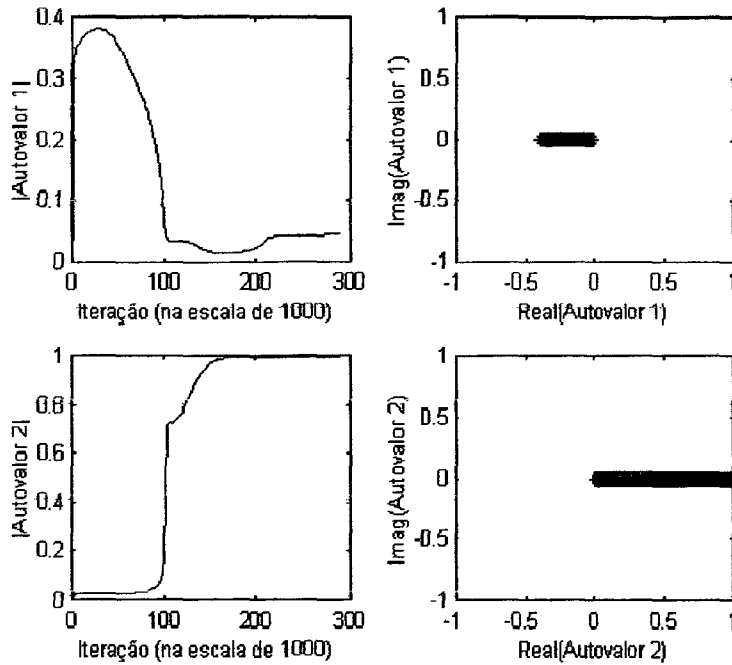


Figura 5. 16 - Comportamento dos autovalores da matriz  $A$  do modelo neural para um processo orientados à blocos que tem modelo inverso, onde observa-se o limiar da estabilidade do modelo neural

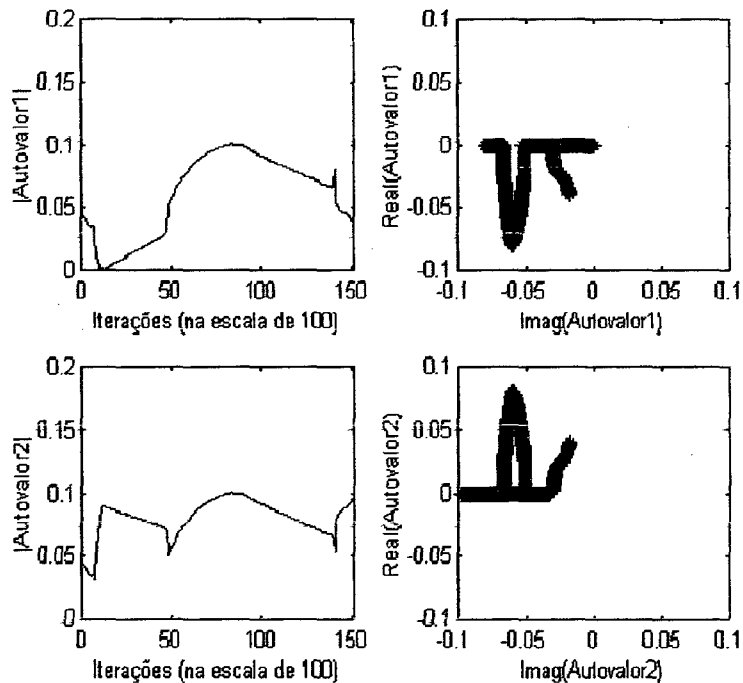


Figura 5. 17 - Comportamento dos autovalores da matriz  $A$  do modelo neural para um processo orientados à blocos que não tem modelo inverso

Ou seja, apenas o comportamento dos autovalores correspondentes aos neurônios dinâmicos não informa o quão “bom” está o desempenho do controlador

---

*feedforward* neural. Este desempenho tem que ser uma mistura dos valores das funções custo  $E[u_A(t)]$  e  $E[e(t)]$  e do comportamento dos autovalores dos neurônios dinâmicos da rede.

## 6 - Conclusões e Trabalhos Futuros

Este trabalho mostra a utilização de um novo modelo de RNA nas tarefas de identificação e controle de processos não-lineares. A nova RNA apresenta neurônios dinâmicos lineares caracterizando uma representação da dinâmica na forma interna, através do uso dos estados dos neurônios. A caracterização do estado do neurônio é determinada pelas definições formais do modelo de neurônio artificial encontradas no Anexo, que são parte do trabalho de De Azevedo (De Azevedo, 1993).

Este modelo de RNA preserva a característica de mapeamento não-linear universal encontrada em praticamente todos os modelos de RNA's descritos na literatura especializada. Esta universalidade leva em conta apenas a relação entrada-saída do processo, não preservando, necessariamente, a sua estrutura interna.

A nova rede neural tem uma estrutura particular, dinâmica linear e não-linearidade estática, que "casa" perfeitamente com processos orientados à blocos. Neste caso, existe a possibilidade de preservação da estrutura interna do processo. Neste trabalho, esta preservação não foi alcançada porque necessitaria de um algoritmo de aprendizado que estimasse estados e não apenas parâmetros. O aprendizado foi realizado com um algoritmo de estimação paramétrica que preserva apenas a relação entrada-saída do processo dinâmico não-linear.

Uma tarefa de identificação real apresenta sinais medidos que contêm informações contaminadas por ruído, seja ruído de medida ou ruído interno ao processo. Por este motivo, este trabalho apresenta simulações em que leva-se em conta a presença de ruído de medida no sinal de saída do processo. Este ruído pode influenciar a saída de maneira linear ou não-linear. Em ambos os casos, a RNA proposta apresentou um desempenho adequado para a identificação não-linear de processos orientados à blocos ou não. Este desempenho é "medido" através de testes de correlação.

A tarefa de controle é realizada através do uso da RNA com dinâmica interna como um controlador *feedforward*. A RNA, através de simulações, controla processos discretos orientados à blocos ou processos contínuos não orientados à blocos. Isto demonstra a versatilidade do modelo neural proposto, característica esta encontrada nos outros modelos de RNA's existentes.

Como possíveis trabalhos a serem realizados em continuação as questões apresentadas aqui, pode-se citar :

- inclusão, no algoritmo de aprendizado, do não aceite de ajuste nos parâmetros que tornem instável o modelo neural com dinâmica interna proposto, ou seja, uma versão do Algoritmo de Projeção, já desenvolvido para modelos lineares e para RNA's com dinâmica externa que apresentam uma relação linear entre os parâmetros e o vetor regressor;
- realização de estudos que possibilitem a estimação de estados do processo através do modelo neural com dinâmica interna proposto;
- utilização do modelo neural proposto em procedimentos de identificação e/ou controle de processos reais;
- utilização do modelo neural proposto em estimação de séries temporais;
- inclusão, no algoritmo de aprendizado, de inovações que melhorem o seu desempenho e já encontradas no algoritmo de treinamento *backpropagation* para RNA's com dinâmica externa, tais como : uso da informação de segunda derivada, uso de técnicas de computação evolucionária para ajuste de parâmetros da rede e para determinação do número de neurônios em cada camada da rede, etc.

# ANEXO

# Redes Neurais Artificiais : Conceitos, Definições e Modelos Utilizados

## A.1 Introdução

Neste anexo define-se formalmente um Neurônio Artificial (NA) e uma Rede Neural Artificial (RNA). Devido as RNA's serem um modelo bástante simplificado das redes neurais naturais, escolheu-se os conceitos da Teoria de Sistemas para formalizar um NA e também uma RNA. Antes de definir NA e RNA, apresenta-se alguns conceitos que são a base desta definições.

O uso destes conceitos e definições a serem apresentados, foram utilizados, pela primeira vez no contexto das RNA's, no trabalho de Barreto (Barreto, 1992) e mais tarde em trabalhos subsequentes (De Azevedo, 1993).

### A.1.1. Alguns Conceitos da Teoria de Sistemas

Matematicamente, Teoria de Sistemas é o estudo das iterações e comportamento de cada agrupamento de 'objetos' quando sujeito a certas condições ou entradas. A natureza abstrata da Teoria de Sistemas está relacionada ao fato que a mesma se interessa mais com as propriedades matemáticas do que com as formas físicas das partes que constituem o sistema (Brogan, 1985).

O estilo de apresentação, dos conceitos básicos da Teoria de Sistemas, é puramente matemático, baseado na teoria dos conjuntos. para permitir generalidade suficiente no uso dos conceitos apresentados.

#### A.1.1.1. Modelos

Para tratar matematicamente um sistema natural ou feito pelo homem, é necessário obter um *modelo* do mesmo. Normalmente o modelo é constituído por um conjunto de equações matemáticas que personificam os conceitos fundamentais de

uma teoria. Fundamentalmente, os modelos servem para colocar uma hipótese em uma forma concisa, forçando o investigador a especificar precisamente suas suposições (Barreto, 1995).

Definição A. 1 - Em modelagem, um sistema S é dado e um segundo sistema S' é construído utilizando-se de algum conhecimento sobre S. S' é dito ser um modelo de S.

Uma vez que o modelo matemático tenha sido estabelecido, o mesmo pode ser utilizado para quantificar os efeitos de uma mudança de parâmetros, muitas vezes com resultados inesperados (principalmente nos casos não-lineares). Um modelo é considerado válido, quando utilizado em várias situações diferentes, apresenta um comportamento suficientemente próximo do sistema real. Se um modelo é válido, é possível usá-lo para estudar o objeto real em situações que não são desejadas de serem realizadas com o sistema real.

Muitas vezes na construção de modelos existem dois objetivos conflitantes: o modelo deve representar o sistema o mais preciso possível e ao mesmo tempo, deve ser o mais simples possível. Encontrar o equilíbrio entre estas duas premissas é uma tarefa baseada na experiência do experimentador, no conhecimento sobre o sistema e nos objetivos dos estudos realizados.

Convém lembrar que ao construir-se modelos, a importância dos efeitos de cada parâmetro utilizado, tem diferentes graus de influência sobre o comportamento do sistema global. Se um conjunto de propriedades do sistema global deve ser representado com uma dada precisão, muitas das propriedades dos sistemas que compõem o sistema global podem ser negligenciadas. Entretanto o conjunto mínimo de componentes necessários para caracterizar adequadamente as propriedades de um sistema composto de vários outros sistemas não é uma tarefa fácil, e é possível que o modelo final torne-se não-realístico (Kalman et al, 1969).

### A.1.1.2. Sistemas em Diferentes Níveis de Descrição

Um primeiro passo, quando aplica-se métodos da Teoria de Sistemas é definir o que constitui o sistema real sobre estudo. Em outras palavras, o que pertence ao sistema e o que pertence ao mundo exterior. Em termos matemáticos isto significa uma partição do mundo, onde observa-se o sistema através de suas iterações com o exterior.

Um segundo passo a considerar, é que em função do objetivo da modelagem, aspectos do sistema real não são necessariamente retidos no modelo do sistema : algumas das facetas do sistema real devem ser retidas no modelo. Ou seja, para o mesmo sistema real, em função do que se deseja estudar, devem ser considerados diferentes modelos, dando uma coleção de modelos diferentes, com cada um considerando diferentes aspectos do sistema real.

Em outras palavras, de acordo com os limites do sistema real e da proposta para o modelo, são possíveis diferentes níveis de descrição.

### A.1.1.3. Sistema Geral

O nível mais geral de descrição de um sistema é quando somente as iterações com o mundo exterior são retidas no modelo. Este conceito é denominado de Sistema Geral.

Definição A. 2 - Um *sistema geral*  $\Sigma_g$  é definido pôr um conjunto de relações entre as entidades, caracterizando as iterações com o mundo exterior. Logo,  $\Sigma_g \in I$ , onde  $I$  é o conjunto de todas as iterações.

Neste nível de descrição a noção de causa e efeito não existe. Para se decidir o que é causa e o que é efeito, é necessário mais informações. Isto leva a noção de sistema orientado.



#### A.1.1.4. Sistema Orientado

Se é possível identificar as entradas  $Z$  e as saídas  $C$  do sistema, tem-se um sistema orientado.

Somente objetos que comunicam-se com o exterior nos interessam. De fato, um objeto que não tem comunicação com o exterior é um buraco negro: qualquer coisa que vai para o objeto desaparece e o objeto não influencia o exterior, logo, não pode ser observado. Mas, nem todos os atributos que caracterizam a comunicação com o exterior nos interessa, somente os atributos relevantes ao nosso objetivo de estudo. Uma vez que é possível identificar, alguns destes pontos de comunicação com o exterior na forma de entradas e saídas, tem-se um sistema orientado.

#### Definição 2.3

Definição A. 3 - Um *sistema orientado* pode ser caracterizado por uma relação entre o conjunto de entradas e saídas;

$$S \subset \Omega \times \Gamma$$

onde

$S$  é o sistema;

$\Omega$  é o conjunto de entradas admissíveis;

$\Gamma$  é o conjunto de saídas admissíveis.

#### A.1.1.5. Sistema Temporal

Os dois conceitos apresentados, de sistema geral e sistema orientado não incluem o tempo como algo intrínseco ao sistema. Incluindo o tempo em uma definição de sistema tem-se;

Definição A. 4 - Um sistema orientado onde  $e$  e  $s$  são funções do tipo:

$$\Omega : T \rightarrow U$$

$$\Gamma : T \rightarrow Y$$

onde

U é o conjunto de valores de entrada;

Y é o conjunto de valores de saída;

T é um conjunto ordenado, com um primeiro elemento muitas vezes denominado de  $t_0$ , que é denominado de conjunto tempo;

é um *sistema temporal*.

Utilizando a notação  $A^T$  para indicar um conjunto de funções com domínio T e conjunto imagem A, um sistema temporal é representado pela relação;

$$S \subset Y^T \times U^T \quad (\text{A. 1})$$

**Definição A. 5** - Um sistema temporal cujas funções  $\Omega$  e  $\Gamma$  são funções constantes (a imagem é sempre o mesmo elemento de U e Y respectivamente) é dito ser um *sistema estático*.

**Definição A. 6** - Um *sistema causal* é um sistema temporal  $\Sigma_C \subset \Gamma^T \times \Omega^T$  tal que;

$$\forall t \in T, \text{ se } \omega_1(t_0, t] = \omega_2(t_0, t] \text{ então } \gamma_1(t) = \gamma_2(t)$$

onde  $t_0$  é o primeiro elemento do conjunto tempo.

**Definição A. 7** - Se T é um intervalo do conjunto dos números reais, nós dizemos que o sistema é um *sistema contínuo no tempo*.

**Definição A. 8** - Se T é um sub-conjunto dos números inteiros, então o sistema é um *sistema discreto no tempo*.

#### A.1.1.6. Sistema Funcional

Até este momento o sistema é descrito em relação ao mundo exterior (sistema é descrito no nível de comportamento). Este nível de comportamento é importante porque é nesse nível de descrição que tem-se informações retiradas através de experimentos. Entretanto, esta descrição não diz nada sobre o mecanismo

necessário para produzir este comportamento, e é possível que dois mecanismos totalmente diferentes produzam o mesmo comportamento.

Este mecanismo pode ser associado com um novo conjunto  $X$ , tal que, dado um elemento do conjunto de entradas  $\Omega$  e um elemento deste novo conjunto de parâmetros  $X$ , pode-se definir univocamente o elemento do conjunto de saída. Isto nos leva a definição de um sistema através de uma função, ou seja um sistema funcional. Esta descrição é dita ser a nível de estrutura de estado.

**Definição A. 9** - Um *sistema funcional* é caracterizado pôr uma função  $f$ ;

$$f : \Omega \times X \rightarrow \Gamma$$

onde

$\Omega$  é o conjunto de entradas admissíveis;

$\Gamma$  é o conjunto de saídas admissíveis;

$X$  é o vetor de estados ou conjunto de estados.

#### A.1.1.7. Sistema Dinâmico

Um sistema pode ser funcional e temporal. Neste caso o estado muda com o tempo. Este sistema é denominado de sistema dinâmico. Em um sistema dinâmico a descrição do sistema é feita como se fosse descrito o mecanismo de como o sistema trabalha internamente, pela especificação de como os estados modificam-se com o tempo.

**Definição A. 10** - Um *sistema dinâmico* é o objeto matemático;

$$S = \{ T, U, \Omega, Y, \Gamma, X, \Phi, \eta \}$$

onde

$T$  é o conjunto tempo;

$\Omega$  é o conjunto de funções de entrada  $\omega \in \Omega = \{ \omega : T \rightarrow U \}$ ;

$U$  é o conjunto de valores de entrada;

$Y$  é o conjunto de valores de saída;

$\Gamma$  é o conjunto de funções de saída  $\gamma \in \Gamma = \{ \gamma : T \rightarrow Y \}$ ;

$X$  é o vetor de estados (ou conjunto de estados);

$\Phi$  é a função transição de estados  $\Phi : T \times T \times X \times \Omega \rightarrow X$ ;

$\eta$  é a função de saída  $\eta : T \times X \times U \rightarrow Y$ ;

satisfazendo algumas condições de compatibilidade (Kalman et al, 1969).

#### A.1.1.8. Tipos Particulares de Sistemas Dinâmicos

Escolhas particulares dos conjuntos envolvidos na definição de sistemas dinâmicos leva a diferentes tipos de sistemas.

**Definição A. 11** - Um *sistema dinâmico contínuo no tempo* (ou apenas *sistema contínuo no tempo*) é um sistema dinâmico onde;

$T$  é um compacto subconjunto completo dos números reais;

$X, U, Y$  são subconjuntos de  $R^n, R^m, R^p$  no espaço real de  $n, m, p$  dimensões<sup>1</sup>;

$\Phi$  é um conjunto de funções diferenciáveis por partes em relação a  $t$ .

**Definição A. 12** - Um *sistema dinâmico discreto no tempo* (ou apenas *sistema discreto no tempo*) é um sistema dinâmico cujo conjunto  $T$  é um sub-conjunto dos inteiros.

**Definição A. 13** - Um *sistema invariante no tempo* (ou *sistema estacionário*) é um sistema dinâmico em que a função transição de estados  $\Phi$  depende somente de um elemento de  $T$  e a função saída é independente de  $T$ .

É normal fazer, como primeira aproximação do sistema real, um modelo invariante no tempo. O fato de que a função transição depende somente de um elemento de  $T$  significa que o valor deste estado não depende do tempo inicial e do instante considerado. A função de saída sendo independente do tempo significa que para qualquer instante, estados e entradas iguais produzem a mesma saída. Se é

este o caso, para qualquer tempo inicial, somente a duração do experimento é importante para determinar o estado e a saída em qualquer instante.

Como tratar com sistemas variantes no tempo não é uma tarefa fácil, algumas vezes o espaço de estados é enriquecido com uma nova variável, a variável tempo, tornando-se agora um sistema invariante no tempo com a dimensão do espaço de estados aumentada de uma unidade, com a variável tempo sendo um elemento do conjunto de estados.

Um tipo de sistema dinâmico também utilizado, é a 'máquina de estado finito'. Informalmente uma 'máquina de estado finito' é um sistema dinâmico onde o conjunto tempo é o conjunto de números inteiros, e entradas, saídas e estados são conjuntos finitos.

Uma generalização da 'máquina de estado finito' é o 'automata'. Formalmente tem-se a seguinte definição para um 'automata'.

Definição A. 14 - Um *automata* (ou máquina) é descrito de forma abstrata como a sextupla;

$$S = \{U, Y, X, x_0, \lambda, \eta\}$$

onde

U é o conjunto finito de entradas;

Y é o conjunto finito de saídas;

X é o conjunto finito de estados ou espaço de estados;

$x_0 \in X$  é o estado inicial;

$\lambda : U \times X \rightarrow X$  é a função transição de estados (próximo-estado);

$\eta : U \times X \rightarrow Y$  é a função próxima-saída.

note que o tempo não aparece explicitamente. mas um conjunto tempo, materializado no conceito 'próximo', está presente.

---

<sup>1</sup> É possível também ter o n infinito. no caso de sistemas definidos por equações diferenciais parciais.

Um automata é um sistema dinâmico discreto e invariante no tempo.

#### **A.1.1.9. Sistemas Complexos**

Em sistemas dinâmicos complexos, uma descrição do sistema tal que especifique como o mesmo é construído, pode ser feita pela conexão de vários sistemas elementares. As caixas-pretas elementares são sistemas definidos segundo as definições anteriores. O acoplamento entre cada sistema elementar determina uma estrutura complexa. Esta classe de sistema é também denominada de sistema hierárquico.

Definição A. 15 - Um *sistema dinâmico complexo* é uma rede de sistemas interconectados.

Uma regra referente as possíveis descrições diferentes de um sistema é que, dado uma especificação de sistema para um certo nível, pode-se associar, ao menos, outra especificação para o próximo nível abaixo. Ou em outras palavras, descendo na descrição de um sistema, tem-se em geral várias escolhas, por exemplo, para uma mesma descrição de sistema como um sistema orientado, pode corresponder vários subsistemas do tipo funcional, pela definição de diferentes conjuntos de estados. A Figura A. 1 ilustra uma hierarquia nos conceitos de sistemas.

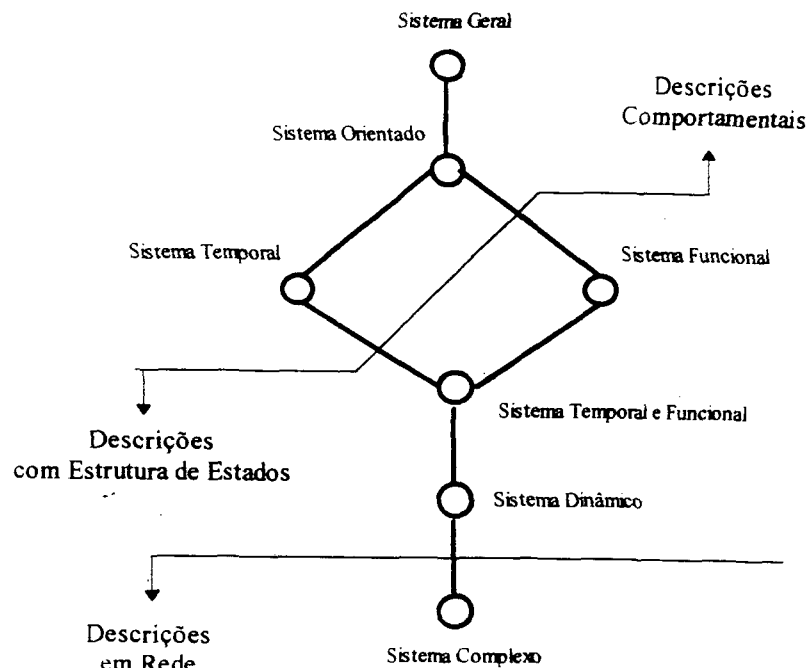


Figura A. 1 - Hierarquia dos Conceitos de Sistemas (Barreto, 1996).

#### A.1.1.10. Sistemas Lineares

Sistemas lineares formam uma importante classe de sistemas. A hipótese de linearidade é tão tentadora que textos elementares na teoria de sistemas normalmente iniciam pela explanação do 'princípio' (antes da hipótese) da superposição<sup>2</sup> sem prestar atenção à motivação do mesmo, e sem dar uma definição clara do que um sistema é (Barreto, 1992). Pôr mais que esta abordagem possa ter virtudes pedagógicas no princípio, o leitor enfrenta sérias quando aprende sobre outras partes da Teoria de Sistemas, tais como mecanismos não-lineares, estabilidade, etc.

Devido a este fato, o conceito de espaço vetorial é importante porque é esta estrutura matemática que permite falar de multiplicação por uma constante e adição

<sup>2</sup> O termo 'princípio da superposição' dá a impressão de ter sido emprestado da mecânica quântica. Isto é arcaico e mal direcionado. Em mecânica quântica 'superposição' significa que os estados estão em um espaço linear; na Teoria de Sistemas, 'superposição' significa que a função entrada/saída é linear.

de dois valores (necessários para o princípio da superposição). De fato, em um espaço vetorial existem dois conjuntos envolvidos,  $V$  e  $K$  denominados respectivamente conjunto de vetores e conjunto de 'escalares' ou 'constantes'. O conjunto  $V$  é um 'grupo abeliano' e  $K$  é um campo. Existe uma operação externa definindo a combinação de elementos de  $V$  com elementos de  $K$ , com imagem em  $V$ , denominada de multiplicação do vetor pôr um escalar.

Com base nos conceitos básicos e no princípio da superposição, apresenta-se duas definições, uma utilizando estruturas matemáticas básicas da álgebra linear e outra baseada no 'princípio da superposição'.

**Definição A. 16** - Um sistema dinâmico  $S$  é *linear* se e somente se;

- i)  $X, U, \Omega, Y$  e  $\Gamma$  são espaços vetoriais (sobre um campo arbitrário  $K$ );
- ii) o mapeamento dado pôr  $\Omega : T \times T \times X \times \Omega \rightarrow X$  é  $K$ -linear para todo  $t$ ;
- iii) o mapeamento dado pôr  $\eta : T \times X \times U \rightarrow Y$  é  $K$ -linear para todo  $t$ .

Convém observar que a definição (2.16) versa sobre sistema dinâmico. Utilizando apenas o 'princípio da superposição', tem-se uma definição que não envolve necessariamente um sistema dinâmico.

**Definição A. 17** - Um sistema é *linear* se é homogêneo e aditivo. Um sistema é *homogêneo* se ao multiplicar a entrada pôr uma constante a saída aparece multiplicada pela mesma constante. Um sistema é *aditivo* se a saída da soma de duas entradas é a soma das saídas das entradas aplicadas separadamente.

Da Definição A. 17 temos que o conceito de sistema linear pode ser aplicado no caso de sistemas orientado, funcional, temporal, dinâmico e complexo.

Também pode-se afirmar que, um sistema é dito ser não-linear se não é linear.

Sistemas lineares são importantes, principalmente pelas seguintes razões :



- existem vários métodos bem desenvolvidos para estudar sistemas lineares.
- como uma primeira aproximação, sistemas não-lineares podem ser estudados utilizando uma aproximação linear e estas aproximações permitem tirar algumas conclusões sobre o comportamento do sistema não-linear. Como por exemplo, análise de estabilidade através do teorema de estabilidade local de Lyapunov (ou primeiro teorema de Lyapunov).

Da álgebra linear é conhecido que operadores lineares podem ser representados por matrizes. Logo, todo sistema orientado linear pode ser representado por uma matriz. Se o sistema linear é um sistema temporal, os elementos da matriz tornam-se funções do tempo.

Deve-se também lembrar que a palavra 'escalar' não é utilizada aqui como sinônimo de número real ou complexo, mas sim como um elemento de um campo, que é parte da estrutura algébrica do espaço vetorial.

Os mais comuns sistemas dinâmicos lineares são definidos por equações diferenciais ou equações a diferenças. Entre estes, os mais importantes são os invariantes no tempo. Os quais são representados por equações diferenciais ou diferença com coeficientes constantes (números reais ou complexos).

#### **A.1.1.11. Propriedades de Sistemas**

##### **A.1.1.11.1. Realimentação**

Realimentação é o resultado de uma das mais importantes topologia de um sistema descrito a nível de composição de estrutura. Existe realimentação quando a entrada do sistema depende da sua saída, como mostrado na Figura A. 2.

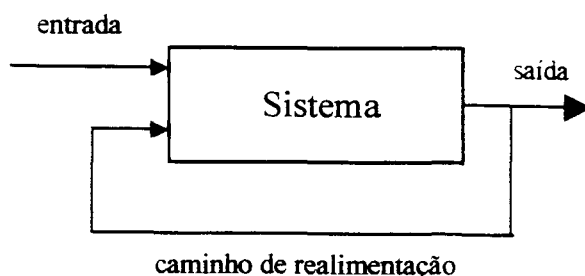


Figura A. 2 - Sistema com Realimentação.

Essencialmente existem duas classes de realimentação : positiva e negativa. Existe realimentação positiva quando o aumento na saída é refletido na entrada por um aumento na mesma. Existe realimentação negativa quando um aumento na saída serve para gerar um decréscimo na entrada.

#### A.1.1.11.2. Identificabilidade

Na área de Identificação de Sistemas estima-se ou calcula-se um modelo matemático de um sistema baseado nas medidas da entrada e saída do sistema (Kalman et al, 1969). Três passos estão presentes na identificação: a determinação da estrutura do modelo, a identificação dos parâmetros da estrutura determinada e a validação do modelo encontrado.

Uma vez que tenha-se adotado uma estrutura, tem-se o conceito de identificabilidade.

**Definição A. 18** - Um sistema é identificável se todos os seus parâmetros podem ser calculados utilizando as medidas de entradas e saídas.

Um problema fundamental na identificação de sistemas dinâmicos é a determinação da estrutura do modelo. O problema pode ser brevemente explorado como segue-se. Identificação consiste na escolha de uma estrutura de modelo candidata, estimação dos parâmetros do modelo escolhido e validação (isto é, aceitação ou rejeição) do modelo estimado. Se o modelo é insatisfatório, outra estrutura é escolhida (ou outra técnica de estimação de parâmetros é escolhida). A

estrutura do modelo é uma representação parametrizada do sistema dinâmico que depende de um número finito de parâmetros (embora em alguns casos esta estrutura seja conhecida, baseada em leis físicas, tais como lei da conservação, leis da mecânica e da eletricidade, etc). A questão da identificabilidade pode então ser formulada : O mapeamento entrada/saída feito pelo modelo depende de maneira única do vetor de parâmetros ? Somente se a resposta a esta pergunta é 'sim', os parâmetros podem ser unicamente identificáveis a partir dos dados de entrada/saída. Se a estrutura do modelo não é identificável, então a identificação não faz sentido.

#### A.1.1.11.3. Controlabilidade e Observabilidade

Controlabilidade e observabilidade são conceitos duais que referem-se a representações de um sistema para o nível de descrição de estrutura de estado. Intuitivamente um sistema é controlável se existe uma função entrada capaz de transferir o estado de um sistema do estado inicial para outro estado final.

Até os dias atuais não existe um método geral para determinar se um dado sistema, independentemente do seu modelo matemático, é controlável ou observável. Este requisito é encontrado apenas para sistemas dinâmicos lineares. Assim, apresenta-se as seguintes definições, enfocando o caso linear.

Definição A. 19 - Um sistema dinâmico linear é dito ser *controlável* para  $t_0$  se é possível encontrar alguma função  $\Omega$  que transfere o estado inicial  $x(t_0)$  para a origem do espaço de estado em um tempo finito  $t_1 \in T$ , com  $t_1 > t_0$ .

O sistema é observável se o conhecimento da saída é suficiente para determinar o estado do sistema.

Definição A. 20 - Um sistema dinâmico linear é dito ser *observável* para  $t_0$  se  $x(t_0)$  pode ser determinado a partir de uma sequência de saída  $Y(t_0, t_1)$  para  $t_0 \in T$ , com  $t_1 \geq t_0$ , onde  $t_1$  é algum instante de tempo finito pertencente a  $T$ .

Controlabilidade e observabilidade são duas qualidades quase que sempre requeridas em sistemas que sejam utilizáveis. Sistemas não-controláveis e não-observáveis normalmente são resultados de uma descrição mal feita do sistema.

De fato, dizer que um sistema é não-controlável é reconhecer que a suposta forma de controle do sistema não é adequada. Adicionalmente, dizer que um sistema é não-observável é equivalente a reconhecer de que a maneira pretendida para medir o comportamento do sistema não foi imaginada adequadamente e deve ser modificada se realmente pretende-se observar o comportamento do sistema

Um ponto importante é que controlabilidade e observabilidade são propriedades de um sistema que estão intrinsecamente associadas com a entrada e saída consideradas. Com diferentes escolhas para as saídas e entradas, a condição de controlabilidade e observabilidade pode ser diferente.

#### A.1.1.11.4. Minimalidade

Um modelo deve ser o mais simples possível para responder as necessidades do experimentador. Um modelo satisfazendo este requisito é dito ser parcimonioso (Kalman et al, 1969).

Logo, quando seleciona-se um modelo os seguintes aspectos são relevantes:

- i) o sistema deve ser controlável e observável;
- ii) a maioria dos parâmetros utilizados devem ter um significado físico real;
- iii) os parâmetros utilizados devem ser identificáveis;
- iv) o comportamento do modelo e do sistema real devem ser similares quando observados do exterior (pares entrada/saída);
- v) o modelo deve ser o mais simples possível, dentro dos modelos possíveis.

Assim, pode-se ter dois modelos representando o mesmo sistema físico real, que observando as suas saídas, sejam equivalentes. Mas, um pode exigir menos recursos computacionais que o outro.

**Definição A. 21** - Dois sistemas  $S'$  e  $S''$  são equivalentes se é impossível distingui-los pôr qualquer experimento simples : para todo  $x'$  e para todo  $u$ , existe um  $x''$  tal que  $h(x', u) = h(x'', u)$ .

Minimalidade está intrinsecamente associado com os conceitos de controlabilidade e observabilidade. O sistema mínimo deve ser a parte do sistema que é controlável e observável, uma vez que é possível imaginar o sistema decomposto em quatro partes, como mostrado na Figura A. 3.

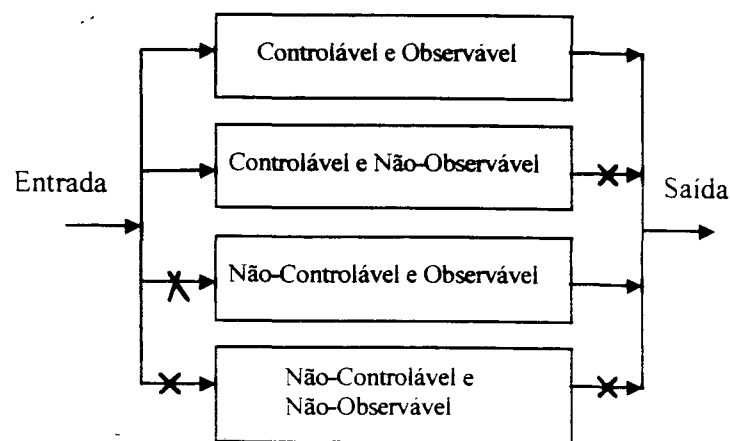


Figura A. 3 - Decomposição Canônica.

#### A.1.1.11.5. Estabilidade

Informalmente, estabilidade refere-se a um sistema que tem sua funcionalidade imutável quando sofre uma perturbação de alguma interferência externa. Neste conceito, existem vários pontos que devem ser discutidos antes de uma definição precisa de estabilidade ser apresentada :

- Como estabilidade envolve a idéia de uma possível mudança no estado ou saída do sistema, o sistema deve ser pelo menos um sistema temporal, ou seja, um sistema estático não pode ser estável ou instável.
- A precisa funcionalidade que deseja-se considerar deve ser estática. Pôr exemplo, ponto de equilíbrio estável ou trajetória estável.

- A importância da interferência externa. Normalmente em estabilidade existe a noção de domínio de estabilidade, ou seja, um sistema não é estável para qualquer tipo de perturbação, mas sim para perturbações que modifiquem o estado do sistema dentro da região de estabilidade.
- A duração da perturbação também é importante. É completamente diferente falar sobre estabilidade durante a presença de uma perturbação ou se está interessado em conhecer se após a ocorrência da perturbação, o sistema é capaz de retornar ao estado antes da presença da perturbação.

As várias condições diferentes sobre as quais é possível falar de estabilidade permite existir vários conceitos relativos a estabilidade.

Começaremos apresentando a definição proposta por Lyapunov (Lyapunov, 1966), em sua tese de doutorado no ano de 1892. Esta definição é válida para sistemas lineares e não-lineares, e refere-se a estabilidade de um ponto de equilíbrio e de uma trajetória de equilíbrio.

Definição A. 22 - *Ponto de equilíbrio* : suponha um sistema e a entrada do sistema é nula para todo instante de tempo maior que um certo  $t=T$ . O valor dos estados do sistema  $x_0 \in X$  é um ponto de equilíbrio do sistema se e somente se para  $\forall t < T$ , tem-se  $x(T) = x_0 \Rightarrow x(t) = x_0$ . Existe o conceito análogo referindo-se a saída do sistema.

Definição A. 23 - *Trajétoria de equilíbrio* : suponha um sistema e a entrada do sistema é nula para todo instante de tempo maior que um certo  $t = T$ . Se o estado do sistema é periódico para  $\forall t > T$ , então os valores dos estados no período constituem uma trajetória de equilíbrio. Existe conceito análogo referindo-se a saída do sistema.

É interessante observar que o ponto de equilíbrio e a trajetória de equilíbrio não se preocupam com o comportamento do sistema se a entrada é não-nula durante algum intervalo de tempo ou se existe perturbação. Acontecendo uma destas duas situações, os estados do sistema voltam para o equilíbrio (ponto ou trajetória) ?

Quão grande deve ser a entrada ou perturbação para que os estados do sistema retornem ao equilíbrio (ponto ou trajetória) ? Estas questões levam as definições que se seguem.

**Definição A. 24 - Estabilidade de Lyapunov** : Considere um sistema com entrada  $U$  e saída  $Y$ . O sistema é dito ser estável se, dado um número positivo  $\varepsilon$ , é possível encontrar outro  $\eta$  positivo tal que, se o sistema tem entrada nula para  $\forall t > 0$ , se as condições iniciais dos estados estão no interior da esfera de raio  $\varepsilon$ , existirá um  $T$  para o qual os estados para  $\forall t > T$  permanecem no interior da esfera de raio  $\eta$ .

**Definição A. 25 - Estabilidade Local** : um ponto (trajetória) de equilíbrio é dito ser localmente estável se existe um conjunto aberto incluindo o ponto de equilíbrio no qual, todas as condições iniciais correspondem a um comportamento estável do sistema.

**Definição A. 26 - Domínio de Estabilidade** : o domínio de estabilidade de um ponto (trajetória) de equilíbrio é o conjunto de pontos correspondendo as condições iniciais para o qual o sistema é estável.

**Definição A. 27 - Estabilidade Global** : um sistema é dito ser globalmente estável em uma região se é estável para todas as condições iniciais pertencentes a pontos nesta região.

## **A.2 Neurônio Artificial**

### **A.2.1. Neurônio Biológico**

Para entender melhor o funcionamento da abordagem neural, é necessário entender o funcionamento e a evolução do sistema nervoso dos seres vivos, e como, a partir do funcionamento deste sistema, parece emergir uma consciência inteligente.

Com o surgimento dos primeiros organismos multicelulares, há aproximadamente 1 bilhão de anos atrás, algumas células passaram a ter funções

especializadas tais como digestão de alimentos, transporte de nutrientes para outras células, contração e alongamento para produzir movimento, etc. O resultado desta forma de organização é um sistema mais durável do que cada um de seus componentes e, portanto, com mais chances de reproduzir do que seus concorrentes unicelulares.

A coordenação das partes especializadas requer uma comunicação entre as células. As próprias células possuem as características básicas necessárias a criação de um elo de comunicação. A maioria das células mantém uma pequena diferença de voltagem, uma polarização, entre as superfícies interna e externa da membrana que as envolve. Um determinado distúrbio em qualquer ponto da membrana pode causar uma súbita despolarização naquele ponto e se propagar por certa distância na superfície da célula. Após a despolarização, a célula procura ativamente se repolarizar. Se as células possuírem uma forma alongada, com filamentos de um metro ou mais, em casos extremos, tem-se elementos perfeitos para um sistema de comunicação : células nervosas especializadas e capazes de conduzir impulsos eletro-químicos pôr longas distâncias e a altas velocidades.

Com a articulação destas células, tem-se o início do sistema nervoso periférico e do sistema nervoso central. Estas células alongadas, capazes de transportar impulsos elétricos são denominados de neurônios. O cérebro humano contém aproximadamente  $10^{11}$  neurônios. Cada um deles está conectado a aproximadamente  $10^4$  outros neurônios.

Existem centenas de tipos de neurônios, cada um com suas funções características, forma e localização. Entretanto, a estrutura básica é sempre a mesma, e consiste no corpo da célula, denominado soma, os dentritos e o axônio, como visto na Figura A. 4.



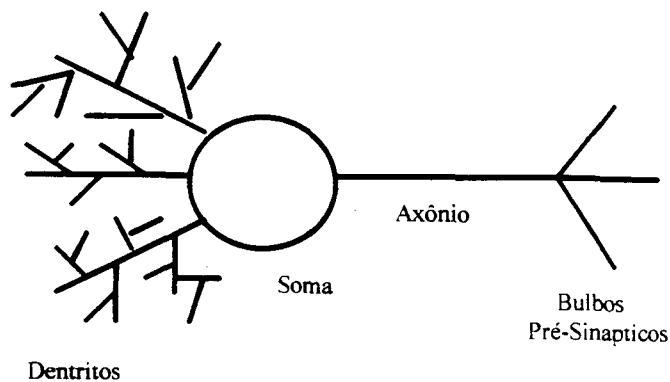


Figura A. 4 - Representação Simplificada de um Neurônio Biológico.

O soma possui um diâmetro entre 5 e 100  $\mu\text{m}$ , contém o núcleo da célula e as funções necessárias à manutenção da vida dos neurônios, tais como síntese de enzimas. Os dentritos atuam como os canais de entrada dos sinais externos recebidos pelos neurônios e o axônio atua como o canal de saída. Os axônios de vários outros neurônios fazem contato com os dentritos de um dado neurônio. Estas conexões são denominadas de sinapses, e permitem que uma célula influencie a atividade de outra célula da seguinte maneira: quando um pulso de despolarização, denominado de potencial de ação, chega ao final de um axônio, o mesmo provoca a liberação de uma substância química, denominada neuro-transmissor, contida em pequenos bulbos nas extremidades do axônio; estes neuro-transmissores atravessam o pequeno espaço da sinapse e são recebidos pelos dentritos do neurônio seguinte. Dependendo da natureza dos neuro-transmissores liberados pelos bulbos e da natureza dos receptores químicos que os recebem, no outro lado da sinapse, a mesma é denominada de inibitória ou excitatória.

Em uma sinapse inibitória, a transmissão sináptica provoca uma pequena hiper-polarização no potencial elétrico do neurônio afetado. Isto faz com que seja mais difícil para este neurônio se despolarizar e propagar seu próprio potencial de ação ao longo de seu axônio. Ao contrário, em uma sinapse excitatória, a transmissão sináptica provoca uma pequena despolarização no neurônio afetado, rebaixando seu potencial elétrico em direção a um ponto mínimo crítico, onde o

neurônio abruptamente pode se despolarizar a ponto de iniciar a transmissão de seu próprio potencial de ação, através do seu axônio.

Estes dois tipos de sinapses, juntas, em cada neurônio, formam um palco de competição (dispara-não-dispara) onde um dos fatores que determina qual o comportamento vencedor, parece ser o número e a proximidade das sinapses excitatórias e inibitórias. Em um pequeno intervalo de tempo, estas conexões apresentam uma característica relativamente estável para cada neurônio. No entanto, novas conexões podem aparecer e outras deixarem de existir, algumas vezes estas mudanças ocorrem em questões de minutos ou menos; deste modo, as propriedades funcionais de um neurônio são pôr si só plásticas, o que poderia explicar mudanças de comportamento em função do aprendizado.

### A.2.2. Modelo de Neurônio

A construção de RNA's tem inspiração nos neurônios biológicos e nos sistemas nervosos<sup>3</sup>. Entretanto, é importante compreender que atualmente as RNA's estão muito distantes das redes neurais naturais e frequentemente as semelhanças são mínimas. Dois fatores diferentes motivam a pesquisa hoje em dia (Barreto, 1996) :

- O primeiro é modelar o sistema nervoso com suficiente precisão de tal modo a poder observar um comportamento emergente que sendo semelhante ao comportamento do ser vivo modelado, possa servir de apoio às hipóteses usadas na modelagem.
- O segundo é construir computadores com um alto grau de paralelismo.

---

<sup>3</sup>Detalhes de fisiologia, indispensáveis à compreensão das RNA's, podem ser encontrados em livros de fisiologia tal como Guyton (Guyton, 1976).

### A.2.2.1. Modelo Geral de Neurônio

O modelo geral de neurônio é mostrado na Figura A. 5. Neste modelo as entradas  $w_i u_i$  são combinadas usando uma função  $\phi$  para produzir um estado de ativação do neurônio que através da função  $\lambda$  vai produzir a saída do neurônio (correspondente à frequência de descarga do neurônio biológico), sendo que  $d$  indica a possibilidade de inclusão de dinâmica no processamento do neurônio.

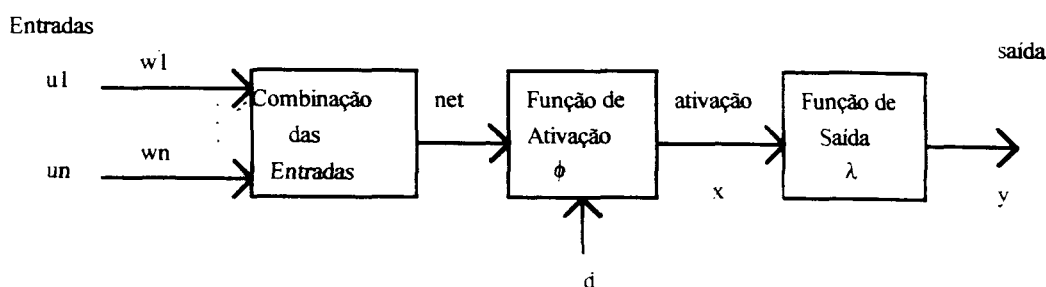


Figura A. 5 - Elementos Básicos de um Neurônio Artificial.

Note que as conexões sinápticas são consideradas como externas ao modelo do neurônio, tal como ocorre no sistema nervoso biológico e não como fazendo parte do neurônio, como usado por alguns autores. Este detalhe proporciona a possibilidade de interpretar a matriz de conexões como a matriz de pesos de um grafo, o grafo representativo da rede neural (Barreto, 1996; De Azevedo, 1993).

As características deste modelo permitem definir o modelo geral de um neurônio como sendo um sistema dinâmico. Permitindo também, junto com os conceitos de sistemas apresentados, ter vários tipos de neurônios.

No trabalho apresentado por Barreto (Barreto, 1996) tem-se o teorema de definição de um neurônio artificial ou neurônio formal, e as definições de alguns tipos neurônios:

Teorema A. 1 - O neurônio formal é um sistema dinâmico.

Definição A. 28 - Um neurônio é estático quando o valor de  $x$  e de  $y$  se referem ao mesmo instante que as excitações. Assim, pode-se escrever :

$$x = \phi (w_i, u_i)$$

$$y = \lambda (x)$$

**Definição A. 29** - O neurônio é linear se  $\phi$  e  $\lambda$  são funções lineares.

**Definição A. 30** - O neurônio é não-estacionário se as funções  $\phi$  e/ou  $\lambda$  são funções do tempo. Neste caso deve-se escrever :

$$x = \phi (w_i, u_i, t)$$

$$y = \lambda (x, t)$$

**Definição A. 31** - O neurônio é dinâmico se para o cálculo de  $x$  em um determinado instante é necessário o conhecimento de  $x$  em um instante anterior. Sendo assim, como um exemplo de neurônio dinâmico, escreve-se :

$$x(t+h) = \phi [x(t), w_i, u_i, t, h]$$

$$y(t) = \lambda (x(t), t)$$

#### A.2.2.1.1. Formalismo Matemático do Neurônio Artificial

Dos elementos básicos do NA, mostrado na Figura A. 5, tira-se várias relações matemáticas, que serão mostradas a seguir.

##### A.2.2.1.1.1. Sinais de Entrada

Da Figura A. 5 tem-se que cada entrada do neurônio possui um conjunto de pesos associados, que pode ser entendido como a 'força' da conexão sináptica quando comparado ao neurônio biológico. As entradas de um neurônio podem ser as saídas de outros neurônios,  $y(t)$ , entradas externas  $u$ , um *bias*  $b$  ou qualquer combinação destes elementos. O somatório de todas estas entradas dá origem ao chamado *net* de um neurônio, que pode ser representado, para o  $i$ -ésimo neurônio, por:

$$net_i(t) = \sum_{j=1}^p w_{ji} y_j(t) + \sum_{l=1}^m w_{li} u_l(t) + w_{(m+p)i} b_i \quad (\text{A. 2})$$

onde  $y_j$  é a saída do  $j$ -ésimo neurônio,  $u_l$  é a  $l$ -ésima entrada externa,  $b_i$  é o bias do  $i$ -ésimo neurônio e  $w$  refere-se ao peso correspondente.

É importante observar que os pesos  $w$ , que representam a intensidade de uma conexão sináptica entre dois neurônios, não fazem parte do NA formal. O que significa dizer que os pesos só possuem sentido prático quando um neurônio se conecta a outro, exercendo sobre este uma ação que é ponderada pelo valor do peso.

Normalmente se encontra na literatura a notação da equação (A.2), onde todas as entradas de um neurônio são incorporadas sobre a representação  $u_j$  e os pesos simplesmente por  $w_{ji}$ , com  $j$  variando de 1 até  $m+p$ , ou apenas de 1 até  $n$ ;

$$net_i(t) = \sum_{j=1}^n w_{ji} u_j(t) \quad (\text{A. 3})$$

com  $w_{ji}$  sendo um número real representando a conexão sináptica da entrada do  $i$ -ésimo neurônio com a  $j$ -ésima entrada, ou a saída do  $j$ -ésimo neurônio. Esta conexão será excitatória se  $w_{ji} > 0$  ou inibitória se  $w_{ji} < 0$ .

#### A.2.2.1.1.2. A Função de Ativação $\phi(\cdot)$

Após a determinação do valor de  $net_i(t)$ , o estado do neurônio, ou ativação, é atualizado através da função de ativação  $\phi(\cdot)$ . Analiticamente, a ativação do neurônio é dada pôr;

$$x(t+h) = \phi [x(t), net_i(t), t, h] \quad (\text{A. 4})$$

onde observa-se que os estados futuros de um neurônio são afetados pelo estado atual do mesmo e pelo valor de  $net_i(t)$ , indicando uma 'memória' que implica em um neurônio dinâmico. Se a função  $\phi(\cdot)$  for constante, tem-se neurônios que não

possuem 'memória', ou seja, o estado atual é igual aos estados anteriores, o que indica um neurônio estático.

#### A.2.2.1.1.3. A Função de Saída $\lambda(.)$

Após a determinação do estado do neurônio  $x(t)$ , o sinal de saída do neurônio é calculado através da função de saída  $\lambda(.)$ . Matematicamente tem-se;

$$y(t) = \lambda [x(t), t] \quad (\text{A. 5})$$

onde, essencialmente, qualquer função contínua e monotonicamente crescente pode ser utilizada como função de saída<sup>4</sup>. Entretanto, existem algumas funções que são mais utilizadas na literatura; estas funções são :

- função linear :  $\lambda (x) = ax$
- função logística :  $\lambda(x) = \frac{1}{1 + e^{-\rho x}}$  , que é a função unipolar mais popular onde  $\rho$  é um fator de escala positivo
- função tangente hiperbólica :  $\lambda (x) = \frac{1 - e^{-\rho x}}{1 + e^{-\rho x}}$  , que é a função bipolar mais popular.

#### A.2.2.2. Dinâmica no Modelo do Neurônio

Como mostrado na seção anterior, o NA pode apresentar dinâmica, representada através da função de ativação. A equação (A.2) determina, segundo a nomenclatura da Teoria de Sistemas, uma relação não-linear, entretanto o neurônio pode apresentar uma função de ativação linear. A ativação linear no neurônio não implica em perda da capacidade de mapeamento não-linear da rede composta por estes neurônios porque a característica não-linear permanece na função de saída.

---

<sup>4</sup>É comum encontrar na literatura a descrição apenas de neurônios estáticos. Devido a este motivo, encontra-se com frequência o termo função de ativação como sendo a função que converte o valor de *net* no valor de saída do neurônio, o que é incompatível com as definições formais apresentadas neste texto.

Considerando-se o caso mais simples, onde não se tem qualquer estímulo externo, ou de outros neurônios, em um determinado neurônio cuja função de ativação é contínua no tempo e de primeira ordem; a ativação pode ser representada através de uma equação diferencial do tipo;

$$C_i \frac{dx_i(t)}{dt} = -D_i x_i(t) + R_i \quad (\text{A. 6})$$

onde  $\frac{C_i}{D_i}$  é a *constante de tempo* de resposta do neurônio e  $R_i$  equivale ao valor de equilíbrio para o qual tende a ativação do neurônio na ausência de sinais de entrada. A solução da equação (A.6) determina como o estado do neurônio evolui, isto é

$$x_i(t) = x_i(0)e^{-\frac{D_i}{C_i}t} + \frac{R_i}{D_i} \left( 1 - e^{-\frac{D_i}{C_i}t} \right) \quad (\text{A. 7})$$

onde  $x_i(0)$  é o estado inicial do neurônio.

Ao se considerar a dinâmica do neurônio como sendo discreta no tempo, utiliza-se as versões discretas da equação (A.6) e da equação (A.7), ou seja;

$$\bar{C}_i x_i(k+1) = -\bar{D}_i x_i(k) + \bar{R}_i \quad (\text{A. 8})$$

$$x(k) = \left( -\frac{\bar{D}_i}{\bar{C}_i} \right)^k x_i(0) + \sum_{i=k}^1 \left( -\frac{\bar{D}_i}{\bar{C}_i} \right)^{i-1} \left( \frac{\bar{R}_i}{\bar{C}_i} \right) \quad (\text{A. 9})$$

Como todo sistema dinâmico, o estado do neurônio dinâmico leva um certo tempo para estabilizar, tempo este que é denominado de *tempo de estabilização*. Este tempo de estabilização depende dos valores de  $D_i$  e  $C_i$ .

Do desenvolvimento acima observa-se que, apesar dos neurônios serem elementos computacionais extremamente interessantes, isoladamente os mesmos não são poderosos para processar ou representar conhecimento. Entretanto, se um

conjunto de neurônios é interconectado em uma rede de neurônios, denominada RNA, emerge uma série de propriedades computacionais não encontradas em um simples neurônio.

### A.3 Rede Neural Artificial

Como citado no final da seção anterior, pode-se dizer que uma RNA é um sistema composto pôr vários neurônios formando um sistema complexo. Com os neurônios sendo conectados pôr conexões sinápticas e normalmente agrupados em camadas.

Em uma RNA alguns neurônios recebem excitações do exterior e são denominados *neurônios de entrada*, formando a *camada de entrada* da RNA, e correspondem aos neurônios dos órgãos dos sentidos. Outros têm suas respostas usadas para alterar, de alguma forma, o mundo exterior e são denominados *neurônios de saída*, formando a *camada de saída* da RNA, e correspondem aos neurônios que excitam os músculos. Os neurônios que não são nem entrada nem saída são conhecidos como *neurônios internos* e forma a *camada intermediária* da RNA.

Para caracterizar uma RNA é importante especificar os seguintes pontos:

- Os componentes da rede : os neurônios.
- A resposta de cada neurônio.
- O estado global de ativação da rede.
- A conectividade da rede dada pelos valores de conexões sinápticas.
- Como se propaga a atividade da rede.
- Como se estabelece a conectividade da rede.
- O ambiente externo à rede.
- Como o conhecimento é representado na rede.

#### A especificação dos neurônios



É necessário determinar as características dos neurônios, que constituem a RNA, definindo o seu modelo. A rede é *homogênea* se todos os neurônios são iguais, ou *heterogênea* caso os neurônios sejam diferentes. Geralmente as RNA's são do tipo homogêneas, principalmente pôr razões de simplicidade, o que não ocorre com as redes biológicas que são bastantes heterogêneas.

### **A resposta de cada neurônio**

Cada neurônio da rede tem um sinal de saída, é excitado por outros neurônios ou por sinais externos e um valor de estado individual.

### **Ativação global da rede**

Em cada instante de tempo adotado, cada neurônio da rede está em um estado de ativação (ou excitação). O vetor de espaço  $\mathfrak{R}^n$  onde  $n$  é o número de neurônios da rede e que tem por componentes o estado de ativação individual de cada neurônio da rede, chama-se *estado de ativação* ou *padrão de ativação* da rede. Este padrão de ativação pode ser considerado com uma metáfora do 'pensamento' da rede em um dado momento.

### **Padrão de conectividade**

O padrão de conectividade define como os neurônios estão conectados. Um modo de representa-lo é através da *matriz de conectividade*  $W$ . Esta matriz se modifica durante o aprendizado da rede, normalmente, efetivado por um algoritmo iterativo.

### **Propagação da atividade da rede**

Em uma rede que os neurônios sejam sistemas a tempo contínuo a atividade da rede se propaga segundo a solução de equações diferenciais simultâneas. Embora seja normal representar os neurônios pôr equações a diferenças finitas, onde neste caso pode-se ter dois tipos de propagação da atividade da rede : síncrona e assíncrona. Propagação síncrona é quando a atividade de todos os

neurônios muda ao mesmo tempo sincronizados pôr um relógio, e pode ser considerada uma aproximação do sistema contínuo. Na propagação assíncrona os neurônios tem suas atividades modificadas em momentos que independem dos outros neurônios tal como ocorre nos sistemas vivos.

### **Como se estabelece a conectividade da rede**

A conectividade da rede permite representar o conhecimento que a rede necessita para a resolução de um problema específico. Frequentemente a conectividade se estabelece por um mecanismo de aprendizado, em que de modo iterativo, a rede se ajusta para representar um certo conjunto de conhecimentos. Esta não é, no entanto, a única maneira de estabelecer um padrão de conectividade, podendo os valores das conexões serem colocados diretamente representando conhecimentos.

### **O Ambiente**

É o ambiente que define o uso da RNA. Ao se utilizar uma RNA para o controle de um robô, será o processo robô que representará o ambiente em que a RNA é utilizada.

### **Como o conhecimento é representado na rede**

Existem dois modos de representar o conhecimento : localizado e distribuído. O conhecimento é localizado quando a um neurônio ou a uma sinapse corresponde um conhecimento determinado, e distribuído quando a cada conhecimento correspondem vários neurônios ou sinapses.

#### **A.3.1. Uma Definição de Rede Neural Artificial**

Para definir uma rede neural pode-se utilizar a mesma definição que é usada para um único neurônio. Neste caso, no entanto, tem-se a seguinte questão : com as definições anteriores não é possível representar de maneira explícita os pesos das conexões entre os diferentes neurônios da rede. Para resolver este problema,

podemos considerar uma rede neural como um Sistema Dinâmico Complexo seguindo a Definição A. 15. De modo a facilitar a representação de redes neurais como um sistema dinâmico complexo, é útil introduzir alguns conceitos da Teoria de Grafos.

Definição A. 32 - Um Grafo  $G$  consiste de um conjunto finito não vazio de vértices  $V = v_i$  e de um conjunto não-ordenado de arcos  $A$  que conectam alguns pares de vértices. Cada par de vértices  $v_i, v_j$  em  $V$  é um arco de  $G$  que 'liga'  $v_i$  e  $v_j$ .

Definição A. 33 - Um Grafo  $G$  é dito *Rotulado* quando os  $p$  vértices são distinguidos uns dos outros através de nomes ou rótulos.

Definição A. 34 - Um Grafo  $G$  é dito *Arco-Rotulado* quando também os arcos são distinguidos uns dos outros através de nomes ou rótulos.

Uma vez estabelecida a definição de Grafo, é possível definir uma RNA.

Definição A. 35 - Uma Rede Neural Artificial (RNA) é um Sistema Dinâmico Complexo representado pôr um grafo arco-rotulado no qual cada vértice é um Sistema Dinâmico denominado Neurônio Artificial (NA).

Uma vez definido formalmente o conceito de RNA, pode-se definir vários tipos de redes neurais, seguindo as definições apresentadas na seção A.1.1.

Definição A. 36 - Uma Rede Neural Artificial Contínua no Tempo é uma rede neural definida em um subconjunto do tempo  $T = \mathbb{R}$ .

Definição A. 37 - Uma Rede Neural Artificial Discreta no Tempo é uma rede neural definida em um subconjunto do tempo  $T = \mathbb{Z}$ .

Definição A. 38 - Uma Rede Neural Invariante no Tempo, também denominada de Rede Neural Estacionária é um rede neural em que a função de transição de estados  $\phi$  depende de apenas um elemento de  $T$  e a função de saída  $\lambda$  é independente de  $T$ .

### A.3.2. Classes e Topologias de Redes Neurais Artificiais

A maneira como os neurônios estão interconectados e as características dinâmicas dos neurônios formam as diferentes topologias de RNA's. Cada topologia apresenta o seu algoritmo de aprendizado e determina a capacidade da rede em resolver certas classes de problemas. A seguir são apresentadas duas classes de redes e duas das topologias mais comuns de RNA's.

#### A.3.2.1. Redes com uma Única Camada

Esta é a classe mais simples de RNA's em camadas. Neste caso tem-se apenas uma camada de neurônios de entrada que se conectam com os neurônios da camada de saída. Esta rede é denominada de *Única Camada*, ou *single-layer*, porque refere-se apenas aos neurônios da camada de saída, pois a função dos neurônios da camada de entrada é apenas de 'bufferização', não sendo portanto considerados.

A grande limitação desta classe de rede é a impossibilidade, demonstrada pôr Minsky e Papert (Minsky e Papert, 1969), de se resolver problemas classificados como linearmente não-separáveis. Apesar desta limitação, existe o Teorema do Perceptron que prova, que se uma solução existe, a rede eventualmente convergirá para esta solução (Rosenblatt, 1958).

#### A.3.2.2. Redes Multi-Camadas

Esta segunda classe de redes neurais, muitas vezes denominada de *multilayer*, se caracteriza pela presença de uma ou mais camada intermediária de neurônios. A função dos neurônios internos, pertencentes a camada intermediária, é extrair características dos neurônios da camada de entrada e fornecê-las como entrada para os neurônios da camada de saída. Deste modo, a rede é capaz de extrair estatísticas de mais alta ordem (Roisenberg, 1998).

Para esta classe de RNA existem trabalhos mostrando que redes neurais multi-camadas com funções de saída não-lineares, termos de *bias* e pesos de conexões ajustáveis, são capazes de aproximar, com a precisão que se deseje, qualquer função mensurável (Cybenko, 1988).

### A.3.2.3. Redes Diretas

Redes Diretas, ou topologias *Feedforward*, são aquelas cujo grafo não tem ciclos (De Azevedo, 1993). Frequentemente é comum representar estas redes em camadas, com camadas de entrada, de saída e intermediária<sup>5</sup>, como mostrado na Figura A. 6.

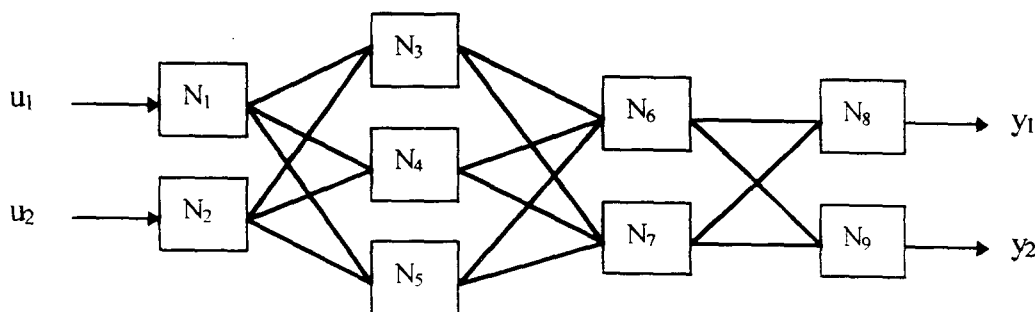


Figura A. 6 - RNA Direta com quatro camadas de neurônios.

Estas redes, atualmente, são as mais utilizadas, isto ocorre, principalmente por existirem fáceis algoritmos de aprendizado para estes tipos de RNA's. Um dos algoritmos mais usado é o *backpropagation*, sendo que muita vezes este tipo de rede é denominada, impropriamente, de *rede backpropagation*. Outro motivo que tornou as redes diretas bastantes populares, é a sua capacidade de aproximar, com maior ou menor precisão, dependendo do número de neurônios da rede, qualquer função não-linear (Funahashi, 1989).

<sup>5</sup>Em alguns trabalhos encontrados na literatura é comum se referir a "camada" como sendo uma camada de conexões. Neste texto camada se refere sempre à camada de neurônios, e considerar-se-a os neurônios de entrada apenas um *buffer*.

#### A.3.2.4. Redes Recorrentes

A topologia de RNA's com ciclos, ou com realimentação, ou com retroação, ou com *feedback*, são aquelas cujo grafo de conectividade contém ao menos um ciclo. Quando além disto envolvem neurônios dinâmicos, são denominadas *recorrentes*.

Uma rede recorrente pode ser composta pôr uma única camada de neurônios em que cada neurônio fornece o seu sinal de saída como entrada para cada um dos outros neurônios, como visto na Figura A. 7. A estrutura descrita na Figura A. 7 não possui camada intermediária. Além disto as malhas de realimentação envolvem apenas a utilização de ramos particulares, compostos pôr elementos de atraso unitário (representado pôr  $z^{-1}$ , que é um operador da Transformada-Z) que resulta em um comportamento dinâmico não-linear, em virtude da natureza não-linear dos próprios neurônios.

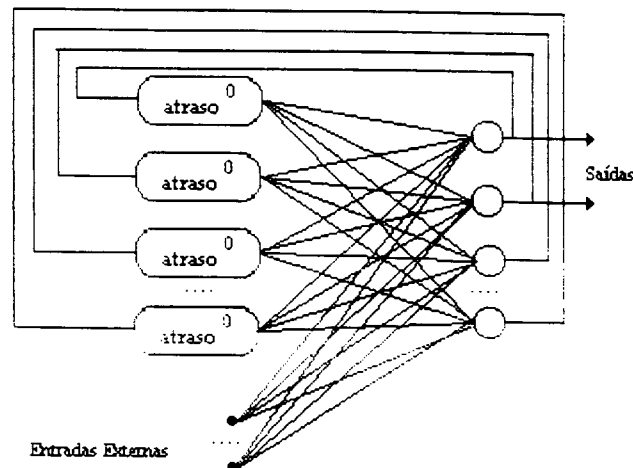


Figura A. 7 - RNA Recorrente *Single-Layer*.

Esta característica de dinâmica não-linear, das redes recorrentes, permite que esta topologia seja utilizada em problemas e aplicações que exijam representação de estados, como por exemplo : controle industrial, filtragem adaptativa, predição de séries temporais, etc. Atualmente, as RNA's recorrentes têm sido objeto de inúmeros

estudos e em um futuro próximo, acredita-se que suplantará em utilização as RNA's diretas, justamente pela sua facilidade de representar sistemas dinâmicos.

A literatura apresenta diferentes topologia de redes, onde cada autor apresenta o modelo e solução para determinados problemas. Uma boa referência é o trabalho de Giles, Kuhn e Williams (Giles et al, 1994) onde encontra-se uma boa fundamentação teórica para este tipo de rede.

## REFERÊNCIAS BIBLIOGRÁFICAS

- [1] ABDALLAH, C., DAWSON, D., JAMSHIDI, M., Survey of Robust Control for Rigid Machines. **IEEE Cont. Syst. Mag.**, February, pp. 24-30, 1991.
- [2] ABDULAZIZ, A., FARSI, M., Non-Linear System Identification and Control Based on Neural and Self-Tuning Control. **International Journal of Adaptive Control and Signal Processing**, Vol. 7, pp. 297-307, 1993.
- [3] AL-DUWAISH, H., KARIM, M. N., CHANDRASEKAR, V., Use of Multilayer Feedforward Neural Networks in Identification and Control of Wiener Model. **IEE Proc.-Control Theory Appl.**, Vol. 143, No. 3, pp. 255-258, 1996.
- [4] ANDERSON, J. A., Neural Models with cognitive implications. In D. LaBerge and S. J. Samuels, eds. **Basic processes in reading perception and comprehension**. Hillsdale, NJ: Erlbaum, 1977.
- [5] ANTSAKLIS, P. J., Defining Intelligent Control – Report of the Task Force on Intelligent Control. **IEEE Control Systems Magazine**, Vol. 14, No. 3, pp. 4-66, 1994.
- [6] ÅSTRÖM, K. J., **Introduction to Stochastic Control Theory**. Academic Press. New York, 1970.
- [7] ÅSTRÖM, K. J., Toward Intelligent Control. **IEEE Control Systems Magazine**, 9:(4), pp. 60-69, 1989.
- [8] ÅSTRÖM, K. J., ANTON, J. J., ÅRZÉN, K.-E., Expert Control. **Automatica**, 22:(3), pp. 277-286, 1986.
- [9] ÅSTRÖM, K. J., WITTENMARK, B., **Adaptive Control**. NY, CA: Addison-Wesley, 1995.



- 
- [10] ÅSTRÖM, K. J., WITTENMARK, B., **Computer Controlled Systems**. Prentice-Hall, Englewood Cliffs, New Jersey, 1997.
- [11] ATHANS, M., FALB, P. L., **Optimal Control : An Introduction to the Theory and Its Applications**. New York : McGraw-Hill, 1966.
- [12] BAI, E.-W., An Optimal Two-Stage Identification Algorithm for Hammerstein-Wiener Nonlinear Systems. *Automatica*, Vol. 34, No. 3, pp. 333-338, 1998.
- [13] BARREIROS, J. A. L., **Métodos de Controle Adaptativo Auto-Ajustável Aplicados à Síntese de Estabilizadores de Sistemas de Potência**. Florianópolis, 1995. Tese (Doutorado em Engenharia Elétrica, área de concentração Sistemas de Potência) – Centro Tecnológico, Universidade Federal de Santa Catarina.
- [14] BARRETO, J. M., Modeling and Simulation in Biomedicine. Text of the Courses **Applications Biomédicales des Méthodes de Simulation I et II**, UCL, Belgium, 1992.
- [15] BARRETO, J. M., **Conexionismo e a Resolução de Problemas**. Florianópolis, 1996. Monografia (Concurso público para Professor Titular - Departamento de Informática e Estatística) - Centro Tecnológico, Universidade Federal de Santa Catarina.
- [16] BARRETO, J. M., AND PROYCHEV, T., **Control of the Standing Position**. Bruxelas, 1994. Relatório Técnico (Projeto MUCON "Multisensory Control of Movement" do Programa ESPRIT de Pesquisa Básica) - CEE (Comissão Econômica Européia), Lab. of Neurophysiology, UCL.
- [17] BARS, R., BÉZI, I., PILIPÀR, B., OJHELY, B., Nonlinear and Long-Range Control of a Distillation Pilot Plant. **Preprints of 9th IFAC/IFORS Symp. in Identification and Syst. Parameter Estimation**, Vol. 2, pp. 848-853. Budapest, Hungary, 1990.
- [18] BEGHELLI, S., GIUDORZI, R., Bilinear Systems Identification from Input-Output

- 
- Sequences. **4th IFAC Symp. Ident. & Syst. Par. Est.**, Tbilisi, pp. 1759-1764, 1976.
- [19] **BELLMAN, R., Dynamic Programming.** Princeton University Press, Princeton, New Jersey, 1957.
- [20] **BILLINGS, S. A., VOON, W. S. F., Structure Detection and Model Validity Tests in the Identification of Nonlinear Systems. Proc. IEE Pt. D, Vol. 130, pp. 193-199, 1983.**
- [21] **BILLINGS, S. A., Identification of Nonlinear Systems - A Survey. Proc. IEE, Pt. D, Vol. 127, No. 6, pp. 272-284, 1980.**
- [22] **BILLINGS, S. A., CHEN, S., Chaper 11 - Neural Networks and System Identification. In G. W. Irwin, K. Warwick and K. J. Hunt, eds. Neural Networks Application in Control, IEE Control Engineering Series 53, The Institution of Electrical Engineers, London, United Kingdom, 1995.**
- [23] **BILLINGS, S. A., FAKHOURI, S. Y., Identification of a Class of Nonlinear Systems using Correlation Analysis. Proc. IEE, 125, pp. 691-697, 1978.**
- [24] **BILLINGS, S. A., FAKHOURI, S. Y., Identification of Factorable Volterra Systems. Proc. IEE, 126, (10), pp. 1018-1024, 1979.**
- [25] **BILLINGS, S. A., FAKHOURI, S. Y., Identification of Systems Containing Linear Dynamic and Static Nonlinear Elements. Automatica. Vol. 18, N° 1, pp. 15-26, 1982.**
- [26] **BILLINGS, S. A., JAMALUDDIN, H. B., CHEN, S. A., Comparison of the Backpropagation and Recursive Prediction Error Algorithms for Training Neural Networks. Mechanical Systems and Signal Processing, Vol. 5, pp. 233-255, 1991.**
- [27] **BILLINGS, S. A., JAMALUDDIN, H. B., CHEN, S., Properties of Neural Networks**

- 
- with Applications to Modelling Non-Linear Dynamical Systems. **Int. J. Control**, Vol. 55, No. 1, pp. 193-224, 1992.
- [28] BILLINGS, S. A., VOON, W. S. F., Correlation Based Model Validity Tests for Nonlinear Models. **Int. J. Control**, Vol. 44, No. 1, pp. 235-244, 1986.
- [29] BISSEL, C., Six Decades in Control - An Interview with Winfried Oppelt. **IEE Review**, January, pp. 17-21, 1992.
- [30] BITTANTI, S., PIRODDI, L., Neural implementation of GMV control schemes based on affine input/output models. **IEE Proc. – Control Theory Appl.**, Vol. 144, No. 6, pp. 521-530, 1997.
- [31] BLACK, H. S., Stabilized Feedback Amplifiers. **Bell System Technical Journal**, 1934.
- [32] BODE, H. W., **Network Analysis and Feedback Amplifier Design**. Van Nostrand, New York, 1945.
- [33] BRASIL, L. M., **Proposta de Arquitetura para Sistema Especialista Híbrido e a Correspondente Metodologia de Aquisição do Conhecimento**. Florianópolis, 1999. Tese (Doutorado em Engenharia Elétrica, área de concentração Sistemas de Informação) – Centro Tecnológico, Universidade Federal de Santa Catarina.
- [34] BRINKER, A. C. Den, A Comparison of Results from Parameter Estimation of Impulse Responses of the Transient Visual System. **Biol. Cybern.**, Vol. 61, pp. 139-151, 1989.
- [35] BROCKETT, R. W., Poles, Zeros and Feedback : State Space Interpretation. **IEEE Transactions on Automatic Control**, Vol. AC-10, pp. 129-135, 1965.
- [36] BROGAN, W. L., **Modern Control Theory**. Prentice-Hall Inc., Englewood Cliffs, New Jersey, 1985.
- [37] CAMPOS, G. A. L., Planejamento de Trajetórias de Robôs Utilizando Rede de

- Neurônios Artificiais. Uberlândia, 1990. Dissertação (Mestrado em Engenharia Elétrica, Departamento de Engenharia Elétrica) - Universidade Federal de Uberlândia.
- [38] CHEN, C.-T., **Linear System Theory and Design**. Holt, Rinehart and Winston, Holt-Saunders Japan, 1984.
- [39] CHEN, F., KHALIL, H. K., Adaptive Control of Non-Linear Systems using Neural Networks. *Int. J. Control*, 55, (6), pp. 1299-1317, 1992.
- [40] CHEN, S., BILLINGS, S., Neural Networks for Nonlinear Dynamic System Modelling and Identification. *Int. J. Control*, 56(2):319-346, 1992.
- [41] CHEN, S., BILLINGS, S. A., COWAN, C. F., GRANT, P. M., Practical Identification of NARMAX Models using Radial Basis Functions. *Int. J. Control*, 52: 1327-1350, 1990(a).
- [42] CHEN, S., BILLINGS, S., GRANT, P., Non-Linear System Identification Using Neural Networks. *Int. J. Control*, 51(6), pp:1191-1214, 1990(b).
- [43] CHEN, S., COWAN, C. F. N., BILLINGS, S. A., GRANT, P. M., Parallel Recursive Prediction Error Algorithm for Training Layered Neural Networks. *Int. J. Control*, Vol. 51, pp. 1215-1228, 1990(c).
- [44] CHESTER, D. L., Why Two-Hidden Layers are Better than One ?. In **Proceedings of the International Joint Conference on Neural Networks IJCNN'90**. Edited by Caudill, M., pp. A265-A268, Lawrence Erlbaum Associates, 1990.
- [45] CICHOCKI, A., UNBEHAUEN, R., **Neural Networks for Optimization and Signal Processing**. John Wiley & Sons Ltd. & B. G. Teubner. Stuttgart. German, 1993.
- [46] COELHO, A. A. R., Controle Adaptativo para Processos Multivariáveis : Aspectos Teóricos e Simulação. Campinas, 1991. Tese (Doutorado em Engenharia Elétrica, Departamento de Computação e Automação) - Faculdade de Engenharia Elétrica,

Unicamp.

- [47] CRUZ, J. C., AGUIRRE, L. A., JARDIN, E. M., Metodologia para Identificação de um Processo Químico com Rede Neural Artificial. **Anais do XII Congresso Brasileiro de Automática**, Vol. V, pp. 1785-1790, Uberlândia, MG, Brasil, 1998.
- [48] CYBENKO, G., **Continuous Valued Networks With Two Hidden Layers are Sufficient**. Technical Report, Tufts University, Department Science, 1988.
- [49] CYBENKO, G., Approximation by Superposition of a Sigmoidal Function. **Mathematics of Control, Signals, and Systems**, 2:303-314, 1989.
- [50] D'AZZO, J. J., HOUPIS, C. H., **Linear Control System Analysis and Design - Conventional and Modern**. McGraw-Hill International Editions, 3rd. Edition, 1988.
- [51] DAYHOFF, J., **Neural Networks Architectures - An Introduction**. Van Nostrand Reinhold, 1990.
- [52] DE AZEVEDO, F. M., **Contribution to the Study of Neural Networks in Dynamical Expert System**. Ph.D. Thesis, Institut d'Informatique, FUNDP, Namur, Belgium, 1993.
- [53] DE AZEVEDO, F. M., BARRETO, J. M., IMC Scheme Using Neural Networks for Robot Arm. **Proc. of the X Brazilian Automatic Congress**. Vol. II. pp. 891-986, Rio de Janeiro, Brazil, 1994.
- [54] DE OLIVEIRA, R. C. L., **Arquitetura de Controle Utilizando Modelos de Redes Neurais**. São José dos Campos, 1991. Dissertação (Mestrado em Engenharia Eletrônica e Computação, Divisão de Engenharia Eletrônica) - Instituto Tecnológico de Aeronáutica. CTA.
- [55] DE OLIVEIRA, R. C. L., DE AZEVEDO, F. M., BARRETO, J. M., Rede Neural com Dinâmica Linear Representada no Espaço de Estados. **Anais do III Congresso Brasileiro de Redes Neurais - III CBRN**, Florianópolis, SC, pp. 115-120, 1997(a).

- 
- [56] DE OLIVEIRA, R. C. L., DE AZEVEDO, F. M., BARRETO, J. M., Identificação Neural Não-Linear Utilizando Neurônios Dinâmicos. **Anais do III Simpósio Brasileiro de Automação Inteligente - III SBAI**, Vitória, ES, pp. 418-423, 1997(b).
- [57] DE OLIVEIRA, R. C. L., DE AZEVEDO, F. M., BARRETO, J. M., Dynamic Neural Net in the State Space Utilized in Non-Linear Processes Identification. **Lectures Notes in Artificial Intelligence - Artificial Neural Nets and Genetic Algorithms**, Springer Verlag, London, pp. 588-591, 1997(c).
- [58] DE OLIVEIRA, R. C. L., DE AZEVEDO, F. M., BARRETO, J. M., Utilização de Redes Neurais Artificiais em Sistemas de Controle, *Pesquisa Naval*, **ISSN 1414-8595**, **Suplemento Especial da Revista Marítima Brasileira**, No. 11, Outubro, pp. 235-257, 1998.
- [59] DE OLIVEIRA, R. C. L., NASCIMENTO JR., C. L., YONEYAMA, T., A Fault Tolerant Controller Based on Neural Nets, **Proc. of the IEE Int. Conf. on Control'91**, Vol. 1, pp. 399-404, Edinburgh, UK, 1991.
- [60] DELGADO, A., KAMBHAMPATI, C., WARWICK, K., Dynamic Recurrent Neural Network for System Identification and Control. **IEE Proc.-Control Theory Appl.**, Vol. 142, No. 4, pp. 307-314, 1995.
- [61] DERFFNER, G., Neural Networks for Time Series Processing. **Neural Network World**, 6. (4), pp. 447-468, 1996.
- [62] DIAS, J. S., **Sensibilidade Paramétrica como Guia para o Treinamento Híbrido de Redes Neurais**. Florianópolis, 1999. Tese (Doutorado em Engenharia Elétrica. área de concentração Sistemas de Informação) – Centro Tecnológico, Universidade Federal de Santa Catarina.
- [63] DOEBLIN, E. O., **Measurement Systems**. McGraw Hill Int., Auckland. New Zealand, 1983.

- 
- [64] DORATO, P., YEDAVALLI, R. K., (Eds.), **Recent Advances in Robust Control**. New York : IEEE Press, 1990.
- [65] DOYLE, J. C., GLOVER, K., KHARGONEKAR, P. P., FRANCIS, B. A., State Space Solutions to Standard  $H_2$  and  $H_\infty$  Control Problems. **IEEE Transactions on Automatic Control**, Vol. 34, pp. 831-847, 1989.
- [66] DUNOYER, A., BALMER, L., BURNHAM, K. J., JAMES, D. J. G., On the Discretization of Single-Input Single-Output Bilinear Systems. **Int. J. Control**, Vol. 68, No. 2, pp. 361-372, 1997.
- [67] ELMAN, J. L., Finding Structure in Time. **Cognitive Science**, 14, pp. 179-211, 1990.
- [68] EVANS, W. R., Control Systems Synthesis by Root Locus Method. **AIEE Transactions**, Part II, Vol. 69, pp. 66-69, 1950.
- [69] FARRELL, J., BERGER, T., APPLEBY, B., Using Learning Techniques to Accommodate Unanticipated Faults. **IEEE Contr. Syst. Mag.**, June, pp. 40-49, 1993.
- [70] FENG, G., Stable Identification of Nonlinear Dynamic Systems using RBF Nets. **Proceedings of the 12th IFAC World Congress**, Vol. 9. pp. 269-272, Sydney, Australia, 1993.
- [71] FILHO, P. R. B., BARATA, A. J. A., SOARES, R. P. O., DE OLIVEIRA, R. C. L., GARCEZ, J. N., Um Estabilizador PID Inteligente para Sistemas de Potência utilizando Redes Neurais Artificiais. **Anais do X Congresso Brasileiro de Automática**. Vol. I, pp. 412-417, Rio de Janeiro, RJ, Brasil, 1994.
- [72] FU, K. S., Learning Control Systems - Review and Outlook. **IEEE Trans. Automat. Contr.**, pp. 210-221, April, 1970.
- [73] FULLER, A. T., The Early Development of Control Theory. **Transactions of the**

- 
- ASME (Journal of Dynamic Systems, Measurement & Control)**, Vol. 96G, pp. 109-118, 1976.
- [74] FUNAHASHI, K., On the Approximate Realization of Continuous Mappings by Neural Networks. **Neural Networks**, 2, pp. 183-192, 1989.
- [75] GALLMAN, P. G., A Comparison of Two Hammerstein Model Identification Algorithms. **IEEE Transactions on Automatic Control**, Vol. AC-21, pp. 124-126, February, 1976.
- [76] GILES, C., KUHN, G., WILLIAMS, R., Dynamic Recurrent Neural Networks : Theory and Applications. **IEEE Transactions on Neural Networks**, Vol. 5, No. 2, pp. 152-156, 1994.
- [77] GIROSI, T., POGGIO, T. Representation Properties of Networks : Kolmogorov's Theorem is Irrelevant. **Neural Computation**, 1:405-469, 1989.
- [78] GOODWIN, G. C., MAYNE, D. Q., A parameter estimation perspective of continuous time model reference adaptive control. **Automatica**, Vol. 23, No.1, pp. 57-70, 1987.
- [79] GOODWIN, G. C., PAYNE, R. L., **Dynamic System Identification : Experiment Design and Data Analysis**. New York : Academic Press. 1977.
- [80] GREBLICKI, W., Nonparametric Identification of Wiener Systems. **IEEE Trans. on Information Theory**, Vol. 38, No. 5, pp. 1487-1493, 1992.
- [81] GREBLICKI, W., PAWLAK, M., Hammerstein System Identification by Non-Parametric Regression Estimation. **Int. J. Control**, Vol. 45, No. 1, pp. 343-354, 1987.
- [82] GUPTA, M. M., (Ed.), **Adaptive Methods for Control Systems Design**. New York : IEEE Press. 1986.
- [83] GUPTA, M. M., SINHA, N. K., (Eds.), **Intelligent Control Systems - Theory and**



- Applications.** IEEE Press, 1996.
- [84] GUYTON, A., **Textbook of Medical Physiology.** W. B. Sanders Company, Philadelphia, 1976.
- [85] HABER, R., UNBENHAUEN, H., Structure Identification on Nonlinear Dynamic Systems – A Survey of Input-Output Approaches. **Automatica**, Vol. 26, pp. 651-677, 1990.
- [86] HANDELMAN, D. A., LANE, S. H., Chapter 8 - Human-to-Machine Skill Transfer Through Cooperative Learning. **Intelligent Control Systems - Theory and Applications**, Ed. by M. M. Gupta e N. K. Sinha, IEEE Press, 1996.
- [87] HANG, C. C., HO, W. K., LEE, T. H., Chapter 14 - Knowledge-Based PID Control - Heuristics and Implementation. **Intelligent Control Systems - Theory and Applications**, Ed. by M. M. Gupta e N. K. Sinha, IEEE Press, 1996.
- [88] HARP, S. A., SAMAD, T., Genetic Synthesis of Neural Network Architecture. In **Advances in Neural Information Processing**, Vol. 2, Ed. by L. D. Davis, pp. 202-221, New York, Van Nostrand Reinhold, 1991.
- [89] HAYKIN, S., **Neural Networks - A Comprehensive Foundation.** IEEE Press, New York, 1994.
- [90] HEBB, D. O., **The Organization of Behavior.** John Willey, New York, 1949.
- [91] HECHT-NIELSEN, R., Kolmogorov's Mapping Neural Networks Existence Theorem. In **Proceedings of The International Joint Conference on Neural Networks IJCNN'87**, Vol. 3, pp. 11-14, 1987.
- [92] HERMANN, R., KRENER, A. J., Non-Linear Controllability and Observability. **IEEE Trans. on Automatic Control**, 22, pp:728-748, 1977.
- [93] HOPFIELD, J. J., Neural Networks and Physical Systems with Emergent Collective Computational Abilities. **Proc. of the National Academy of Sciences**,

- 79, pp. 2554-2558, 1982.
- [94] HOPFIELD, J. J., Neurons with graded response have collective computational properties like those of two-state neurons. **Proc. of the National Academy of Sciences**, 81, pp. 3088-3092, USA, 1984.
- [95] HORNE, B. G., GILES, C. L., An Experimental Comparison of Recurrent Neural Networks. **Neural Information Processing Systems 7**, Tesauro, G., Touretzky, D. and Leen, T., Editors. MIT Press, pp. 697-705, 1995.
- [96] HORNIK, K., STINCHCOMBE, M., WHITE, H. Multilayer Feedforward Networks are Universal Approximators. **Neural Networks**, 2:359-366, 1989.
- [97] HUNT, K. J., SBARBARO, D., Neural Networks for Nonlinear Internal Model Control. **IEE Proceedings - Part D**, Vol. 138, No. 5, pp. 431-438, 1991.
- [98] HUNT, K. J., SBARBARO, D., Chapter 6 - Studies in Artificial Neural Networks based Control. **Neural Networks Application in Control**, Edited by G. W. Irwin, K. Warwick and K. J. Hunt, IEE Control Engineering Series 53, The Institution of Electrical Engineers, London, United Kingdom, 1995.
- [99] HUNT, K. J., SBARBARO, D., ZBIKOWSKI, R., GAWTHROP, P. J., Neural Networks for Control Systems - A Survey. **Automatica**, Vol. 28, No. 6, pp. 1083-1112, 1992.
- [100] HUNTER, I. W., KORENGERG, M. J., The Identification of Nonlinear Biological Systems : Wiener and Hammerstein Cascade Models. **Biol. Cybern.**, 55, 135, 1986.
- [101] HWANG, C., SHYU, K.-K., Series Expansion Approach to the Analysis and Identification of Discrete Hammerstein Systems. **Int. J. Control**, Vol. 47, No. 6, pp. 1961-1972, 1988.
- [102] **IEEE Control Systems Magazine**, Special section on neural networks for control

---

systems, Vol. 8, No. 2, 1988; Special section on neural networks for control systems, Vol. 9, No. 2, 1989; Special section on neural networks for control systems, Vol. 10, No. 3, 1990.

- [103] IFAC Special Issue on Identification and System Parameter. **Automatica**, Vol. 17, 1981.
- [104] IEEE Special Issue on Linear Multivariable Control Systems. **IEEE Transactions on Automatic Control**, Vol. AC-26, 1981.
- [105] IEEE Special Issue on Linear-Quadratic-Gaussian Problem. **IEEE Transactions on Automatic Control**, Vol. AC-16, December, 1971.
- [106] IRWIN, G. H., WARWICK, K., HUNT, K. J., (Eds.) **Neural Networks Application in Control**. IEE Control Engineering Series 53, The Institution of Electrical Engineers, London, UK, 1995.
- [107] JOHANSSON, R., **System Modeling & Identification**. Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1993.
- [108] JORDAN, M. I., **Serial Order : A Parallel Distributed Processing Approach**, Report No. 8604, Institute for Cognitive Science, University of California, La Jolla, CA, USA, 1986.
- [109] JORDAN, M. I., JACOBS, R., Learning to Control and Unstable System with Forward Modeling. **Advances in Neural Information Processing Systems 2**, D. Touretzky, Ed., Morgan Kaufmann, San Mateo, CA, 1990.
- [110] JORDAN, M. I., RUMELHART, D. E., Forward Models : Supervised Learning with a Distal Teacher. **Cognitive Science**, 16, No. 3, pp:307-354, 1992.
- [111] JURY, E. I., **Theory and Application fo the Z-Transform Method**. John Wiley, New York, 1964.
- [112] KAILATH, T., **Linear Systems**. Englewood Cliffs, N.J.: Prentice-Hall, Inc, 1980.

- 
- [113] KALMAN, R. E., Contributions to the Theory of Optimal Control. **Proceedings 1959 Mexico City Conference on Differential Equations**, Mexico City, pp. 102-119, 1960.
- [114] KALMAN, R. E., BUCY, R. S., New Results in Linear Filtering and Prediction Theory. **Transactions of the ASME (Journal of Basic Engineering)**, Vol. 83D, pp. 95-108, 1961.
- [115] KALMAN, R. E., FALB, P. L., ARBIB, M. A., **Topics in Mathematical System Theory**. McGraw-Hill, 1969.
- [116] KARAKASOGLU A., SUDHARSANAN, S. I., SUNDARESHAN, M. K., Identification and Decentralized Adaptive Control using Dynamical Neural Networks with Application to Robotic Manipulators. **IEEE Transactions on Neural Networks**, Vol. 4, No. 6, pp. 919-929, 1993.
- [117] KAWATO, M., FURUKAWA, K., SUZUKI, R., A Hierarchical Neural-Network Model for Control and Learning of Voluntary Movement. **Biological Cybernetics**, 57, pp. 169-185, 1987.
- [118] KIRK, D. E., **Optimal Control Theory - An Introduction**. New Jersey: Prentice-Hall, 1970.
- [119] KRISTINSSON, K., DUMONT, G. A., System Identification and Control Using Genetic Algorithms. **IEEE Trans. on Systems, Man, and Cybernetics**, Vol. 22, No. 5, pp. 1033-1046, 1992.
- [120] KROSE, B., VAN DER SMAGT, P., **Perceptrons : An Introduction to Neural Networks**. The University of Amsterdam, 1993.
- [121] KU ,C. C., LEE, K. Y., Diagonal Recurrent Neural Networks for Dynamical Systems Control. **IEEE Transactions on Neural Networks**, Vol. 6, No. 1, pp. 144-156, 1995.

- 
- [122] KURDILA, A. J., NARCOWICH, F. J., WARD, J. D., Persistency of Excitation in Identification using Radial Basis Function Approximants. **SIAM J. Control and Optimization**. Vol. 33, No. 2, pp. 625-642, 1995.
- [123] LANDAU, I. D., Evolution of Adaptive Control. **Journal of Dynamic Systems, Measurement, and Control**. 115 : 381-391, 1993.
- [124] LEE, C., Fuzzy Logic in Control Systems : Fuzzy Logic Controller, Part I and Part II. **IEEE Transactions on Systems, Man, and Cybernetics**, Vol. 20, No. 2, pp. 404-435, 1990.
- [125] LEVIN, A. U., NARENDRA, K. S., **Control of nonlinear dynamical systems using neural networks. Part II : Identification**. Technical Report No. 9116, Center for Systems Science, Yale University, New Haven, CT, 1991.
- [126] LEVIN, A. U., NARENDRA, K. S., Control of nonlinear dynamical systems using neural networks : controllability and stabilization. **IEEE Trans. on Neural Networks**, Vol. 4, No. 2, pp. 192-206, 1993.
- [127] LEVIN, U., NARENDRA, K. S., Control of Nonlinear Dynamical Systems using Neural Networks - Observability, Identification and Control. **IEEE Trans. on Neural Networks**, Vol. 7, No. 1, pp. 30-42, 1996.
- [128] LEVINE, W., Ed., **Control Handbook**. Boca Raton, FL: CRC Press, 1996.
- [129] LINKENS, D. A., NYONGESA, H. O., Learning systems in intelligent control : an appraisal of fuzzy, neural and genetic algorithms control applications. **IEE Proc.-Control Theory Appl.**, Vol. 143, No. 4, pp. 367-386, 1996.
- [130] LIU, X., CELIKOVSKÝ, S., Feedback Control of Affine Nonlinear Singular Control Systems. **Int. J. Control**. Vol. 68, No. 4, pp. 753-774, 1997.
- [131] LJUNG, L., **System Identification : Theory for the user**. Prentice-Hall, Englewood Cliffs, NJ, 1987.

- 
- [132] LJUNG, L., Issues in System Identification. **IEEE Control Systems Magazine**, 11(1):25-29, 1991.
- [133] LJUNG, L., SÖDERSTRÖM, T., **Theory and Practice of Recursive Identification**. Cambridge, MIT Press, 1983.
- [134] LJUNG, L., WAHLBERG, B., Asymptotic Properties of the Least-Squares Method for Estimating Transfer Functions and Disturbance Spectra. **Adv. Appl. Prob.**, 24:412-440, 1992.
- [135] LUENBERG, D. G., An Introduction to Observers. **IEEE Transactions on Automatic Control**, Vol. AC-16, pp. 596-602, 1971.
- [136] MANDANI, E. H., Application of Fuzzy Algorithms for Control of Simple Dynamic Plant. **IEEE Proceedings**, 121:(12), 1974.
- [137] MAXWELL, J. C., On Governors. **Philosophical Magazine**, Vol. 35, pp. 385-398, 1868.
- [138] McKELVEY, T., **Identification of State-Space Models from Time and Frequency Data**. Ph.D. Thesis. Division of Automatic Control, Linköping University, 1995.
- [139] MINSKY, M. L., PAPERT, S. A., **Perceptrons : An Introduction to Computational Geometry**. The MIT Press, 1969.
- [140] MIYAMOTO, H., KAWATO, M., SETOYAMA, T., SUZUKI, R., Feedback-Error-Learning Neural Networks for Trajectory Control of a Robotic Manipulator. **Neural Networks**, 1, pp. 251-265, 1988.
- [141] MONTANA, D. J., DAVIS, L., Training Feedforward Neural Networks Using Genetic Algorithms. **Proceedings of the 11th Int. Joint Conf. on Artificial Intelligence (IJCAI-89)**, August, Detroit, USA, pp. 762-767, San Mateo, USA: Morgan Kaufmann, 1989.

- 
- [142] MOODY, J., The effective Number of Parameters : An Analysis of Generalization and Regularization in Nonlinear Learning Systems. **Advances in Neural Information Processing Systems 4**, Moody, J., Hanson, S., and Lippman, R., Editors. Morgan Kaufmann Publishers, San Mateo, CA, 1992.
- [143] NARENDRA, K. S., ANNASWAMY, A. M., **Stable Adaptive Systems**. Prentice-Hall, Englewood Cliffs, NJ, 1989.
- [144] NARENDRA, K. S., LEVIN, A. U., Recursive Identification Using Feedforward Neural Networks. **Int. J. Control**, Vol. 61, No. 3, pp. 533-547, 1995.
- [145] NARENDRA, K. S., PARTHASARATHY, K., Identification and Control of Dynamic Systems using Neural Networks. **IEEE Trans. on Neural Networks**, Vol. 1, No. 1, pp. 4-27, 1990.
- [146] NASCIMENTO Jr., C. L., **Artificial Neural Networks in Control and Optimization**. Ph.D. Thesis in Electrical Engineering, Control Systems Centre, Faculty of Technology, University of Manchester, UK, 1994.
- [147] NAUCK, D., KLAWONN, F., KRUSE, R., **Foundations of Neuro-Fuzzy Systems**. John Wiley & Sons, NY, 1997.
- [148] NETO, W. R., NASCIMENTO Jr. C. L., GÓES, L. C. S., Controle Adaptativo Inverso usando Feedback-Error-Learning. **Anais do XII Congresso Brasileiro de Automática**. Vol. I, pp. 351-356, Uberlândia, MG, Brasil, 1998.
- [149] NG, G. W., **Adaptive Control Using Neural Networks - A Survey**. Control Systems Centre Report Number 813, Control Systems Centre. UMIST, Manchester, UK, 1994.
- [150] NGUYEN, D. H., WIDROW, B., Neural Networks for Self-Learning Control Systems. **IEEE Contr. Syst. Mag.**, April, pp. 18-23, 1990.
- [151] NORIEGA, J. R., WANG, H., A Direct Adaptive Neural-Network Control for

- 
- Unknown Nonlinear Systems and Its Application. **IEEE Transactions on Neural Networks**, Vol. 9, No. 1, pp. 27-34, 1998.
- [152] NORTON, J. P., **An Introduction to Identification**. Academic Press Inc. Ltd., London, 1986.
- [153] NYQUIST, H., **Regeneration Theory**. **Bell System Technical Journal**, Vol. 11, pp. 126-147, 1932.
- [154] ORTEGA, R., TANG, Y., **Robustness of Adaptive Controllers - A Survey**. **Automatica**, Vol. 25, No. 5, pp. 651-677, 1989.
- [155] OTTO, M., **Origins of Feedback Control**. MIT Press, Cambridge, Massachusetts, 1971.
- [156] PAJUNEN, G., **Application of Model Reference Adaptive Technique to the Identification and Control of Wiener type Nonlinear Processes**. Doctoral Dissertation, Depart. of Electrical Engineering, Helsinki University of Technology, Helsinki, Finland, 1994.
- [157] PARASKEVOPOULOS, P. N., **Digital Control Systems**. Prentice-Hall Europe, 1996.
- [158] PASSINO, K. M., Chapter 1 - **Toward Bridging the Perceived Gap Between Conventional and Intelligent Control**. **Intelligent Control Systems - Theory and Applications**, Ed. by M. M. Gupta e N. K. Sinha, IEEE Press, 1996.
- [159] PASSINO, K. M., LUNARDHI, A. D., **Stability Analysis of Expert Control Systems**. In **Proc. of The IEEE Conf. on Decision and Control**, San Antonio, TX, pp. 765-770, 1993.
- [160] PEDRYCZ, W., **Neural Networks : Concepts and Architectures**. **SBA Controle & Automação**, Vol. 4, No. 3, pp. 126-140, 1994.
- [161] PETRIDIS, V., KAZARLIS, S., PAPAICONOMOU, A., **A hybrid genetic algorithm**



- for training neural networks. **Proceedings of Int. Conf. on Artificial Neural Networks**, Brighton, UK, pp. 953-956, 1992.
- [162] PHAM, D. T., LIU, X., Dynamic System Modelling using Partially Recurrent Neural Networks. **Journal of Systems Engineering**, Vol. 2, pp. 90-97, 1992.
- [163] PHAM, D. T., LIU, X., Training of Elman Networks and Dynamic System Modelling. **International Journal of Systems Science**, Vol. 27, No. 2, pp. 221-226, 1996.
- [164] PINEDA, F. J., Generalization of Back-Propagation to Recurrent Neural Networks. **Physical Review Letters**, Vol. 59, No. 19, pp. 2229-2232, 1987.
- [165] POMERLEAU, D. A., GOWDY, J., THORPE, C. E., Combining Artificial Neural Networks and Symbolic Processing for Autonomous robot Guidance. **Engineering Applications of Artificial Intelligence**, Vol. 4, No. 4, pp. 279-285, 1991.
- [166] PSALTIS, D., SIDERIS, A., YAMAMURA, A. A., A Multilayered Neural Networks Controller. **IEEE Control Systems Magazine**, 8, pp:17-21, 1988.
- [167] RANGAN, S., WOLODKIN, G., POULLA, K., Identification Methods for Hammerstein Systems. **Proc. CDC**, pp. 697-702, New Orleans, 1995.
- [168] RAO, B. L. S., **Nonparametric Functional Estimation**. Academic Press, Orlando, FL, 1983.
- [169] RAOL, J. R., Parameter estimation of state space models by recurrent neural networks. **IEE Proc.-Control Theory Appl.**, Vol. 142, No. 2, pp. 114-118, 1995.
- [170] ROISENBERG, M., **Emergência da Inteligência em Agentes Autônomos Através de Modelos Inspirados na Natureza**. Florianópolis, 1998. Tese (Doutorado em Engenharia Elétrica, área de concentração Sistemas de Informação) – Centro Tecnológico, Universidade Federal de Santa Catarina.
- [171] ROSENBLATT, F., **The Perceptron : A Probabilistic Model for Information Storage**

- and Organization in the Brain. **Psychological Review**, 65: 386-408, 1958.
- [172] ROSENBROCK, H. H., **State Space and Multivariable Theory**. Nelson, London, 1970.
- [173] ROUTH, E. J., **A Treatise on the Stability of a Given State of Motion**. MacMillan, London, 1877.
- [174] RUDIN, W., **Principles of Mathematical Analysis**. 3rd Edition, McGraw-Hill, Auckland, 1976.
- [175] RUIZ-VARGAS, J. A., **Identificação de Sistemas Dinâmicos via Redes Neurais Artificiais**. São José dos Campos, 1997. Dissertação (Mestrado em Engenharia Eletrônica e Computação, Divisão de Engenharia Eletrônica) - Instituto Tecnológico de Aeronáutica, CTA.
- [176] RUMELHART, D. E., HINTON, G. E., WILLIAMS, R. J., Learning Representations by Back-Propagation Errors. **Nature**, Vol. 323-9, October, 1986(a).
- [177] RUMELHART, D. E., McCLELLAND, J. L., (Eds.), and The PDP Research Group, **Parallel Distributed Processing : Explorations in the Microstructure of Cognition, Vol. 1: Foundations**. Cambridge, MA: Bradford Books/The MIT Press, 1986(b).
- [178] RUMELHART, D. E., McCLELLAND, J. L., (Eds.), and The PDP Research Group, **Parallel Distributed Processing : Explorations in the Microstructure of Cognition, Vol. 2: Psychological and Biological Models**. Cambridge, MA: Bradford Books/The MIT Press, 1986(c).
- [179] SAINT-DONAT J., BHAT N., McAVOY T. J., Neural Nets Based Model Predictive Control. **Int. Journal Control**, Vol. 54, No. 6, pp. 1453-1468. 1991.
- [180] SANDBERG, I. W., Expansions for Nonlinear Systems. **Bell Syst. Tech. J.**, Vol. 61, pp. 159-199, 1982.

- 
- [181] SAXÉN, H., SAXÉN, B., A Tool for Modeling, Simulation and Prediction Using Feedforward and Recurrent Neural Networks. In **Anais do I Simpósio Brasileiro de Redes Neurais**, pp. 55-60, Caxambu, MG, Brazil, 1994.
- [182] SBÁRBARO, D., JOHANSEN, T. A., Multiple local Leguerre models for modelling nonlinear dynamic systems of the Wiener class. **IEE Proc.-Control Theory Appl.**, Vol. 144, No. 5, pp. 375-380, 1997.
- [183] SCHETZEN, M., **The Volterra and Wiener Theories of Nonlinear Series**. Wiley, NY, 1980.
- [184] SIMPSON, P. K., **Artificial Neural Systems - Foundations, Paradigms, Applications, and Implementations**. New York, USA: Pergamon Press, 1990.
- [185] SIMPSON, R. J., Use of High Frequency Signals in Identification of Certain Nonlinear Systems. **Int. J. Syst. Sci.**, 4, pp. 121-127, 1973.
- [186] SINGH, M., ELLOY, J., MEZENCEV, R., MUNRO, N., **Applied Industrial Control**. Pergamon Press, Oxford, UK, 1980.
- [187] SJÖBERG, J., **Non-Linear System Identification with Neural Networks**. Ph.D. Thesis, Department of Electrical Engineering, Linköping University, Sweden, 1995.
- [188] SJÖBERG, J., LJUNG, L., Overtraining, Regularization, and Searching for Minimum in Neural Networks. In **Preprint 4th IFAC Symposium on Adaptive Systems in Control and Signal Processing**, pp. 669-674, Grenoble, France, 1992.
- [189] SÖDERSTRÖM, T., STOICA, P., **System Identification**. Prentice-Hall International, Hemel Hempstead, Hertfordshire, 1989.
- [190] SONG, Q., Robust Training Algorithm of Multilayered Neural Networks for Identification of Nonlinear Dynamic Systems. **IEE Proc.-Control Theory Appl.**, Vol. 145, No. 1, pp. 41-46, 1998.

- 
- [191] SONTAG, E. D., **Some Topics in Neural Networks and Control**. Report Number LS93-02, Department of Mathematics, Rutgers University, New Brunswick, USA, 1993.
- [192] STOICA, P., SÖDERSTRÖM, T., **Instrumental-Variable Methods for Identification of Hammerstein Systems**. *Int. J. Control*, 35, pp. 459-476, 1982.
- [193] SUYKENS, J. A. K., DE MOOR, B. L. R., VANDERWALLE, J., **Nonlinear System Identification using Neural State Space Models, Applicable to Robust Control Design**. *Int. J. Control*, Vol. 62, No. 1, pp. 129-152, 1995.
- [194] SWACHUK, A. A., STRAND, T. C., **Applications of Optical Fourier Transforms**. Edited by H. Stark, Academic Press, NY, 1982.
- [195] TAKAGI, T., SUGENO, M., **Fuzzy Identification of Systems and its Applications to Modeling and Control**. *IEEE Transactions on Systems, Man, and Cybernetics*, 15, pp. 116-132, 1985.
- [196] TAKAHASHI, Y., RABINS, M. J., AUSLANDER, D. M., **Control and Dynamic Systems**. Addison-Wesley Publishing Company, Second Printing, 1972.
- [197] THATHACHAR, M. A. L., RAMASWAMY, S., **Identification of a Class of Nonlinear Systems**. *Int. J. Control*, 18, pp. 741-752, 1973.
- [198] TSUNG, F. S., **Modeling Dynamical Systems With Recurrent Networks**. PhD Thesis. University of California at San Diego, San Diego, CA, 1994.
- [199] TUNAY, I, KAYNAK, O., **A New Variable Structure Controller for Affine Nonlinear Systems with Non-Matching Uncertainties**. *Int. J. Control*, Vol. 62, No. 4, pp. 917-939, 1995.
- [200] TURCHETTI, C., CONTI, M., CRIPPA, P., ORCIONI, S., **On the Approximation of Stochastic Processes by Approximate Identity Neural Networks**. *IEEE Transactions on Neural Networks*, Vol. 9, No. 6, pp. 1069-1085, 1998.

- 
- [201] TURNER, P., MORRIS, J., MONTAGUE, G., Chapter 7 – Applications of Dynamic Artificial Neural Networks in State Estimation and Nonlinear Process Control. **Neural Network Applications in Control**, Ed. By G. W. Irwin, K. Warwick e K. J. Hunt, IEE Control Engineering Series 53, 1995.
- [202] TZIRKEL-HANCOCK, E., **Stable Control of Nonlinear Systems Using Neural Networks**. Ph.D. Thesis. Trinity College, Cambridge University, Cambridge, England, 1992.
- [203] VAN DEN HOF, P., HEUBERGER, P., BOKOR, J., Identification with Generalized Orthonormal Basis Functions - Statistical Analysis and Error Bounds. In **Preprint 10th IFAC Symposium on System Identification**, Blanke, M. and Söderström, T., Editors, Volume 3, pp. 3207-3212, 1994.
- [204] VON ZUBEN, F. J., **Modelos Paramétricos e Não-Paramétricos de Redes Neurais Artificiais e Aplicações**. Campinas, 1996. Tese (Doutorado em Engenharia Elétrica, Departamento de Computação e Automação) - Faculdade de Engenharia Elétrica, Unicamp.
- [205] WAHLBERG, B., System Identification using Laguerre Models. **IEEE Trans. on Automatic Control**, 36(5), pp:551-562, 1991.
- [206] WAHLBERG, B., System Identification using Kautz Models. **IEEE Trans. on Automatic Control**, 39(6), pp:1276-1282, 1994.
- [207] WARWICK, K., Chapter 3 - Intelligent Adaptive Control. **Intelligent Control Systems - Theory and Applications**. Ed. by M. M. Gupta e N. K. Sinha, IEEE Press, 1996.
- [208] WERBOS, P. J., **Beyond Regression : New Tools for Prediction and Analysis in The Behavioral Sciences**. PhD Thesis. Harvard University, Cambridge, MA. 1974.
- [209] WERBOS, P. J., Neural Networks for Control and System Identification. In

- 
- Proceedings of the 28th Conference on Decision and Control**, pp. 260-265, Tampa, FL, 1989.
- [210] WERBOS, P. J., Chapter 13 - Neurocontrol and Elastic Fuzzy Logic - Capabilities, Concepts, and Applications. **Intelligent Control Systems - Theory and Applications**. Ed. by M. M. Gupta e N. K. Sinha, IEEE Press, 1996.
- [211] WHITE, D. A., JORDAN, M. I., Optimal Control : A Foundation for Intelligent Control - Chaper 6. **Handbook of Intelligent Control - Neural, Fuzzy and Adaptive Approaches**. White, D. A. and Sofge, D. A., (Eds.), New York : Van Nostrand Reinhold, 1992.
- [212] WHITE, D. A., SOFGE, D. A., (Eds.), **Handbook of Intelligent Control - Neural, Fuzzy and Adaptive Approaches**. New York : Van Nostrand Reinhold, 1992.
- [213] WIDROW, B., HOFF, M. E., Adaptive Switching Circuits. **Proc. of the 1960 WESCON Convention Record**, pp. 96-104, 1960.
- [214] WIDROW, B., STEARNS, S., **Adaptive Signal Processing**. Englewood Cliffs, USA : Prentice-Hall, 1985.
- [215] WIENER, N., **The Extrapolation, Interpolation and Smoothing of Stationary Time Series**. John Wiley, New York, 1949.
- [216] WIGREN, T., Convergence Analysis of Recursive Identification Algorithms Based on the Nonlinear Wiener Model. **IEEE Transactions on Automatic Control**, Vol. 39, No. 11, pp. 2191-2206, 1994.
- [217] WILLIS, M. J., MASSIMO, C. D., MONTAGUE, G. A., THAM, M. T., MORRIS, A. J., On Artificial Neural Networks in Process Engineering. **IEE Proceedings - Part D**, 138, pp:256-266. 1991.
- [218] WONHAM, W. M., **Linear Multivariable Control : A Geometric Approach**. Springer Verlag, New York, Second Edition, 1979.

- 
- [219] YU, S.-H., ANNASWAMY, A. M., Adaptive Control of Nonlinear Dynamic Systems Using  $\theta$ -Adaptive Neural Networks. **Automatica**, Vol. 33, No. 11, pp. 1975-1995, 1997.
- [220] ZAMES, G., Feedback and Optimal Sensitivity : Model Reference Transformations, Multiplicative Seminorms and Approximate Inverses. **IEEE Transactions on Automatic Control**, Vol. 26, pp. 301-320, 1981.
- [221] ZBIKOWSKI, R., **Recurrent Neural Networks : Some Control Problems**. Ph.D. Thesis, Department of Mechanical Engineering, Glasgow University, Glasgow, Scotland, U.K. (Disponível no servidor FTP [ftp.mech.gla.ac.uk](ftp://ftp.mech.gla.ac.uk) como /rafal/zbikowski-phd.ps), 1994.
- [222] ZBIKOWSKI, R., GAWTHROP, P. J., Chapter 3 - Fundamentals of Neurocontrol : a survey. **Neural Networks Application in Control**, Edited by G. W. Irwin, K. Warwick and K. J. Hunt, IEE Control Engineering Series 53, The Institution of Electrical Engineers, London, United Kingdom, 1995.
- [223] ZHANG, J., MORRIS, A. J., Fuzzy Neural Networks for Nonlinear Systems Modelling. **IEE Proc.-Control Theory Appl.**, Vol. 142, No. 6, pp. 551-561, 1995.
- [224] ZHANG, Y. K., BAI, E. W., Simulation of Spring Discharge from a Limestone Aquifer in Iowa. **Hydrogeol. J.**, 4. pp. 41-54, 1996.
- [225] ZIEGLER, J. G., NICHOLS, N. B., Optimum Settings for Automatic Controllers. **Transactions of the ASME**, Vol. 64, pp. 759-768, 1942.
- [226] ZURADA, J. M., **Introduction to Artificial Neural Systems**. New York, USA: West Publishing Company, 1992.