DERAL DE ENGENHARIA DE ITAJUBÁ

DISSERTAÇÃO DE MESTRADO

"REDES NEURAIS PARA CONTROLE EM TEMPO-REAL"

PAULO CÉSAR DO NASCIMENTO

ORIENTADOR : PROF. ALEXANDRE PINTO ALVES DA SILVA
CO-ORIENTADOR: PROF. GERMANO LAMBERT TORRES

ITAJUBÁ - MG
1995

À Rita, minha família e amigos.

AGRADECIMENTOS

Este trabalho só foi possível graças ao apoio e colaboração das seguintes pessoas:

Aos meus pais, agradeço pelo incentivo e a confiança que em mim depositaram.

Aos professores, Alexandre Pinto Alves da Silva e Germano Lambert Torres, agradeço pela orientação e dedicação.

Aos colegas do GIA, agradeço pela amizade e pelo apoio nos momentos mais difíceis.

À CAPES, agradeço pelo apoio financeiro.

Agradeço a todos os amigos da EFEI, que de uma forma ou outra ajudaram na realização deste trabalho.

RESUMO

O problema de controle adaptativo é ajustar dinamicamente os parâmetros do controlador, de tal maneira que a saída do sistema siga o sinal de referência. Esquemas de controle adaptativo podem ajustar o controlador conforme as características do sistema, proporcionando um alto nível de desempenho.

No entanto, o controle adaptativo convencional tem algumas desvantagens. Em aplicações práticas, é difícil expressar um sistema dinâmico real através de equações matemáticas. Os algoritmos de controle adaptativo existentes funcionam para problemas específicos, mas, não funcionam uniformemente bem para um número significativamente grande de problemas. Toda aplicação deve ser analisada individualmente, isto é, um problema específico é resolvido usando-se um algoritmo específico. Tal algoritmo envolve algum conhecimento sobre o sistema, a fim de se ter uma escolha adequada do modelo de identificação, e um estudo de compatibilidade entre este modelo e um algoritmo adaptativo.

Nos últimos anos, as redes neurais artificiais (RNA's) têm atraído o interesse de pesquisadores de diferentes áreas, por possuírem características tais como, processamento paralelo intrínseco, capacidade de aprendizagem, mapeamento não-linear e capacidade de generalização. Controle de sistemas é uma das mais promissoras áreas de aplicação das redes neurais.

O perceptron multicamadas, treinado com o algoritmo retro-propagação de erro ("back-propagation"), tem sido a arquitetura de rede neural mais popular aplicada em controle. No entanto, este tipo de arquitetura de rede neural tem várias imperfeições, tais como, dificuldade no ajuste dos parâmetros de aprendizagem, convergência lenta, falhas no treinamento devido a mínimos locais e arquitetura pré-especificada (modelo paramétrico).

Apesar de muitas tentativas para melhorar o desempenho da regra delta generalizada com retro-propagação de erro, e encontrar uma alternativa, não há ainda um método eficiente e confiável para treinar uma rede perceptron multicamadas. O problema principal é a alta não-linearidade da função de erro.

Esta dissertação apresenta um modelo não-paramétrico de RNA para o controle adaptativo de sistemas não-lineares. A arquitetura de RNA proposta combina o conceito de funções de base ortonormais com a idéia de redes polinomiais. Uma combinação de funções ortogonais pode ser usada para produzir um mapeamento desejado. No entanto, não há

outra maneira além da tentativa e erro para a escolha das funções ortogonais. As redes polinomiais também podem ser usadas para aproximação de funções, porém, não é fácil estabelecer a ordem da função de ativação.

A combinação dos dois conceitos produz um modelo poderoso de RNA, devido à capacidade de seleção automática na entrada das redes polinomiais. As vantagens da RNA proposta são:

- A arquitetura da RNA é automaticamente definida pelo processo de treinamento (modelo não paramétrico);
- Não há parâmetro de aprendizagem para ser ajustado;
- Sem problemas de mínimos locais;
- Convergência rápida.

A arquitetura de RNA proposta foi testada em um esquema de controle adaptativo direto para controle de velocidade de um motor CC. Os resultados foram comparados com os produzidos por um esquema de controle adaptativo indireto, baseado em perceptrons multicamadas, treinados pelo algoritmo retro-propagação de erro ("back-propagation").

ABSTRACT

The adaptive control problem is to dynamically adjust the parameters of the controller such that the output of the plant follows the reference signal. Adaptive control schemes can adjust the controller according to process characteristics, providing a higher level of performance.

However, traditional adaptive control has several drawbacks. In practical applications, it is difficult to express the real plant dynamics in mathematical equations. Existing adaptive control algorithms work for specific problems, but they do not work uniformly well for a wide range of problems. Every application must be analyzed individually, that is, a specific problem is solved using a specific algorithm. Such an algorithm involves some knowledge about the process for a proper choice of the identification model, and a study of compatibility between this model and an adaptive algorithm.

In recent years, artificial neural networks (ANN's) have attracted the interest of researchers in different areas due to features such as intrinsic parallel processing, learning capacity, nonlinear mapping and generalization capability. System control is one of the most promising areas of neural network application.

The multilayer perceptron trained with error back propagation has been the most popular ANN applied to control. However, it has several shortcomings such as the difficult setting of learning parameters, slow convergence, training failures due to local minima, pre-specified architecture (parametric model).

Despite many attempts to improve the performance of the generalized delta rule with backward error propagation, and to find an alternative to it, there is still no efficient and reliable method to train a multilayer perceptron. The main problem is the high non linearity of the error function.

This dissertation presents a nonparametric ANN model for adaptive control of nonlinear systems. The proposed ANN mixes the concept of orthonormal basis functions with the idea of polynomial networks. A combination of orthogonal functions can be used to produce a desired mapping. However, there is no way besides trial and error to choose which orthogonal functions should be selected. Polynomial nets can be used for function approximation, too. However, it is not easy to set the order of the activation function.

The combination of the two concepts produces a very powerful ANN model due to the automatic input selection capability of the polynomial networks. The advantages of the proposed ANN are:

- The ANN architecture is automatically difined by the training process (nonparametric model);
- There is no learning parameter to be set;
- No local minima problems;
- Fast convergence.

The proposed ANN has been tested for speed control of a DC motor. The results have been compared with the ones provided by an indirect adaptive control scheme based on multilayer perceptrons trained by back-propagation.

SUMÁRIO

CAPÍTULO 5 - RESULTADOS DAS SIMULAÇÕES

CAPÍTULO 6 - CONCLUSÕES

CAPÍTULO 1

SISTEMAS DE CONTROLE

1.1 - INTRODUÇÃO

Este capítulo mostra algumas estruturas de controle, baseadas na teoria de controle clássico e moderno. Controle adaptativo é a técnica que mais tem se desenvolvido na teoria de controle; no entanto, o seu uso está limitado a sistemas dinâmicos lineares variantes no tempo, ou invariantes no tempo com parâmetros parcialmente desconhecidos e a sistemas dinâmicos não-lineares que aceitam hipóteses simplificadoras, como a linearização.

Entre as indústrias que utilizam massivamente controladores adaptativos, pode-se citar por exemplo, as indústrias químicas, de papel e celulose, aeronáutica e bélica. Porém, os controladores adaptativos convencionais geralmente são de uso específico e dependem de um bom conhecimento sobre o sistema a ser controlado.

Atualmente, pesquisadores procuram desenvolver sistemas de controle inteligentes, capazes de lidar diretamente com sistemas não-lineares variantes no tempo, com parâmetros desconhecidos e que tenham um alto grau de autonomia e segurança.

1.2 - SISTEMA DE CONTROLE DE MALHA ABERTA

A figura 1.1 mostra uma estrutura de controle de malha aberta, esta é a maneira mais simples de se controlar um sistema, no entanto, é a menos eficiente.

O controlador recebe em sua entrada a referência r, o mesmo gera em sua saída o sinal de controle u, necessário para o acionamento do sistema P. A saída y do sistema deve acompanhar a referência r dada pelo operador. Se ocorrer uma variação na variável externa T, ocorrerá também uma variação na saída y, porém, o controlador não toma conhecimento

de tal variação e portanto, não é capaz de modificar o sinal u a fim de que o erro $r - y$ seja minimizado.

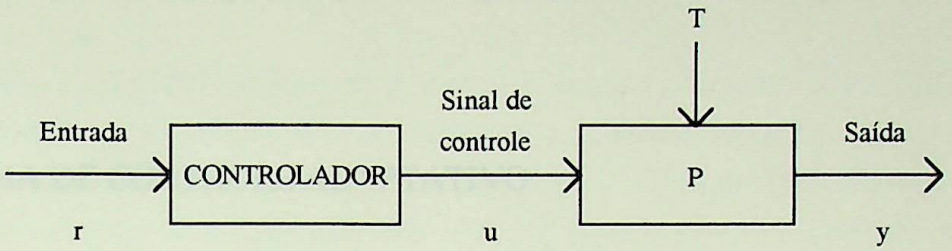


Figura 1.1 - Sistema de controle de malha aberta.

1.3 - SISTEMA DE CONTROLE COM REALIMENTAÇÃO

A figura 1.2 mostra uma estrutura de controle com realimentação, também chamada de sistema de controle de malha fechada. Pode-se observar que o sinal de saída y, do sistema P a ser controlado, é realimentado e comparado com um valor de referência r, gerando um sinal de erro $e = r - y$, que alimentará o controlador. Se ocorrer uma variação inesperada na saída do sistema P, a mesma é sentida pela realimentação e conseqüentemente, o valor de erro aumentará positivamente ou negativamente. O controlador então corrigirá o sinal de controle u de acordo com o valor do erro, o que deverá ajustar novamente o valor de y no valor desejado, definido pela referência r dada pelo operador.

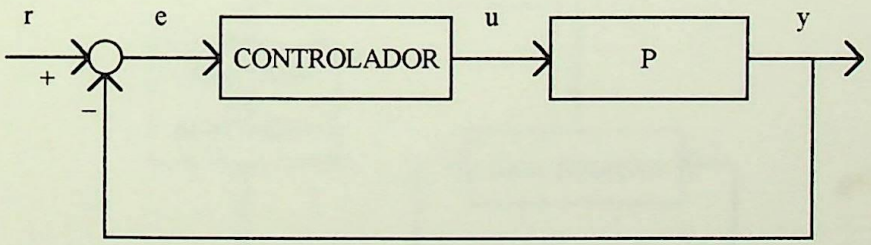


Figura 1.2 - Sistema de controle com realimentação

Pode-se concluir que o sistema de controle com realimentação é capaz de oferecer uma precisão maior na saída do sistema P, uma resposta dinâmica melhor e redução dos efeitos causados por variações inesperadas na saída. No entanto, a realimentação pode trazer problemas de sensibilidade a ruídos, comprometendo assim a estabilidade.

O controlador utilizado no sistema de controle com realimentação é o popular PID; constituído por três parcelas, a proporcional, a integral e a derivativa. Cada parcela possui um ganho correspondente, e para o ajuste dos ganhos são utilizadas regras heurísticas.

1.4 - SISTEMA DE CONTROLE ADAPTATIVO

Com o surgimento de sistemas cada vez mais complexos, o sistema de controle com realimentação mostrou-se ineficaz, como por exemplo, controle da operação de piloto automático de aeronaves. O sistema de controle com realimentação não é capaz de ajustar um novo ponto de operação quando ocorrem variações inesperadas nos parâmetros do sistema controlado (o piloto automático, bem ajustado em um determinado ponto de operação, tinha sua eficiência deteriorada quando este ponto era alterado devido a variações de altitude e velocidade da aeronave). Para resolver tal problema, foi desenvolvido o controle adaptativo, baseado em métodos de identificação, otimização e representação de sistemas por variáveis de estado [11].

A implementação deste tipo de controle só foi possível graças ao desenvolvimento da eletrônica digital, que por sua vez, possibilitou o surgimento dos microprocessadores, necessários à execução dos algoritmos de ajuste automático dos controladores.

O uso de algoritmos convencionais de controle adaptativo é justificado quando; o sistema controlado é invariante no tempo mas seus parâmetros não são perfeitamente conhecidos, ou, o sistema controlado é variante no tempo.

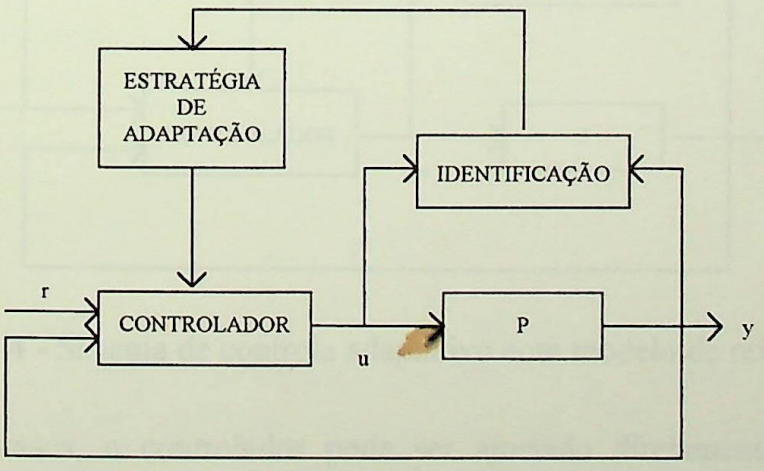


Figura 1.3 - Sistema de controle adaptativo.

A figura 1.3 mostra a estrutura básica de um sistema de controle adaptativo. Pode-se observar que além da malha de realimentação usual, existe outra malha, que é encarregada do ajuste do controlador.

A malha de ajuste geralmente é dividida em dois blocos: o de identificação do sistema controlado e o de adaptação. A função do primeiro bloco é criar um modelo através da identificação dos parâmetros do sistema P, e a função do segundo é ajustar os parâmetros do controlador através de regras pré-definidas, de acordo com os parâmetros do modelo identificado. O objetivo do sistema de controle adaptativo é manter um determinado comportamento da saída y, mesmo que venham a ocorrer variações no sistema P. Cabe agora ressaltar que, para se garantir um bom funcionamento, o projeto do sistema de controle deve estar associado a um certo conhecimento da estrutura e do comportamento do sistema a ser controlado.

A figura 1.4 mostra a estrutura do controle adaptativo indireto com modelo de referência. A função do modelo de referência é especificar as características dinâmicas desejadas na saída y. O sinal de erro de controle é derivado da diferença entre a saída do modelo de referência e a saída y. A malha de realimentação usual e a malha de ajuste devem operar simultaneamente.

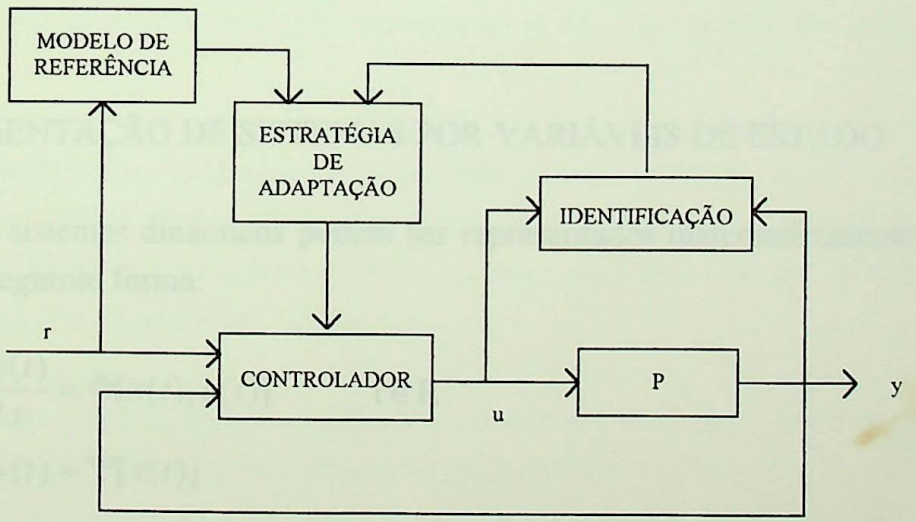


Figura 1.4 - Sistema de controle adaptativo com modelo de referência.

Em alguns casos, o controlador pode ser ajustado diretamente, ou seja, sem a utilização do bloco de identificação, tem-se assim, o controle adaptativo direto com modelo de referência.

No bloco de identificação, o algoritmo mais utilizado é o dos mínimos quadrados recursivo. Para o controlador, geralmente os reguladores PID convencionais têm sido utilizados [10].

1.5 - SISTEMA DE CONTROLE INTELIGENTE

A teoria de controle clássico e moderno gerou técnicas capazes de controlar sistemas lineares invariantes no tempo com parâmetros parcialmente desconhecidos e sistemas não-lineares que aceitam hipóteses simplificadoras, geralmente ligadas à linearização [11]. Porém, estas técnicas mostram uma limitação muito grande quando o assunto é controle de sistemas complexos, variantes no tempo, através de controladores com alto grau de autonomia [12, 18].

Pesquisas estão sendo feitas para criar novas técnicas baseadas em redes neurais artificiais e inteligência artificial, entre outras, a fim de possibilitar a criação de sistemas de controle inteligente com alto grau de autonomia [19]. Estes sistemas, além das características de controle adaptativo, deverão ser capazes de fazer o sensoriamento do ambiente, detectar falhas e tomar decisões.

1.6 - REPRESENTAÇÃO DE SISTEMAS POR VARIÁVEIS DE ESTADO

Muitos sistemas dinâmicos podem ser representados matematicamente por equações de estado da seguinte forma:

$$\begin{aligned} \frac{d\,x(t)}{d\,t} &= \Phi[x(t), u(t)] \qquad t \in R^{+} \\ y(t) &= \Psi[x(t)] \end{aligned} \tag{1.1}$$

onde $x(t) = [x_1(t), x_2(t), ..., x_n(t)]^T$, $u(t) = [u_1(t), u_2(t), ..., u_p(t)]^T$ e $y(t) = [y_1(t), y_2(t), ..., y_m(t)]^T$ representam os n estados, p entradas e m saídas de um sistema de ordem n. $\Phi : R^n \times R^p \rightarrow R^n$ e $\Psi : R^n \rightarrow R^m$ são funções possivelmente não-lineares representando respectivamente, a dinâmica do sistema e a relação do estado com a saída. O estado do sistema no tempo t, representado pelo vetor $x(t)$, é completamente determinado pelo estado no tempo $t_0 < t$, ou seja, $x(t_0)$, e pela entrada $u(t)$

durante o intervalo $[t_0, t)$. A saída $y(t)$ é completamente determinada pelo estado do sistema no tempo t.

Neste trabalho, os sistemas serão representados através de equações discretas no tempo, portanto, a equação 1.1 fica da seguinte forma:

$$\begin{aligned} x(K+1) &= \Phi[x(K), u(K)] \\ y(K) &= \Psi[x(K)] \end{aligned} \tag{1.2}$$

onde $u(.)$, $x(.)$ e $y(.)$ são séries temporais.

Se o sistema descrito pela equação 1.2 for linear e invariante no tempo, tem-se a seguinte equação:

$$\begin{aligned} x(K+1) &= A\, x(K) + B\, u(K) \\ y(K) &= C\, x(K) \end{aligned} \tag{1.3}$$

onde A, B e C são matrizes de dimensões $(n \times n)$, $(n \times p)$ e $(m \times n)$ respectivamente. Quando A, B e C são conhecidas, os conceitos de controlabilidade, observabilidade e estabilidade de sistemas lineares são aplicáveis, e os métodos para determinar os sinais de controle $u(.)$, que otimizem um determinado critério de desempenho, são bem conhecidos.

Se o sistema descrito pelas equações 1.1 e 1.2 é não-linear, tem-se equações diferenciais algébricas não-lineares e a solução é obtida, em geral, através de hipóteses simplificadoras. Estas hipóteses normalmente dizem respeito à linearização de parte do sistema, o que limita o comportamento do mesmo. Portanto, há um grande interesse no tratamento direto de sistemas não-lineares.

Um sistema linear dinâmico discreto, invariante no tempo, de uma entrada e uma saída, pode ser representado pela seguinte equação a diferenças:

$$y(K+1) = \sum_{i=0}^{b-1} \alpha_i\, y(K-i) \quad + \quad \sum_{j=0}^{a-1} \beta_j\, u(K-j) \tag{1.4}$$

conhecida como modelo de média móvel auto-regressivo determinístico, onde α_i e β_j são parâmetros constantes a determinar. Os termos atrasados de y formam o componente auto-regressivo e os termos atrasados de u formam o componente de média móvel [2, 11].

1.7 - OBJETIVOS E CONTRIBUIÇÕES DA DISSERTAÇÃO

Em aplicações práticas, é difícil expressar um sistema dinâmico real através de equações matemáticas. Nos últimos anos, as redes neurais artificiais têm atraído o interesse de pesquisadores da área de controle por possuírem características tais como, processamento paralelo intrínseco, capacidade de aprendizagem, mapeamento não-linear e capacidade de generalização. Estas características possibilitam o tratamento direto de sistemas não-lineares.

O objetivo principal desta dissertação é apresentar um modelo não-paramétrico de rede neural artificial para o controle adaptativo de sistemas, contribuindo assim para o aprimoramento da aplicação das redes em controle [20, 21].

1.8 - PRÓXIMOS CAPÍTULOS

No segundo capítulo, será dada uma base teórica sobre redes neurais artificiais, contendo um histórico seguido de algumas arquiteturas e tipos de redes. A arquitetura de rede do tipo perceptron multicamadas, bem como o respectivo algoritmo de treinamento, retro-propagação de erro ("back-propagation"), serão explicados detalhadamente.

O terceiro capítulo apresenta o modelo não-paramétrico de rede neural proposto, baseado nas redes polinomial e funcional.

O assunto tratado no quarto capítulo é o de redes neurais aplicadas em sistemas de controle adaptativo. Será mostrado como a arquitetura de rede do tipo perceptron multicamadas pode ser usada na modelagem para frente ("forward modelling") como um identificador do sistema, e na modelagem inversa ("inverse modelling") como controlador do sistema formando assim, um sistema de controle adaptativo indireto. Também será mostrado como a arquitetura de rede proposta pode ser usada como controlador do sistema, formando um sistema de controle adaptativo direto.

No quinto capítulo, os dois esquemas de controle adaptativo, direto e indireto, serão simulados em computador. O sistema controlado será um conjunto formado por um amostrador ("zero-order hold"), uma ponte retificadora e um motor de corrente contínua. O objetivo das simulações é mostrar a superioridade da rede proposta.

Finalmente, serão apresentadas as conclusões sobre o desempenho dos dois métodos de controle adaptativo.

CAPÍTULO 2

CONCEITOS SOBRE REDES NEURAIS

2.1 - HISTÓRICO

O estudo de redes neurais teve início nos anos 40, com a realização das primeiras pesquisas sobre a natureza da atividade neural do ser humano, tendo como objetivo, desenvolver os princípios básicos para o processamento inteligente da informação. Na década de 40 surge o princípio da cibernética, proposta por Wiener [23], que é a relação entre princípios de engenharia, realimentação e função cerebral. Com o progresso das pesquisas sobre os computadores e o cérebro humano, tinha-se como objetivo, criar máquinas que tivessem a capacidade de aprender. Foi então proposto por Hebb [24] um modelo de aprendizagem.

No início da década de 60, Rosenblatt definiu um sistema de aprendizagem, capaz de aprender a classificar conjuntos de padrões através da modificação das conexões internas de uma rede neural [25]. O sistema recebeu o nome de perceptron. Porém, em 1969, Minsky e Papert provaram matematicamente que o perceptron não poderia ser usado para funções lógicas complexas [26]. Este fato contribuiu para o desencorajamento das pesquisas sobre o aprendizado de máquinas, mas, deve-se lembrar que Minsky e Papert provaram limitações de redes lineares em tarefas de reconhecimento de padrões; o mesmo não pode ser dito com relação às redes do tipo multicamadas.

Na década de 70, Werbos desenvolveu o algoritmo de treinamento chamado retro-propagação ("back-propagation"), porém, seu trabalho permaneceu quase desconhecido entre a comunidade científica [27]. Seu trabalho foi redescoberto por Parker [28] e Rumelhart [29] nos meados da década de 80.

No início dos anos 80, Hopfield divulgou um novo tipo de arquitetura de rede neural. O modelo é formado por um conjunto de equações diferenciais não-lineares de primeira ordem que minimiza uma certa função de energia. Este conjunto de equações representa o comportamento da conhecida rede de Hopfield [30].

Recentemente, as redes neurais têm sido aplicadas em diversos campos, como reconhecimento de padrões e controle. Em controle, as redes são usadas para tratar sistemas não-lineares graças à sua capacidade de processamento paralelo, aproximação funcional, auto-organização, aprendizagem e adaptação.

Para o futuro, a tendência é a implementação de sistemas de controle inteligentes, onde as redes neurais artificiais têm grande potencial de uso. Sistemas de controle inteligentes serão sistemas híbridos, contendo técnicas de redes neurais, sistemas especialistas, conjuntos difusos e algoritmos genéticos, entre outras.

2.2 - ARQUITETURAS DE REDES NEURAIS ARTIFICIAIS

A arquitetura da rede é definida pelos elementos básicos de processamento, equivalentes aos neurônios do cérebro humano, e a maneira como estes elementos são conectados.

2.2.1 - Modelo básico de um neurônio

O elemento básico de processamento, ou neurônio, é representado por um modelo matemático. O elemento mostrado na figura 2.1 possui n entradas X e uma saída Y.

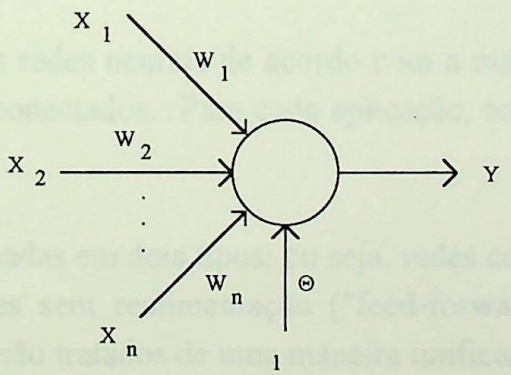


Figura 2.1 - Modelo de neurônio.

A saída do neurônio é dada pela seguinte equação:

$$y(t) = f\left(\sum_{i=1}^{n} w_i x_{i(t)} - \theta\right) \tag{2.1}$$

onde w_i é o peso de cada conexão, θ é um valor limite interno conhecido como "threshold" ou "offset", t é o tempo e f é uma função normalmente não-linear chamada de função de ativação.

Durante o processo de aprendizagem da rede neural, os pesos w_i são ajustados para determinar uma relação entrada-saída desejada e esta relação determina o comportamento da rede.

A função de ativação $f(x)$, dependendo da aplicação, pode ser uma função não-linear, contínua, diferenciável e limitada, como a seguinte função sigmoidal

$$f(x) = \frac{1}{1+e^{-x}} \qquad (0 < f(x) < 1) \tag{2.2}$$

ou como a função tangente hiperbólica

$$f(x) = \frac{1-e^{-x}}{1+e^{-x}} \qquad (-1 < f(x) < 1) \tag{2.3}$$

A referência [3] mostra uma modelagem mais detalhada do elemento básico de processamento.

2.2.2 - Tipos de redes

Pode-se classificar as redes neurais de acordo com a maneira com que os elementos básicos, ou neurônios, são conectados. Para cada aplicação, tem-se uma maneira adequada de se conectar os mesmos.

As redes são classificadas em dois tipos; ou seja, redes com realimentação ("recurrent nets") ou dinâmicas e redes sem realimentação ("feed-forward nets") ou estáticas. Na referência [2], os dois tipos são tratados de uma maneira unificada.

Nesta dissertação serão utilizados dois tipos de arquiteturas de redes sem realimentação. Uma delas será a rede perceptron multicamadas, que utiliza exatamente o modelo de neurônio mostrado no item 2.2.1 e possui uma arquitetura fixa. A outra será a rede polinomial, que ao contrário da primeira, possui uma arquitetura variável e utiliza um modelo de neurônio diferente, baseado em um polinômio de segundo grau.

2.2.2.1 - Redes com realimentação

Neste tipo de arquitetura de rede neural, múltiplos neurônios são interconectados para formar a organização da rede e a introdução da realimentação resulta em uma rede dinâmica, com vários pontos estáveis ou pontos de equilíbrio.

A rede de Hopfield é um exemplo, onde todos os neurônios são interconectados. Pode ser utilizada na implementação de memória associativa e na solução de problemas de otimização. A figura 2.2 mostra a organização da rede, onde os valores X são as entradas aplicadas no tempo zero, ou seja, os valores iniciais de cada neurônio, e os valores X' são as saídas obtidas após a convergência.

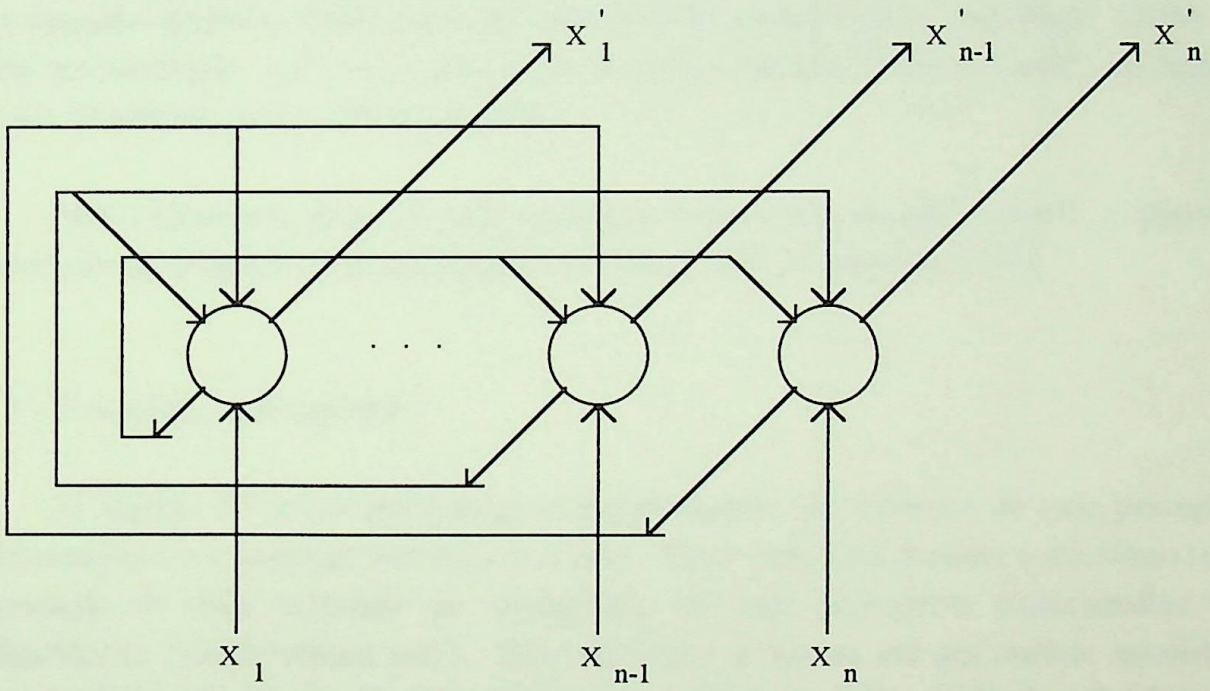


Figura 2.2 - Rede de Hopfield.

2.2.2.2 - Redes sem realimentação

A figura 2.3 mostra um exemplo de arquitetura de rede neural sem realimentação, a mesma é chamada de rede perceptron multicamadas. A estrutura dos neurônios é feita em camadas e os neurônios de cada camada estão interligados com os neurônios das camadas adjacentes, porém, os neurônios da mesma camada não se comunicam.

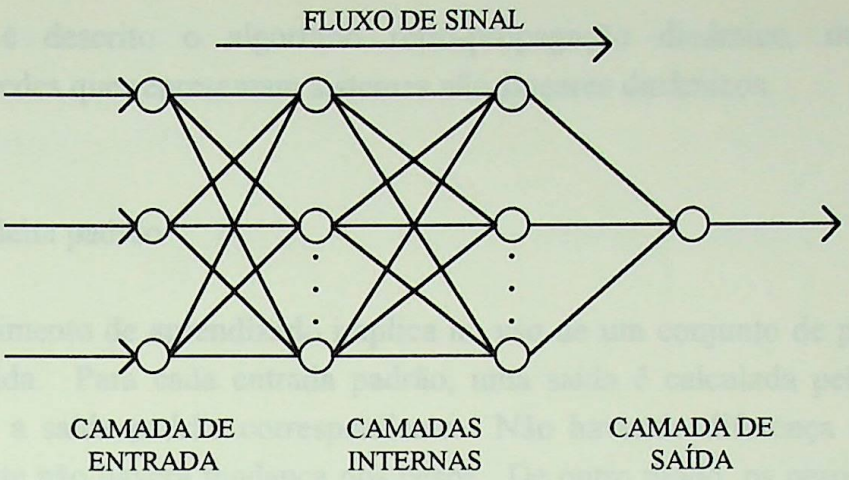


Figura 2.3 - Rede perceptron multicamadas.

A rede mostrada na figura 2.3 possui uma camada de entrada, uma camada de saída e duas camadas internas, sendo que cada camada pode possuir vários neurônios. Como não existe realimentação, o fluxo do sinal segue um único sentido ("feed-forward"), ou seja, da camada de entrada para a camada de saída.

Para o ajuste dos pesos de cada conexão, pode-se utilizar, por exemplo, o algoritmo de treinamento chamado retro-propagação de erro ("back-propagation") [27].

2.2.3 - Perceptron multicamadas

As figuras 2.3 e 2.1 mostraram, respectivamente, um exemplo de rede perceptron multicamadas e o modelo de neurônio utilizado. Neste item, será descrito o algoritmo retro-propagação de erro, utilizado no treinamento da rede perceptron multicamadas sem realimentação ("feed-forward net"). Este algoritmo se baseia em um método estatístico, chamado de método de aproximação estocástica, proposto em 1951 por Herbert Robbins and Sutton Monro da Universidade da Carolina do Norte [31].

Existem algoritmos de aprendizado supervisionados e não-supervisionados. No aprendizado supervisionado, utilizam-se sinais padrões de referência externos; no aprendizado não-supervisionado, utilizam-se sinais externos. O algoritmo retro-propagação de erro é um exemplo de aprendizado supervisionado e é o mais popular.

Uma função de erro é definida como sendo metade do quadrado da diferença entre a saída desejada e a saída atual da rede. A função de erro deve ser minimizada e para isso é utilizada a regra delta generalizada, que é uma técnica de pesquisa de gradiente. Na

referência [2] é descrito o algoritmo retro-propagação dinâmico, utilizado para o treinamento de redes que representam sistemas não-lineares dinâmicos.

2.2.3.1 - Regra delta padrão

O procedimento de aprendizado implica no uso de um conjunto de pares de padrões de entrada e saída. Para cada entrada padrão, uma saída é calculada pela rede e esta é comparada com a saída padrão correspondente. Não havendo diferença entre as saídas, conseqüentemente não haverá mudança nos pesos. De outro modo, os pesos deverão sofrer mudanças a fim de minimizar a diferença.

A regra delta padrão é utilizada para redes perceptron que não possuem camadas internas. A equação que gera a mudança dos pesos para cada padrão p de entrada-saída é dada por

$$\Delta_p W_{ji} = \eta (t_{pj} - o_{pj}) i_{pi} = \eta \delta_{pj} i_{pi} \tag{2.4}$$

onde j e i representam os índices dos neurônios ligados através do peso W_{ji}; t_{pj} é a saída desejada para o neurônio de saída j; o_{pj} é a saída calculada, do mesmo neurônio, correspondente à entrada padrão; i_{pi} é o valor da entrada padrão; η é a taxa de aprendizagem e $\Delta_p W_{ji}$ é a variação do valor do peso W_{ji}, correspondente ao padrão p do conjunto de treinamento. A camada de entrada atua apenas como repetidora dos sinais de entrada, distribuindo-os às entradas dos neurônios da camada de saída.

A regra implementa a descida em gradiente da função erro médio quadrático para funções de ativação lineares. Pelo fato da arquitetura da rede não possuir camadas internas, tem-se apenas um ponto de mínimo; garantindo-se portanto, o ajuste ótimo dos pesos.

2.2.3.2 - Regra delta generalizada

Para mostrar a regra delta generalizada, primeiro define-se o erro correspondente a um padrão p de entrada-saída como sendo o erro médio quadrático parcial dado por

$$E_p = \frac{1}{2} \sum_j (t_{pj} - o_{pj})^2 \tag{2.5}$$

e o erro médio quadrático global dado por $E = \sum E_p$, que é calculado após a apresentação de todos os padrões utilizados no treinamento. A função a ser minimizada pelo algoritmo é o erro médio quadrático global.

Tomando-se como base a camada i, anterior à camada j, define-se

$$S_{pj} = \sum_i W_{ji} o_{pi} \tag{2.6}$$

A saída calculada o_{pj} usa a função sigmoidal, equação 2.2, e é dada por

$$o_{pj} = f_j(S_{pj}) \tag{2.7}$$

Para se obter a generalização correta da regra delta,

$$\Delta_p W_{ji} \propto -\frac{\partial E_p}{\partial W_{ji}} \tag{2.8}$$

A derivada da equação 2.8 é calculada da seguinte maneira,

$$\frac{\partial E_p}{\partial W_{ji}} = \frac{\partial E_p}{\partial S_{pj}} \frac{\partial S_{pj}}{\partial W_{ji}} \tag{2.9}$$

Da equação 2.6 tem-se que:

$$\frac{\partial S_{pj}}{\partial W_{ji}} = \frac{\partial}{\partial W_{ji}} \sum_k W_{jk} o_{pk} = o_{pi} \tag{2.10}$$

Definindo-se δ_{pj} como sendo:

$$\delta_{pj} = -\frac{\partial E_p}{\partial S_{pj}} \tag{2.11}$$

Substituindo 2.10 e 2.11 em 2.9 tem-se,

$$-\frac{\partial E_p}{\partial W_{ji}} = \delta_{pj} o_{pi} \tag{2.12}$$

Então, para a implementação do gradiente de descida em E, a variação do peso é dada por

$$\Delta_p W_{ji} = \eta \delta_{pj} o_{pi} \tag{2.13}$$

O objetivo é fazer a propagação reversa dos δ's através da rede, para isso, a equação 2.11 é calculada da seguinte maneira,

$$\delta_{pj} = -\frac{\partial E_p}{\partial S_{pj}} = -\frac{\partial E_p}{\partial o_{pj}} \frac{\partial o_{pj}}{\partial S_{pj}} \tag{2.14}$$

Da equação 2.7 tem-se que,

$$\frac{\partial o_{pj}}{\partial S_{pj}} = f_j'(S_{pj}) \tag{2.15}$$

Resta calcular o segundo fator, para isso existem dois casos. No primeiro assume-se que a unidade j é uma unidade de saída da rede, então, da definição de E_p tem-se que

$$\frac{\partial E_p}{\partial o_{pj}} = -(t_{pj} - o_{pj}) \tag{2.16}$$

Substituindo as duas últimas equações em 2.14, tem-se

$$\delta_{pj} = (t_{pj} - o_{pj}) f_j'(S_{pj}) \tag{2.17}$$

para qualquer unidade de saída u_j. Se u_j não é uma unidade de saída, tem-se então

$$\sum_k \frac{\partial E_p}{\partial S_{pk}} \frac{\partial S_{pk}}{\partial o_{pj}} = \sum_k \frac{\partial E_p}{\partial S_{pk}} \frac{\partial}{\partial o_{pj}} \sum_i W_{ki} o_{pi} = \sum_k \frac{\partial E_p}{\partial S_{pk}} W_{kj} = -\sum_k \delta_{pk} W_{kj} \tag{2.18}$$

onde k representa o somatório sob os neurônios da camada posterior à camada do neurônio j. Substituindo 2.18 e 2.15 em 2.14,

$$\delta_{pj} = f_j'(S_{pj}) \sum_k \delta_{pk} W_{kj} \tag{2.19}$$

Conclui-se que a regra delta generalizada fornece, através das equações 2.17, 2.19 e 2.13, o procedimento computacional necessário para o ajuste de todos os pesos da rede perceptron multicamadas. Pelo fato da rede possuir camadas internas, pode-se ter vários mínimos locais, comprometendo o ajuste ótimo dos pesos; pois corre-se o risco de encalhar em um mínimo local longe do mínimo global.

O algoritmo retro-propagação ("back-propagation") pode ser descrito através dos seguintes passos de execução:

1) Inicializar pesos e "offsets" com valores aleatórios pequenos (em torno de zero);

2) Propagar um padrão de entrada pela rede;

3) Comparar o sinal de saída obtido com o valor desejado;

4) Calcular e retro-propagar a medida de erro (começando pela camada de saída) através da rede; e

5) Minimizar o erro ajustando os pesos das conexões

$$W_{ji}(K+1) = W_{ji}(K) + \Delta_p W_{ji}(K) \tag{2.20}$$

onde $\Delta_p W_{ji} = \eta \delta_{pj} o_{pi}$, e o índice K refere-se a iteração corrente; nesta equação deve ser observado que, se a unidade j é uma unidade de saída, então a equação 2.17 deve ser usada no cálculo de δ_{pj}. De outro modo, se a unidade j é uma unidade interna, então, a equação 2.19 deverá ser usada. Os "offsets" podem ser encarados como pesos e ajustados de maneira similar.

Um termo de momento pode ser adicionado na equação 2.20 com o objetivo de aumentar a velocidade de convergência da aprendizagem. O termo é colocado da seguinte forma:

$$W_{ji}(K+1) = W_{ji}(K) + \Delta_p W_{ji}(K) + \alpha \, \Delta_p W_{ji}(K-1) \tag{2.21}$$

onde α é basicamente uma constante, que determina o efeito da variação anterior na variação atual.

6) Apresentar o próximo padrão de entrada-saída e retornar ao passo 2. Os padrões de entrada-saída são apresentados, de forma cíclica e contínua, até que a saída acompanhe os resultados esperados. A rede será capaz de generalizar a aprendizagem adquirida se o conjunto de padrões de entrada-saída for escolhido adequadamente. Deve-se salientar que o sucesso total da aprendizagem nem sempre é garantido, a convergência pode tomar tempo e corre-se o risco de encalhar em um mínimo local.

Na referência [8] é descrito o algoritmo retro-propagação através do tempo ("back-propagation through time"), utilizado na identificação e controle de sistemas dinâmicos.

Neste trabalho, o algoritmo de treinamento "back-propagation" foi implementado baseado nas referências [9, 29]. Na referência [9] foi proposto um algoritmo "back-propagation" adaptativo, utilizando valores variáveis para a taxa de aprendizagem e de momento. Entretanto, somente a adaptação dos valores da taxa de momento foi implementada com sucesso e portanto, para a taxa de aprendizagem foi utilizado um valor fixo. Mesmo assim, atingiu-se uma melhora significativa no treinamento em comparação ao algoritmo convencional contido em [29].

2.2.3.3 - Algoritmo Estimativa Ótima de Treinamento (OET - "Optimal Estimating Training")

OET é uma técnica de aprendizado para perceptrons multicamadas. O conjunto de padrões de entrada e saída são representados na forma de matrizes. Um mapeamento não-linear que associa padrões de treinamento de entrada com padrões de treinamento de saída é encontrado resolvendo sucessivamente um sistema linear usando métodos diretos, que é de fato um procedimento baseado no método mínimos quadrados não-linear (aproximação local da função erro por uma função quadrática, e a exata aproximação da mesma por uma função aproximada).

A idéia geral do OET pode ser resumida usando a figura 2.4 [15, 22]. Nesta figura $h_{in,j}$ $(j = 0, 1, \ldots, m)$ é o conjunto das variáveis de entrada ($h_{in,0}$ representa uma entrada adicional com valor constante igual a 1). A propagação das entradas é feita pela soma das entradas ponderadas, isto é,

$$Z_{out,i} = \sum_{j=0}^{m} h_{in,j} w_{ji} \tag{2.22}$$

o valor de $h_{out,i}$ é calculado pela aplicação da função de ativação $h_{out,i} = f(Z_{out,i})$.

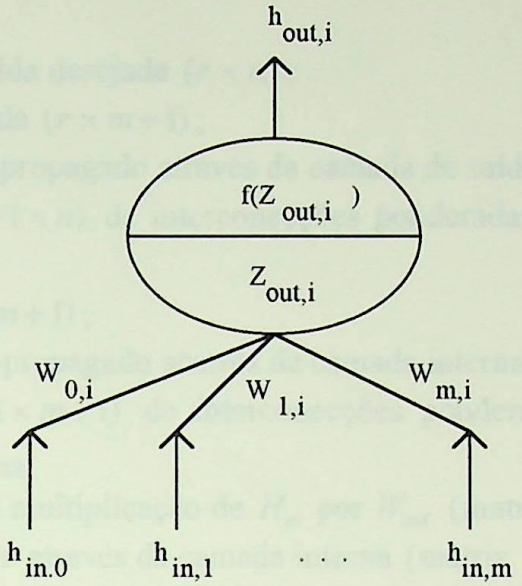


Figura 2.4 - Representação de um neurônio.

Para executar o algoritmo OET, um valor desejado (especificado) de saída $h_{d-out,i}$ é retro-propagado através da função de ativação e o resultado correspondente imediato desejado , $Z_{d-out,i}$, será obtido. Para executar esta operação, é necessário uma função contínua e inversível tal como a tangente hiperbólica. Esta função é adequada para OET, desde que as saídas desejadas possam ser positivas ou negativas. Assim, o conjunto das entradas e resultados intermediários podem ser interrelacionados para calcular a estimativa ótima, para um conjunto de interconecções ponderadas, no sentido dos mínimos quadrados.

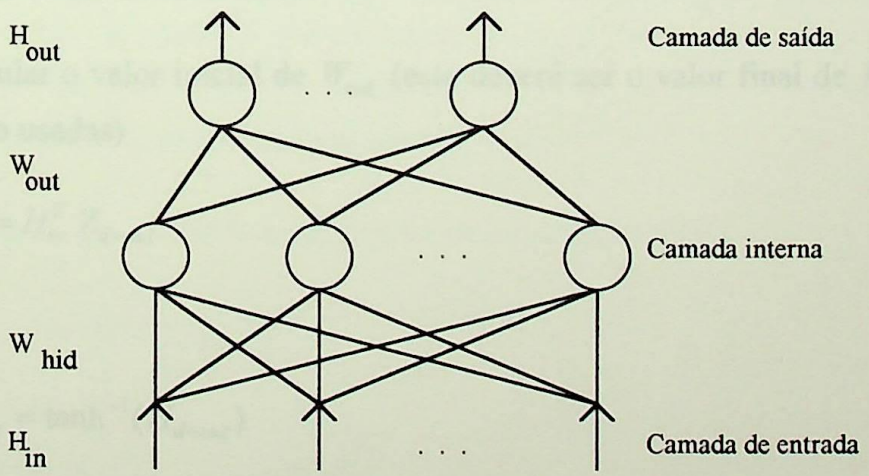


Figura 2.5 - Rede multicamadas.

A lista de símbolos utilizados para explicar o algoritmo será agora descrita (de acordo com a figura 2.5):

H_{d-out} : matriz de saída desejada $(r \times n)$;
H_{in} : matriz de entrada $(r \times m+1)$;
Z_{d-out} : H_{d-out} retro-propagado através da camada de saída (matriz $r \times n$);
W_{d-out} : matriz $(m+1 \times n)$ de interconecções ponderadas ligando a camada interna com a camada de saída;
H_{d-hid} : matriz $(r \times m+1)$;
Z_{d-hid} : H_{d-hid} retro-propagado através da camada interna (matriz $r \times m+1$);
W_{hid} : matriz $(m+1 \times m+1)$ de interconecções ponderadas ligando a camada de entrada com a camada interna;
Z_{hid} : produzida pela multiplicação de H_{in} por W_{hid} (matriz $r \times m+1$);
H_{hid} : Z_{hid} propagado através da camada interna (matriz $r \times m+1$);
Z_{out} : produzido pela multiplicação de H_{hid} por W_{out} (matriz $r \times m+1$);
H_{out} : Z_{out} propagado através da camada de saída (matriz $r \times n$);
r : número de padrões entrada-saída do conjunto de treinamento;
$m+1$: número de neurônios de entrada (que é igual ao número de neurônios da camada interna) e
n : número de neurônios de saída.

O procedimento para treinamento de uma rede com uma camada interna será agora descrito. A extensão deste procedimento de treinamento para mais que uma camada interna é imediato.

Passos de treinamento da rede:

1) Calcular o valor inicial de W_{out} (este deverá ser o valor final de W_{out} se camadas internas não são usadas)

$$W_{out} = H_{in}^{\Gamma} Z_{d-out} \tag{2.23}$$

onde

$$Z_{d-out} = \tanh^{-1}(H_{d-out}) \tag{2.24}$$

e

$$H_{in}^{\Gamma} = (H_{in}^{t} H_{in})^{-1} H_{in}^{t} \tag{2.25}$$

assumindo r maior que $m+1$ (sistema sobredimensionado). O símbolo Γ representa a inversa generalizada ou Moore-Penrose inversa.

2) Obter H_{d-hid} que irá produzir Z_{d-out} dado W_{out}

se $n = m+1$,

$$H^t_{d-hid} = (W^t_{out})^{-1}\, Z^t_{d-out} \tag{2.26}$$

assumindo que W^t_{out} é não singular

se $n > m+1$

$$H^t_{d-hid} = (W^t_{out})^{\Gamma}\, Z^t_{d-out} \tag{2.27}$$

$$(W^t_{out})^{\Gamma} = (W_{out}\, W^t_{out})^{-1}\, W_{out} \tag{2.28}$$

e se $n < m+1$

$$H^t_{d-hid} = (W^t_{out})^{\Gamma}\, Z^t_{d-out} \tag{2.29}$$

$$(W^t_{out})^{\Gamma} = W_{out}\, (W^t_{out}\, W_{out})^{-1} \tag{2.30}$$

caso indeterminado.

3) Normalizar os elementos de H_{d-hid} para garantir que eles irão se situar dentro do limite estipulado para as saídas das funções de ativação da camada interna, $H_{d-hid} = H_{d-hid}\ 0{,}99 / |\lambda|$, onde λ é o elemento de H_{d-hid} com maior amplitude. Um fator de multiplicação, em torno de 0,99, é usado para obter valores finitos de Z_{d-hid} e tirar vantagem da não-linearidade da tangente hiperbólica.

4) Produzir Z_{d-hid} usando a inversa da tangente hiperbólica,

$$Z_{d-hid} = \tanh^{-1}(H_{d-hid}) \tag{2.31}$$

5) Dado H_{in} e Z_{d-hid} , calcular W_{hid}

$$W_{hid} = H^{\Gamma}_{in}\, Z_{d-hid} \tag{2.32}$$

6) Obter o valor da saída da camada interna;

$$Z_{hid} = H_{in}\, W_{hid} \tag{2.33}$$

e

$$H_{hid} = \tanh(Z_{hid}) \tag{2.34}$$

7) Recalcular W_{out},

$$W_{out} = H_{hid}^{\Gamma}\, Z_{d-out} \tag{2.35}$$

onde

$$H_{hid}^{\Gamma} = (H_{hid}^{t}\, H_{hid})^{-1}\, H_{hid}^{t} \tag{2.36}$$

O treinamento da rede será finalizado no passo 7, isto é, uma vez as matrizes de interconecções ponderadas W_{hid} e W_{out} estejam otimamente calculadas usando o critério dos mínimos quadrados.

2.2.4 - CMAC ("Cerebellar Model Arithmetic Computer")

A rede neural CMAC é associativa, ou seja, a saída é influenciada por apenas um subconjunto da rede e este é determinado pela entrada. O mapeamento associativo do CMAC assegura a generalização local, na qual entradas similares produzem saídas similares, enquanto entradas distantes produzem saídas quase independentes. Esta propriedade associativa faz com que o número de passos de treinamento, necessários para a convergência da rede, seja menor no CMAC em comparação com o "back-propagation".

CMAC é descrito de uma maneira geométrica na figura 2.6, sendo composto por: espaço de entrada, saídas e mecanismo de generalização.

Cada vetor de entrada possui dimensão N, contendo por exemplo, os sinais vindos de N sensores. O espaço de entrada S é formado pelo conjunto de vetores possíveis. O algoritmo CMAC faz o mapeamento do vetor de entrada em conjuntos de C pontos dentro da memória conceitual A, de tal maneira que; se dois vetores dentro do espaço de entrada forem próximos, tem-se uma certa sobreposição de seus C pontos; sendo esta sobreposição proporcional à distância dos vetores. Portanto, se dois vetores estiverem muito distantes, não haverá sobreposição, e conseqüentemente não haverá generalização. Desde que a

maioria dos problemas de aprendizagem não envolve todo o espaço de entrada, a necessidade de memória é reduzida graças ao mapeamento da memória A em uma memória menor A', e a saída da rede é o resultado da soma do conteúdo dessa memória.

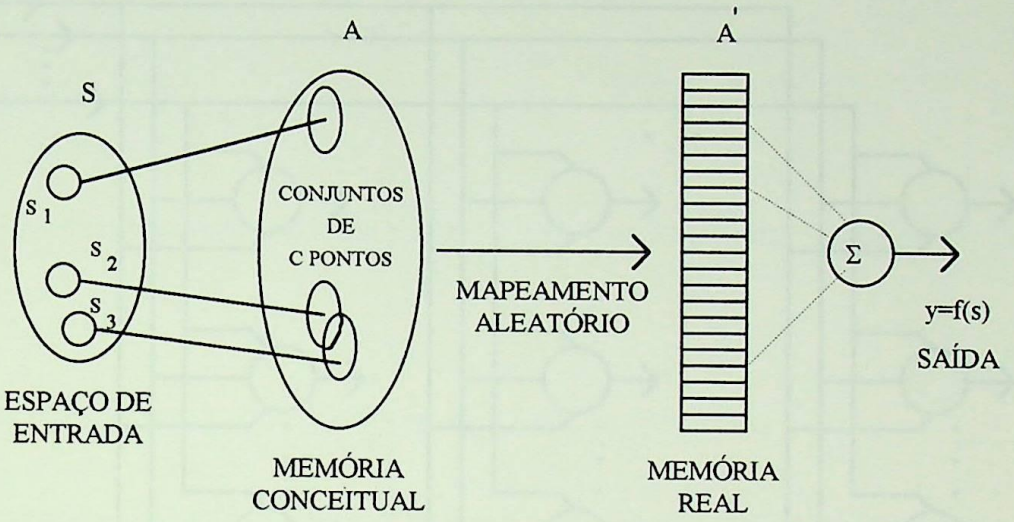


Figura 2.6 - Diagrama de blocos do CMAC.

A aprendizagem da rede é feita de forma supervisionada e utiliza-se uma regra de treinamento baseada nos mínimos quadrados.

Controle de robôs, processamento de sinais e reconhecimento de padrões, são exemplos de aplicação da rede CMAC [7].

2.2.5 - Mapeamento auto-organizado ("self-organizing map")

A rede neural baseada no mapeamento auto-organizado utiliza treinamento não-supervisionado, ou seja, apresenta-se à rede somente os padrões de entrada. A estrutura da rede possui duas camadas, sendo uma de entrada e outra de saída, todos os neurônios que formam as camadas são interconectados e cada conexão possui um peso associado. A camada de saída é representada como sendo um "array" bidimensional de neurônios, como mostra a figura 2.7.

A aprendizagem se baseia nos conceitos de vizinhança dos neurônios, mínima distância Euclidiana entre um vetor de entrada e um vetor de peso, e neurônio "winner-take-all". Após a aprendizagem, pode ser feita uma sintonia fina supervisionada dos pesos utilizando-se o método LVQ ("Learning Vector Quantization") [6].

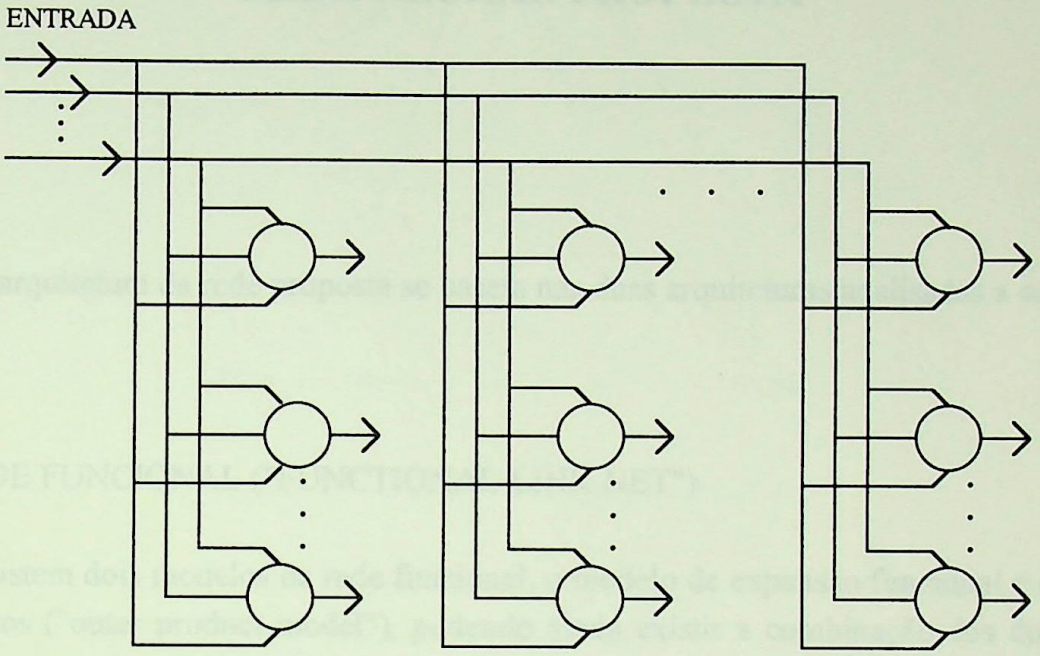


Figura 2.7 - "Array" bidimensional de neurônios.

Controle de processos, robótica e reconhecimento estatístico de padrões, são exemplos de aplicação das redes baseadas no mapeamento auto-organizado.

CAPÍTULO 3

REDE NEURAL PROPOSTA

A arquitetura da rede proposta se baseia nas duas arquiteturas analisadas a seguir.

3.1 - REDE FUNCIONAL ("FUNCTIONAL-LINK NET")

Existem dois modelos de rede funcional, o modelo de expansão funcional e o modelo de produtos ("outer product model"), podendo ainda existir a combinação dos dois. Uma vantagem deste tipo de rede é a simplificação, pois se baseia no conceito de funções ortogonais, combinação de produtos das entradas e rede perceptron com duas camadas, ou seja, não há a necessidade de se usar camadas internas. Portanto, para o seu treinamento pode ser utilizada a regra delta padrão [14].

Uma combinação de funções ortogonais pode ser usada para produzir um mapeamento desejado. No entanto, não há outra maneira além da tentativa e erro para a escolha das funções ortogonais.

A figura 3.1 mostra um exemplo de rede que utiliza o modelo de expansão funcional, onde:

$$y = f(x_i w_1 + \text{sen}(\pi x_i) w_2 + \cos(\pi x_i) w_3 + x_j w_4 + \text{sen}(\pi x_j) w_5 + \cos(\pi x_j) w_6 + \theta),$$

seno e co-seno são funções de base ortonormais linearmente independentes e f é uma função sigmóide. Simplesmente expandiu-se o número de entradas da rede utilizando-se as funções ortogonais seno e co-seno. Deve-se salientar que, nenhuma informação nova é adicionada à rede, somente expandiu-se a dimensão do espaço de representação dos padrões de entrada de modo que uma rede sem camadas intermediárias pudesse ser utilizada. Neste exemplo, o uso de $\text{sen}(\pi x)$ e $\cos(\pi x)$ é suficiente, porém outros problemas poderão exigir dimensões maiores como por exemplo $\text{sen}(\pi x)$, $\cos(\pi x)$, $\text{sen}(2\pi x)$, $\cos(2\pi x)$, $\text{sen}(3\pi x)$, $\cos(3\pi x)$, etc.

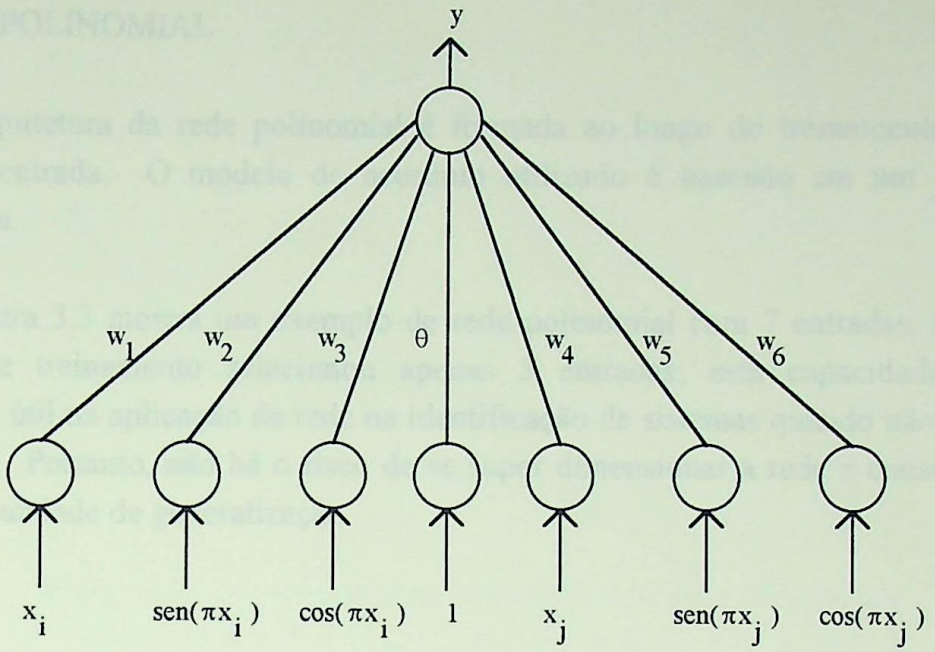


Figura 3.1 - Modelo de expansão funcional.

A figura 3.2 mostra um exemplo de rede funcional que utiliza o modelo de produtos ("outer product model"). Uma combinação de produtos pode ser usada para produzir um mapeamento desejado, no entanto, não há outra maneira além da tentativa e erro para a escolha da dimensão da combinação.

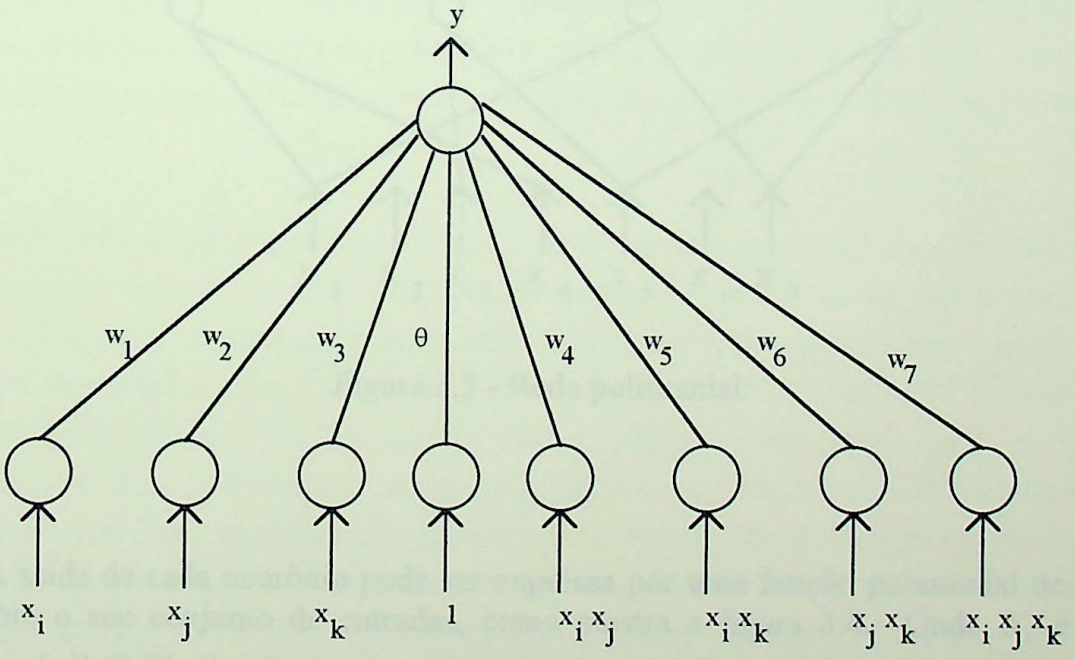


Figura 3.2 - Modelo de produtos.

3.2 - REDE POLINOMIAL

A arquitetura da rede polinomial é formada ao longo do treinamento, a partir da camada de entrada. O modelo de neurônio utilizado é baseado em um polinômio de segundo grau.

A figura 3.3 mostra um exemplo de rede polinomial com 7 entradas, nota-se que o algoritmo de treinamento selecionou apenas 5 entradas; esta capacidade de seleção automática é útil na aplicação da rede na identificação de sistemas quando não se conhece a ordem deste. Portanto, não há o risco de se super dimensionar a rede e conseqüentemente perder a capacidade de generalização.

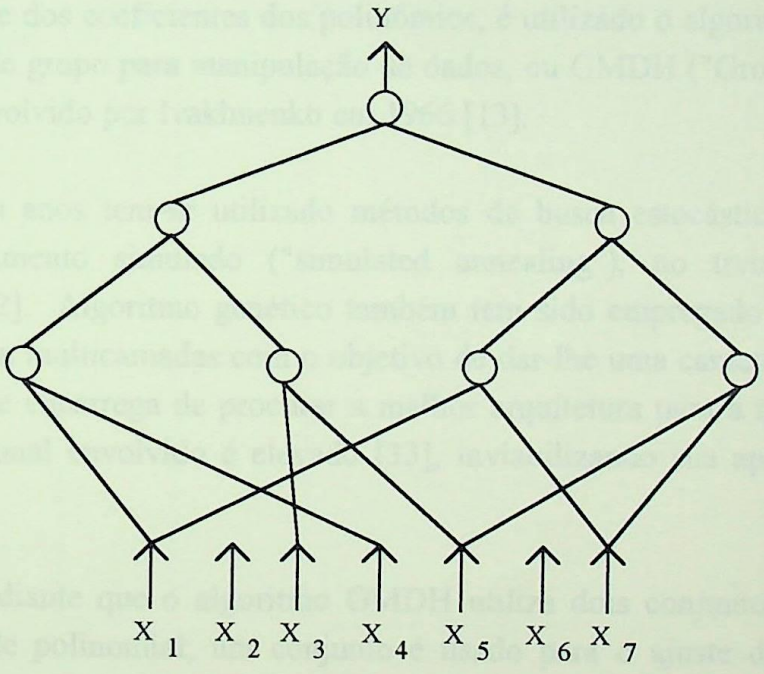


Figura 3.3 - Rede polinomial.

A saída de cada neurônio pode ser expressa por uma função polinomial de segundo grau sobre o seu conjunto de entradas, como mostra a figura 3.4. Onde X_i e X_j são entradas; A, B, C, D, E e F são os coeficientes do polinômio, que equivalem aos pesos, e Y é a saída do neurônio.

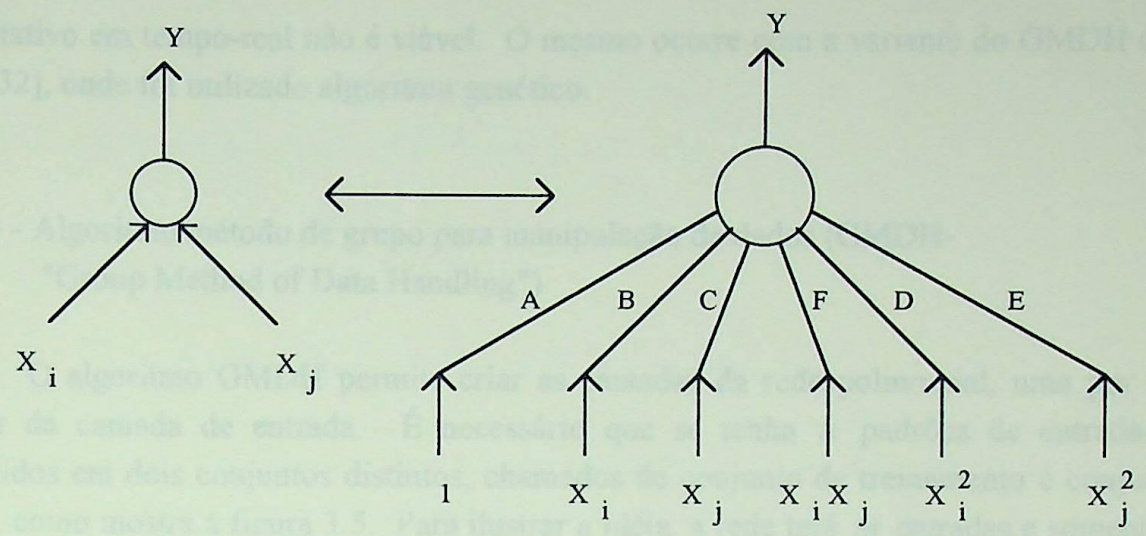


Figura 3.4 - Modelo de neurônio usado pela rede polinomial, onde $Y = A + B\,X_i + C\,X_j + D\,X_i^2 + E\,X_j^2 + F\,X_i\,X_j$.

Para o ajuste dos coeficientes dos polinômios, é utilizado o algoritmo de treinamento chamado método de grupo para manipulação de dados, ou GMDH ("Group Method of Data Handling"), desenvolvido por Ivakhnenko em 1966 [13].

Nos últimos anos tem-se utilizado métodos de busca estocástica, como algoritmo genético e recozimento simulado ("simulated annealing"), no treinamento de redes polinomiais [16, 32]. Algoritmo genético também tem sido empregado no treinamento da rede tipo perceptron multicamadas com o objetivo de dar-lhe uma característica plástica, ou seja, o algoritmo se encarrega de procurar a melhor arquitetura para a rede. Entretanto, o esforço computacional envolvido é elevado [33], inviabilizando sua aplicação em tempo-real.

Será visto adiante que o algoritmo GMDH utiliza dois conjuntos de dados para o treinamento da rede polinomial, um conjunto é usado para o ajuste dos parâmetros dos neurônios e outro é usado para avaliar quais neurônios devem permanecer na rede e quais devem ser descartados. Em [16] é mostrado o algoritmo GMDH-SONN ("Self-Organizing Network for Optimun Supervised Learning", Rede Auto-Organizável para Ótima Aprendizagem Supervisionada), este é uma variante do algoritmo GMDH baseada em recozimento simulado, e utiliza um critério que é função do erro da saída do neurônio e também da complexidade da arquitetura da rede em treinamento, o que dispensa o uso do conjunto de teste. No entanto, a eliminação do conjunto de teste, em redes que se utilizam de critérios como o do SONN, não garante mais a obtenção da arquitetura ótima. Além disso, o algoritmo é muito lento, e portanto, a sua utilização em sistemas de controle

adaptativo em tempo-real não é viável. O mesmo ocorre com a variante do GMDH contida em [32], onde foi utilizado algoritmo genético.

3.2.1 - Algoritmo método de grupo para manipulação de dados (GMDH- "Group Method of Data Handling")

O algoritmo GMDH permite criar as camadas da rede polinomial, uma por vez, a partir da camada de entrada. É necessário que se tenha n padrões de entrada-saída, divididos em dois conjuntos distintos, chamados de conjunto de treinamento e conjunto de teste, como mostra a figura 3.5. Para ilustrar a idéia, a rede terá m entradas e somente uma saída.

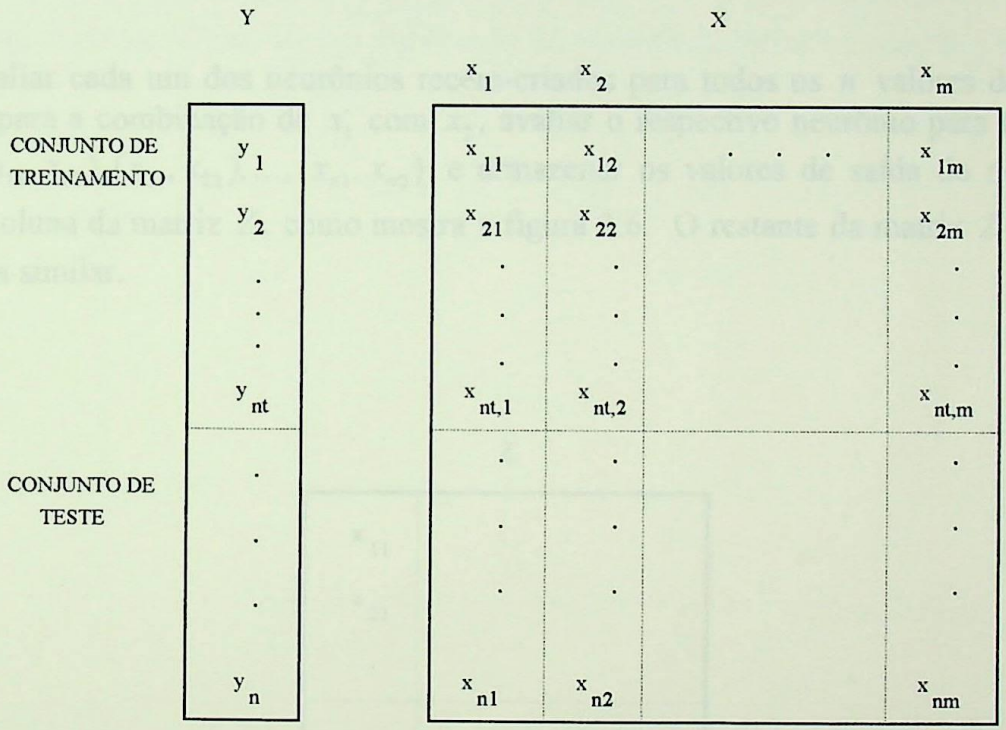


Figura 3.5 - Padrões de entrada-saída.

O algoritmo pode ser descrito através dos seguintes passos de execução:

1) Fazer a combinação dois a dois de todas as colunas de X, utilizando apenas o conjunto de treinamento; com isso tem-se $m(m-1)/2$ novos neurônios na nova camada da rede. Os coeficientes que formarão um determinado neurônio serão gerados resolvendo-se um sistema de equações lineares. Por exemplo, ao se combinar a primeira coluna com a segunda coluna de X, tem-se o seguinte sistema de equações:

$$
\begin{aligned}
y_1 &= A + Bx_{11} + Cx_{12} + Dx_{11}^2 + Ex_{12}^2 + Fx_{11}x_{12} \\
y_2 &= A + Bx_{21} + Cx_{22} + Dx_{21}^2 + Ex_{22}^2 + Fx_{21}x_{22} \\
&\cdot \qquad\qquad \cdot \\
&\cdot \qquad\qquad \cdot \\
&\cdot \qquad\qquad \cdot \\
y_{nt} &= A + Bx_{nt1} + Cx_{nt2} + Dx_{nt1}^2 + Ex_{nt2}^2 + Fx_{nt1}x_{nt2}
\end{aligned}
\tag{3.1}
$$

Resolvendo o sistema pelo critério de mínimos-quadrados, tem-se os valores de A, B, C, D, E e F, e conseqüentemente, o neurônio será representado pela equação:

$$y = \hat{A} + \hat{B}x_1 + \hat{C}x_2 + \hat{D}x_1^2 + \hat{E}x_2^2 + \hat{F}x_1x_2 \tag{3.2}$$

Avaliar cada um dos neurônios recém-criados para todos os n valores de X. Por exemplo, para a combinação de x_1 com x_2, avaliar o respectivo neurônio para todos os n valores $(x_{11}, x_{12}), (x_{21}, x_{22}), \ldots, (x_{n1}, x_{n2})$ e armazenar os valores de saída do neurônio na primeira coluna da matriz Z, como mostra a figura 3.6. O restante da matriz Z é formada de maneira similar.

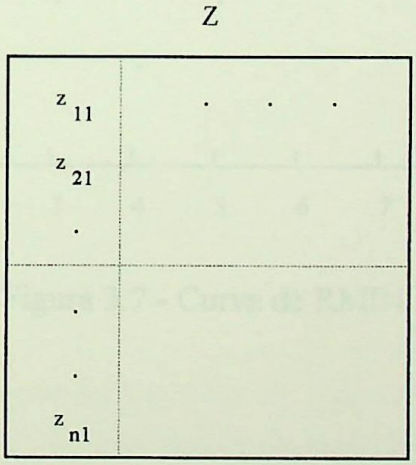


Figura 3.6 - Matriz Z.

O próximo passo tem como objetivo, escolher os neurônios que melhor representam os valores de Y.

2) Para cada coluna j de Z aplicar a seguinte equação:

$$r_j^2 = \frac{\sum_{i=nt+1}^{n} (y_i - z_{ij})^2}{\sum_{i=nt+1}^{n} y_i^2} \qquad j = 1,2,\ldots,\frac{m(m-1)}{2} \tag{3.3}$$

também chamada de critério da regularidade [13]. Ordenar as colunas de Z, em ordem crescente de r_j, de tal modo que a primeira coluna de Z corresponda ao menor valor de r_j. Substituir então, as colunas de X pelas $(m + pi.m)$ primeiras colunas de Z, obtendo-se assim, uma nova matriz X; o valor de pi deve ser $0 \leq pi \leq 1$.

3) Armazenar o menor valor de r_j na variável RMIN. A figura 3.7 mostra um exemplo típico de curva para RMIN em função do número de camadas criadas, ou iterações.

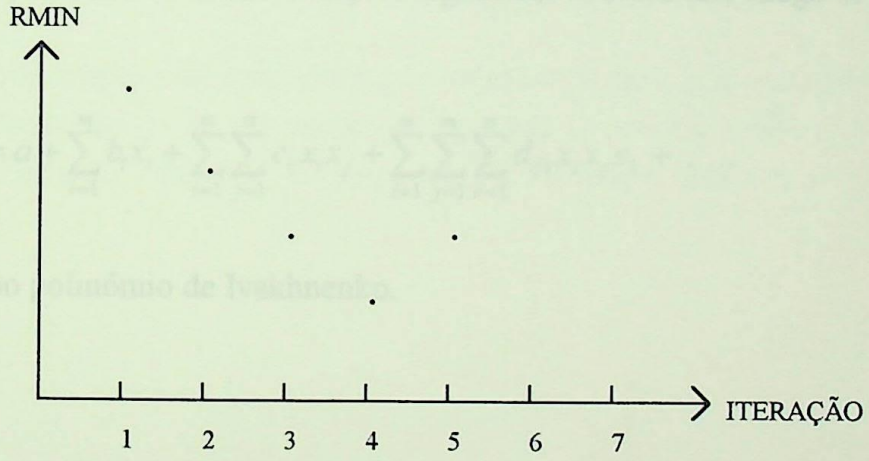


Figura 3.7 - Curva de RMIN.

Se o valor atual de RMIN é menor que o valor de RMIN da camada anterior, então, executar os passos 1 e 2, criando-se assim, uma nova camada para a rede. Porém, se o valor atual de RMIN for maior, assumir que a curva de RMIN alcançou o seu valor mínimo na camada anterior. Portanto, deve-se utilizar os resultados gerados até a camada anterior e parar o processo de treinamento. A figura 3.8 mostra a arquitetura da rede criada.

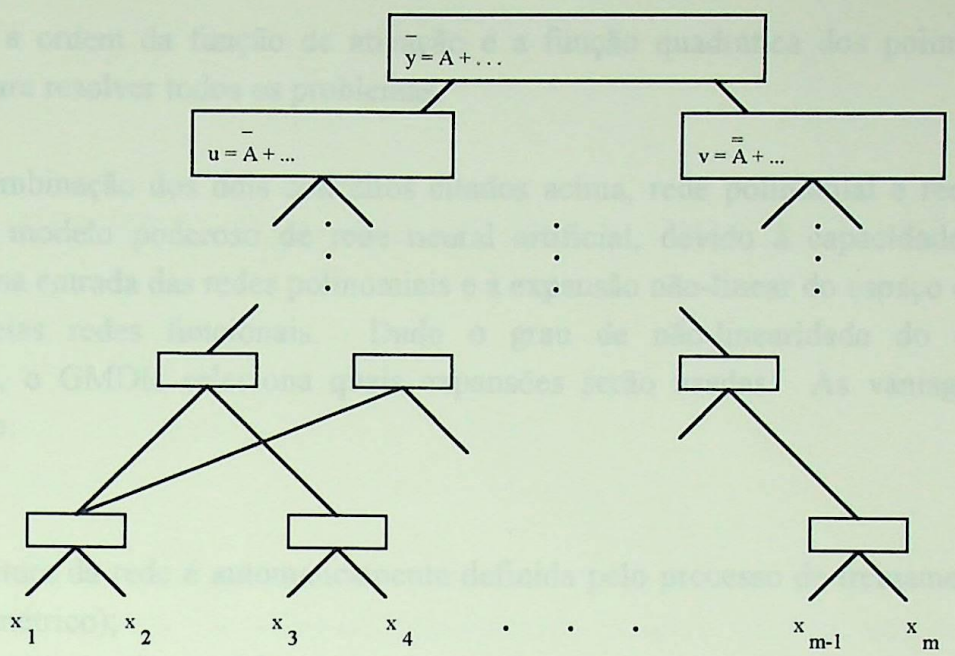


Figura 3.8 - Árvore de polinômios.

Portanto, fazendo-se as substituições algébricas necessárias, chega-se a um polinômio da forma:

$$\bar{y} = a + \sum_{i=1}^{m} b_i x_i + \sum_{i=1}^{m}\sum_{j=1}^{m} c_{ij} x_i x_j + \sum_{i=1}^{m}\sum_{j=1}^{m}\sum_{k=1}^{m} d_{ijk} x_i x_j x_k + \ldots \qquad (3.4)$$

conhecido como polinômio de Ivakhnenko.

3.3 - COMBINANDO REDE FUNCIONAL COM REDE POLINOMIAL

O GMDH é um modelo não-paramétrico de rede neural, ou seja, não necessita de uma arquitetura pré-especificada e de parâmetros de aprendizagem para serem ajustados. A rede funcional "functional-link net" baseia-se no conceito de funções ortogonais, série de Fourier, combinação de produtos das entradas e rede perceptron com duas camadas, obtendo-se com isso uma rede de arquitetura simplificada. Uma combinação de funções ortogonais pode ser usada para produzir um mapeamento desejado. No entanto, não há outra maneira além da tentativa e erro para a escolha das funções ortogonais. As redes polinomiais também podem ser usadas para aproximação de funções, porém, não é fácil

estabelecer a ordem da função de ativação e a função quadrática dos polinômios não é adequada para resolver todos os problemas.

A combinação dos dois conceitos citados acima, rede polinomial e rede funcional, produz um modelo poderoso de rede neural artificial, devido à capacidade de seleção automática na entrada das redes polinomiais e à expansão não-linear do espaço das entradas, efetuada pelas redes funcionais. Dado o grau de não-linearidade do conjunto de treinamento, o GMDH seleciona quais expansões serão usadas. As vantagens da rede proposta são:

- A arquitetura da rede é automaticamente definida pelo processo de treinamento (modelo não-paramétrico);
- Não há parâmetro de aprendizagem para ser ajustado;
- Sem problemas de mínimos locais;
- Convergência rápida.

Uma desvantagem da rede proposta para controle é que o esquema de controle adaptativo indireto não pode ser usado, isto porque o erro de saída não pode ser retro-propagado. Portanto, somente o esquema de controle adaptativo direto pode ser implementado.

CAPÍTULO 4

REDES NEURAIS EM SISTEMAS DE CONTROLE

4.1 - INTRODUÇÃO

As redes neurais possuem características importantes, as quais são úteis na modelagem e controle de sistemas não-lineares; pode-se citar a não-linearidade, a capacidade de aprendizagem, a capacidade de generalização, processamento paralelo intrínseco e a capacidade de aproximar funções contínuas não-lineares. Com relação a aproximação de funções, a precisão está ligada a questões chaves como, quantas camadas internas a rede deverá possuir e quantos neurônios são necessários em cada camada.

Este capítulo visa mostrar como as redes neurais artificiais podem ser utilizadas em modelos de identificação e controle de sistemas dinâmicos não-lineares.

4.2 - IDENTIFICAÇÃO DE SISTEMAS

Baseando-se na referência [3], tem-se dois tipos de modelagem, ou seja, modelagem para frente ("forward modelling") e modelagem inversa. Estes tipos de modelagem são utilizados no treinamento das redes, para que as mesmas possam representar direta ou inversamente os sistemas dinâmicos não-lineares.

4.2.1 - Modelagem para a frente ("forward modelling")

A finalidade da modelagem para a frente é criar um procedimento de treinamento para representar a dinâmica de um sistema, a figura 4.1 mostra a estrutura básica para este tipo de modelagem.

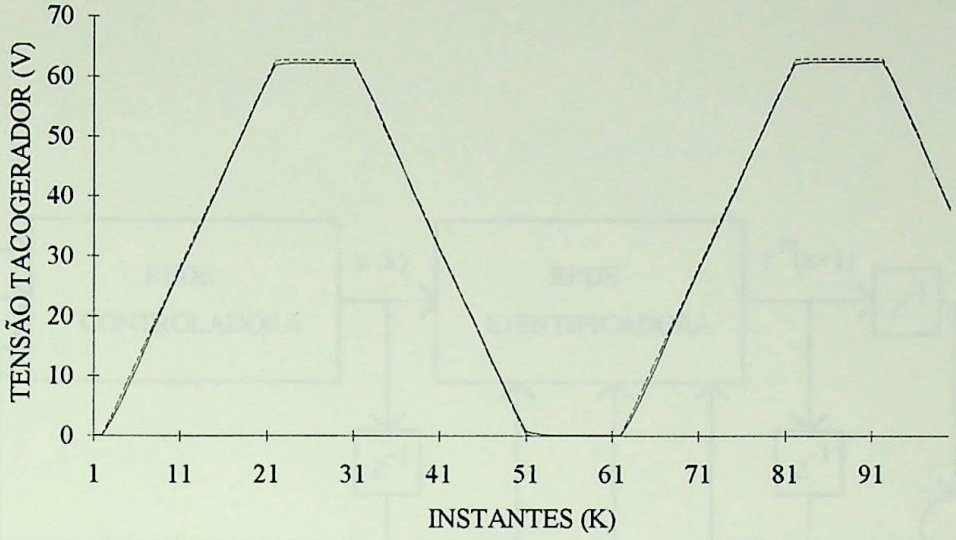


Figura 5.2 - Resposta da rede ao sinal de treinamento.

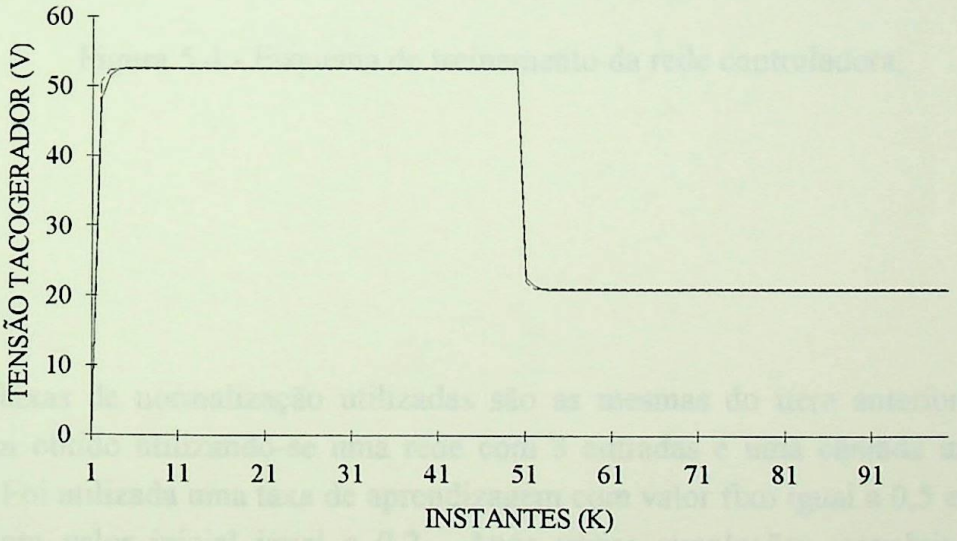


Figura 5.3 - Resposta da rede ao sinal de teste.

5.2.2 - Treinamento da rede controladora

O procedimento de treinamento da rede controladora foi explicado no capítulo 4. A figura 5.4 mostra o esquema de treinamento da rede, onde z^{-1} é um elemento de atraso.

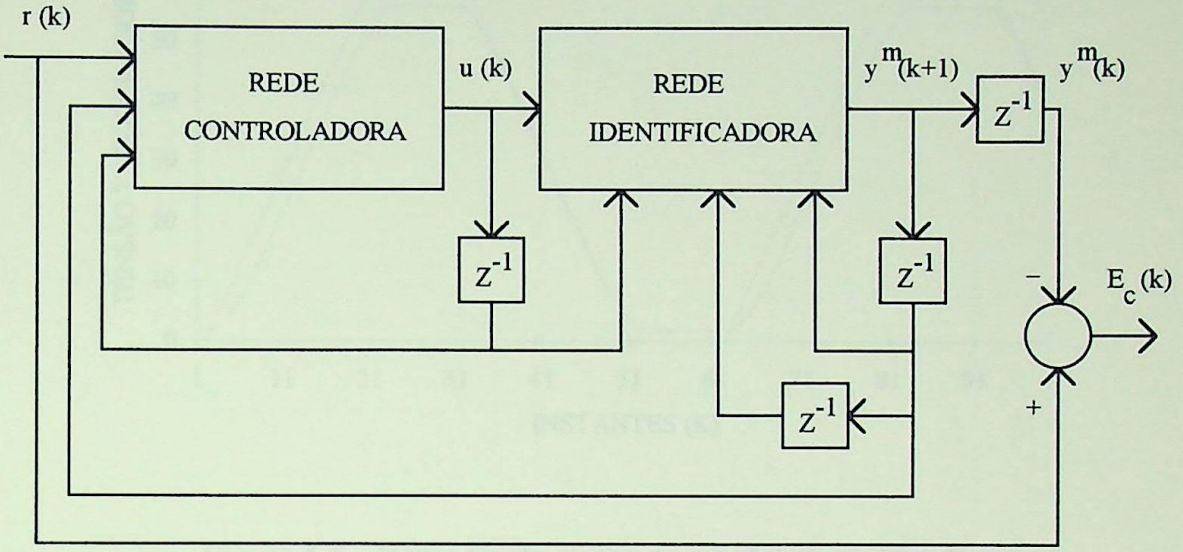


Figura 5.4 - Esquema de treinamento da rede controladora.

As faixas de normalização utilizadas são as mesmas do item anterior. O melhor resultado foi obtido utilizando-se uma rede com 3 entradas e uma camada interna com 3 neurônios. Foi utilizada uma taxa de aprendizagem com valor fixo igual a 0,5 e uma taxa de momento com valor inicial igual a 0,2. Após várias simulações, concluiu-se que 100 iterações eram suficientes para se evitar o super ajustamento dos pesos da rede controladora e grandes oscilações na saída da rede identificadora; foi obtido um erro médio quadrático igual a $7{,}13 \times 10^{-4}$. As figuras 5.5 e 5.6 mostram, respectivamente, um gráfico com a resposta da rede identificadora ao sinal de treinamento e um gráfico com a resposta a um sinal de referência de teste formado por dois degraus, de 52 e 21 volts respectivamente. A linha tracejada representa o sinal de referência e a linha contínua representa a saída da rede identificadora. Cabe aqui lembrar que o sinal de saída deve sempre acompanhar o sinal de referência.

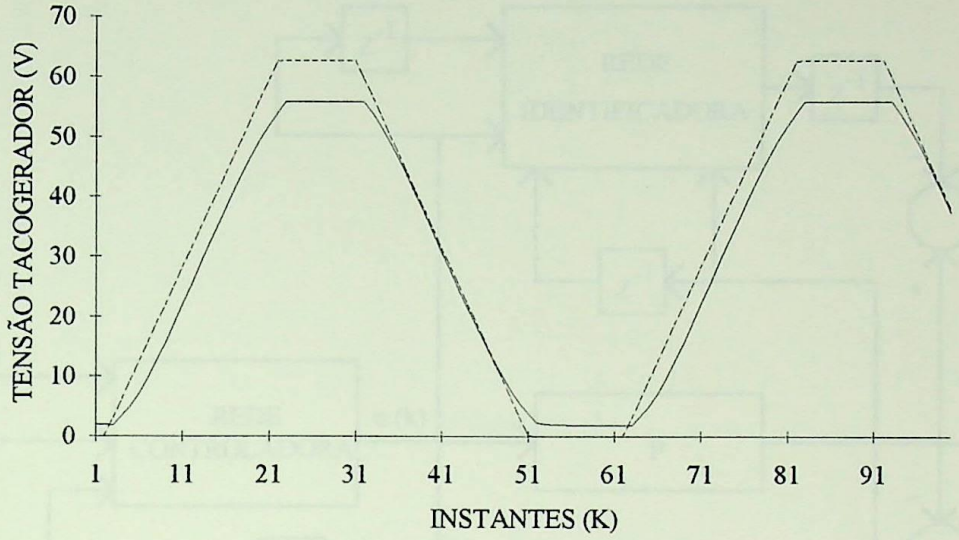


Figura 5.5 - Resposta da malha ao sinal de treinamento.

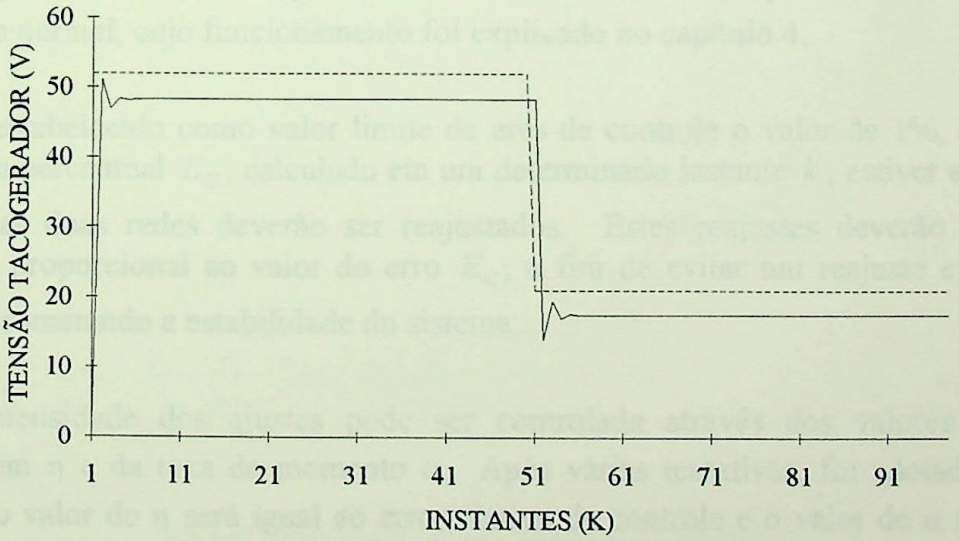


Figura 5.6 - Resposta da malha ao sinal de teste.

5.2.3 - Simulação do sistema de controle em regime de operação normal

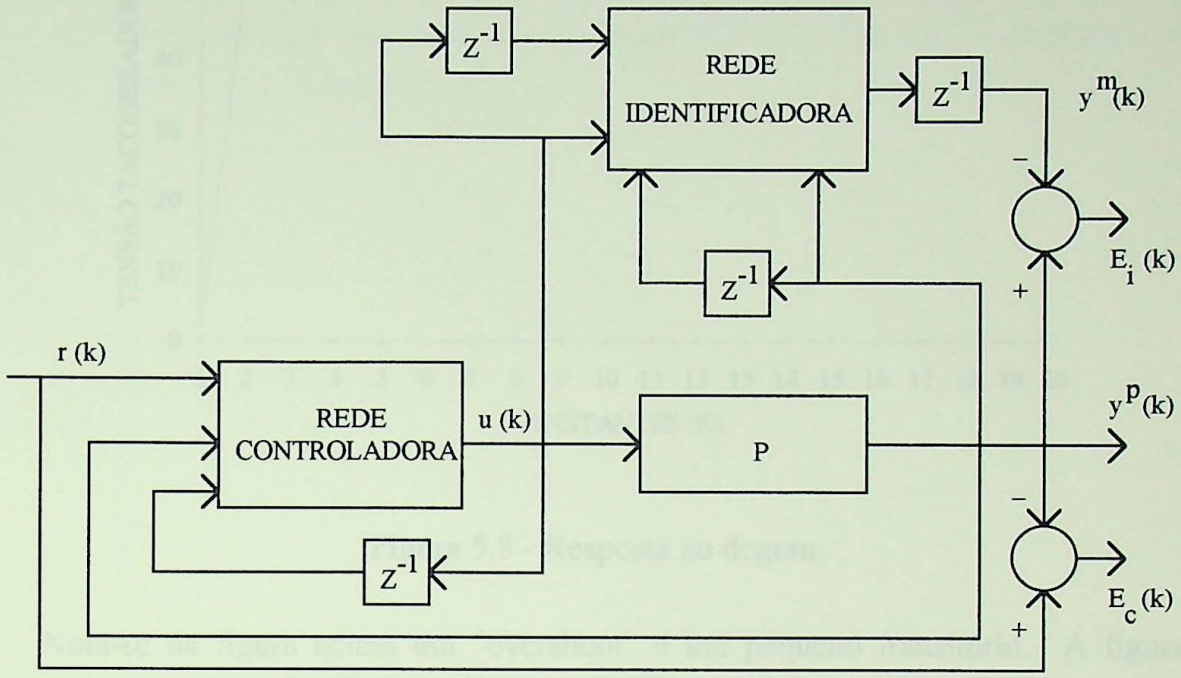


Figura 5.7 - Esquema sistema de controle adaptativo indireto.

A figura 5.7 mostra o esquema do sistema de controle adaptativo indireto em regime de operação normal, cujo funcionamento foi explicado no capítulo 4.

Foi estabelecido como valor limite de erro de controle o valor de 1%, ou seja, se o erro relativo percentual E_C, calculado em um determinado instante k, estiver acima de 1%, os pesos das duas redes deverão ser reajustados. Estes reajustes deverão ocorrer com intensidade proporcional ao valor do erro E_C, a fim de evitar um reajuste excessivo que acabe comprometendo a estabilidade do sistema.

A intensidade dos ajustes pode ser controlada através dos valores da taxa de aprendizagem η e da taxa de momento α. Após várias tentativas, foi adotada a seguinte heurística: o valor de η será igual ao erro relativo de controle e o valor de α será igual ao mesmo erro vezes 0,625. Não há necessidade de se usar o algoritmo adaptativo, portanto, usou-se o algoritmo de retro-propagação básico [29].

A figura 5.8 mostra a resposta da malha de controle a um degrau de 50 volts.

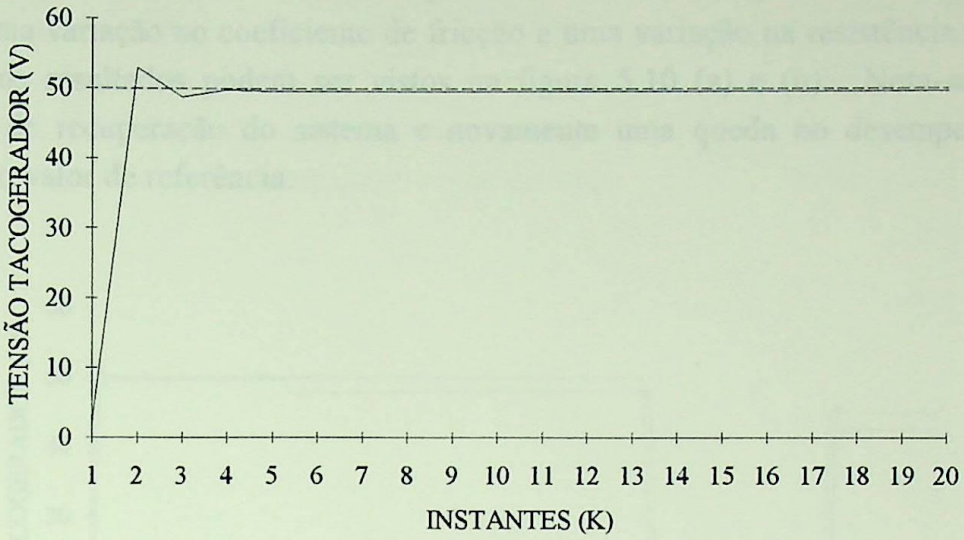


Figura 5.8 - Resposta ao degrau.

Nota-se na figura acima um "overshoot" e um pequeno transitório. A figura 5.9 mostra a resposta da malha de controle a quatro valores diferentes de referência.

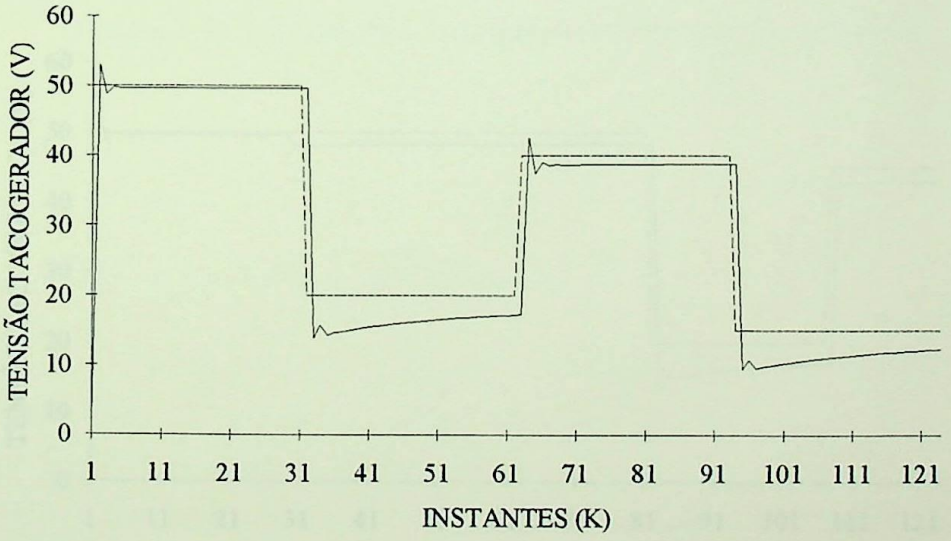
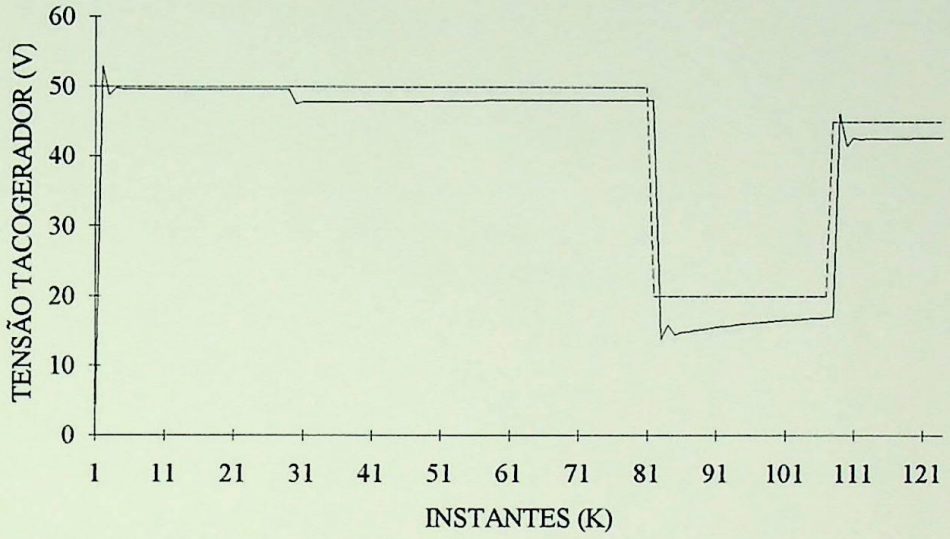


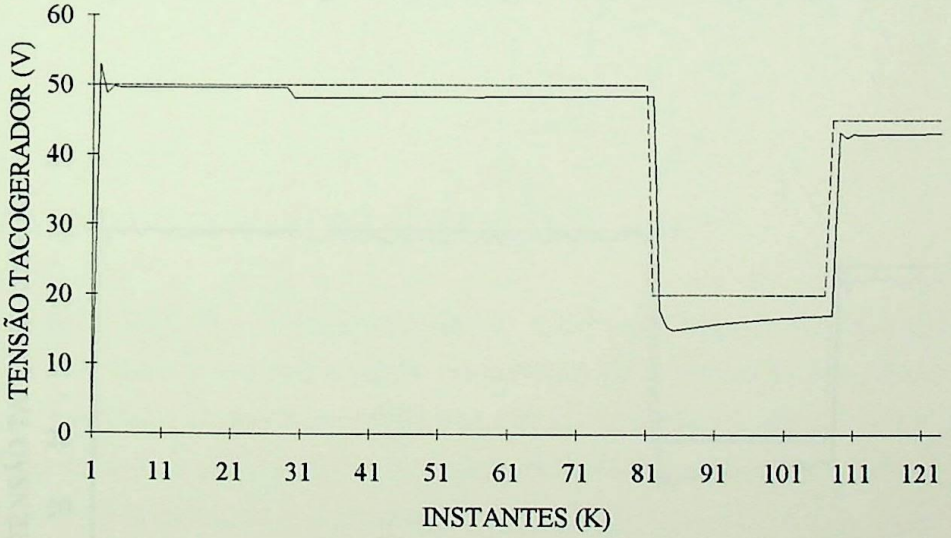
Figura 5.9 - Resposta à variação da referência.

Na figura acima pode-se notar uma queda no desempenho do sistema de controle toda vez que diminui-se o valor de referência.

Para verificar como o sistema reage a uma variação de parâmetro e referência, foi simulada uma variação no coeficiente de fricção e uma variação na resistência de armadura do motor, os resultados podem ser vistos na figura 5.10 (a) e (b). Nota-se uma baixa velocidade de recuperação do sistema e novamente uma queda no desempenho quando diminui-se o valor de referência.



(a)



(b)

Figura 5.10 - Resposta à variação de parâmetros.
(a) Variação do coeficiente de fricção B em 100%;
(b) Variação da resistência de armadura Ra em 80%.

Um dos problemas mais sérios em controle é a presença de ruído no sistema controlado e em sensores. Para testar a imunidade ao ruído, foi introduzido um sinal aleatório variando de -4 a 4 volts na saída do tacogerador. A figura 5.11 mostra a resposta da malha de controle à variação na referência e as figuras 5.12 (a) e (b) mostram a resposta à variação de parâmetro e referência na presença de ruído.

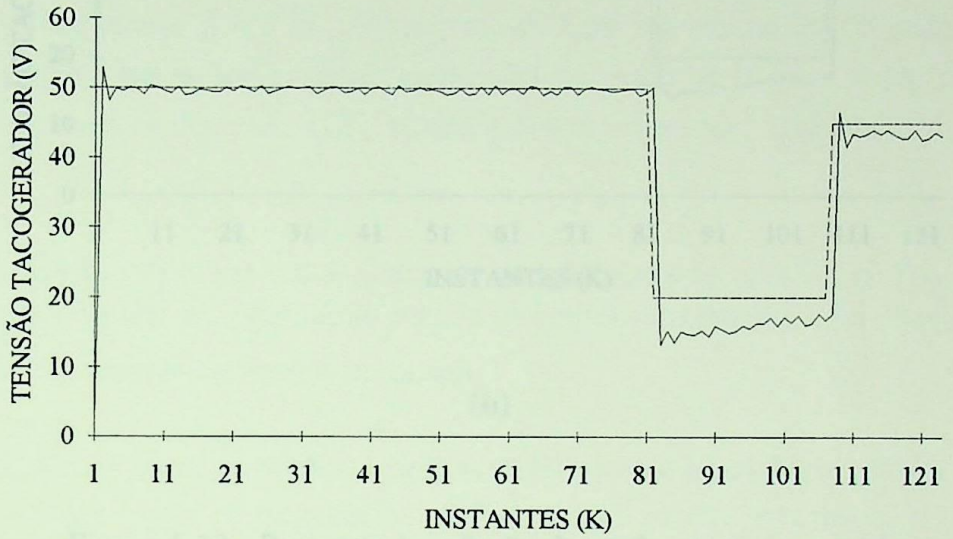
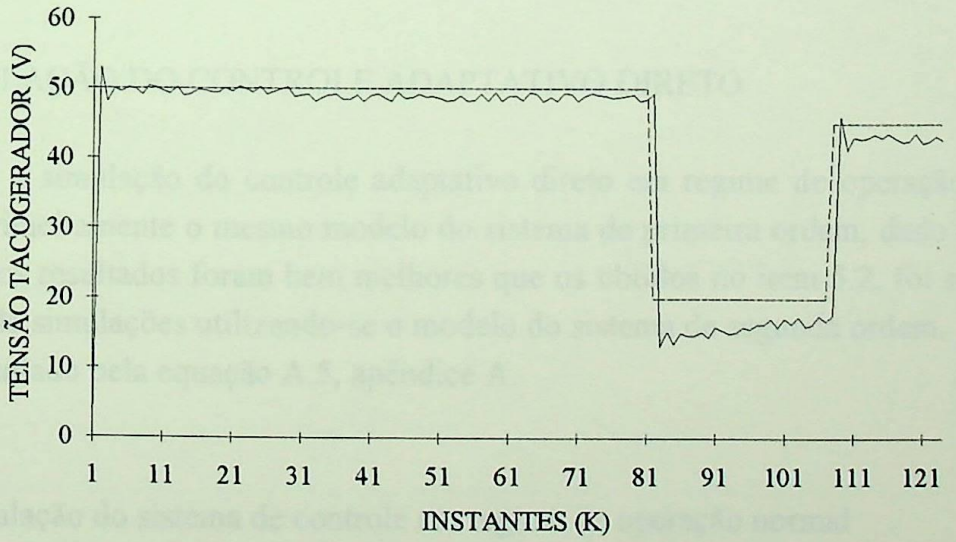
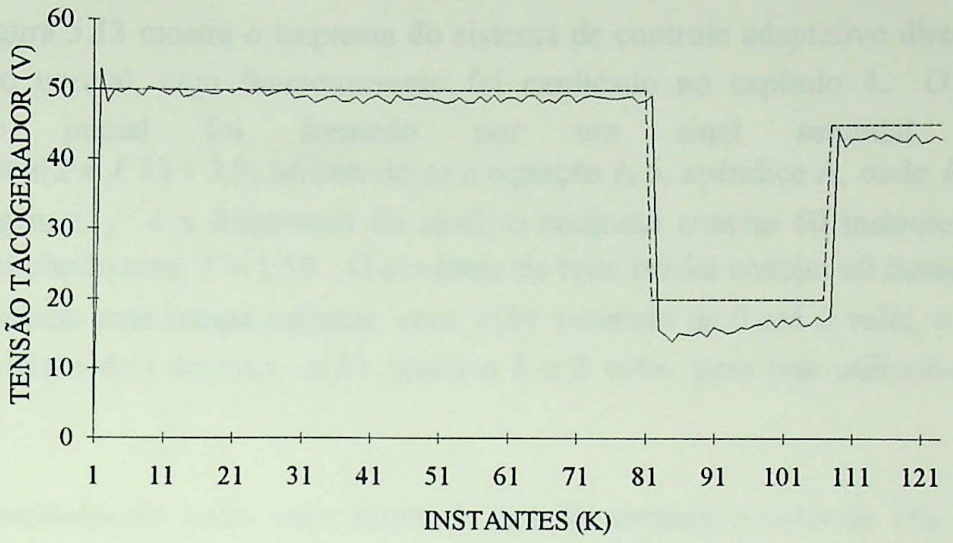


Figura 5.11 - Resposta à variação na referência, com ruído.



(a)



(b)

Figura 5.12 - Resposta à variação de parâmetros, com ruído.
(a) Variação do coeficiente de fricção B em 50%;
(b) Variação da resistência de armadura Ra em 50%.

5.3 - SIMULAÇÃO DO CONTROLE ADAPTATIVO DIRETO

Para a simulação do controle adaptativo direto em regime de operação normal, foi utilizado primeiramente o mesmo modelo do sistema de primeira ordem, dado pela equação A.6; como os resultados foram bem melhores que os obtidos no item 5.2, foi realizada uma nova série de simulações utilizando-se o modelo do sistema de segunda ordem. Este modelo está representado pela equação A.5, apêndice A.

5.3.1 - Simulação do sistema de controle em regime de operação normal

Os próximos itens mostrarão os resultados das simulações utilizando-se os modelos do sistema de primeira e segunda ordem.

5.3.1.1 - Simulações envolvendo o modelo de primeira ordem

A figura 5.13 mostra o esquema do sistema de controle adaptativo direto em regime de operação normal, cujo funcionamento foi explicado no capítulo 4. O conjunto de treinamento inicial foi formado por um sinal senoidal dado por $u(k) = 3{,}0\,\mathrm{sen}(2\,\pi\,f\,k) + 3{,}0$, utilizando-se a equação A.6, apêndice A, onde k é o instante de amostragem e f é a freqüência do sinal; o conjunto contém 60 instantes e o melhor resultado foi obtido com $f = 1/50$. O conjunto de teste inicial contém 60 instantes, onde 30 instantes formam uma rampa patamar, com $u(k)$ variando de 0 até 6 volts, e os outros 30 instantes formam dois degraus, $u(k)$ iguais a 5 e 2 volts; para isso utilizou-se a equação A.6.

Foi estabelecido como valor limite de erro de controle o valor de 1%, ou seja, se o erro relativo percentual E_C, calculado em um determinado instante k, estiver acima de 1%, uma nova rede controladora deverá ser criada.

A figura 5.14 mostra a resposta da malha de controle a um degrau de 50 volts, nota-se a ausência de "overshoot" e transitório. A figura 5.15 mostra a resposta a quatro valores diferentes de referência.

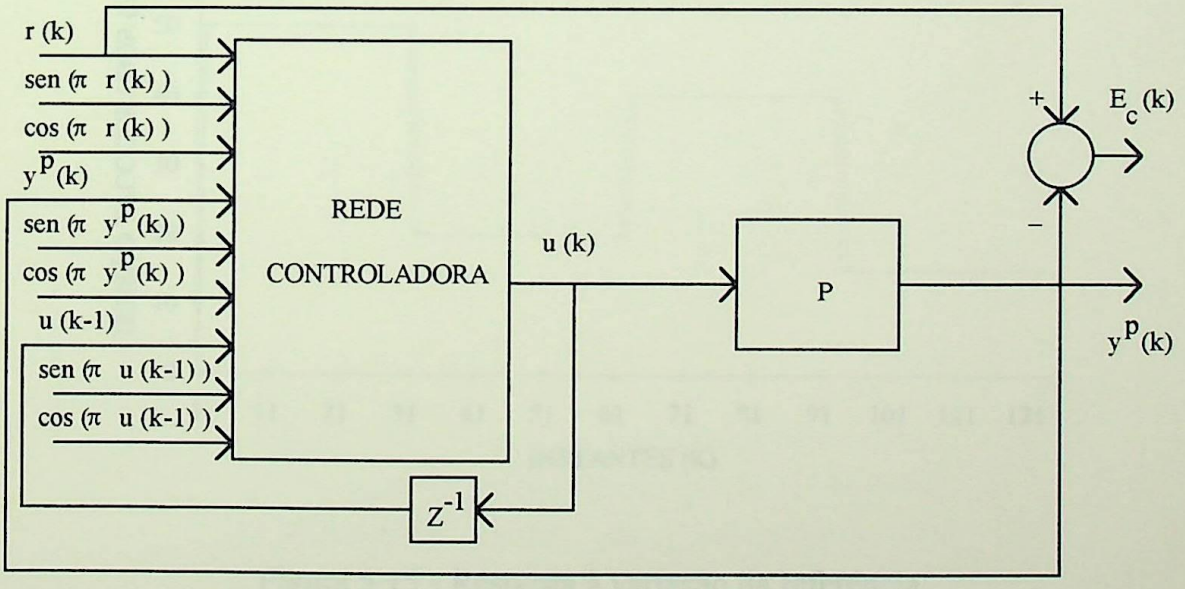


Figura 5.13 - Esquema do sistema de controle

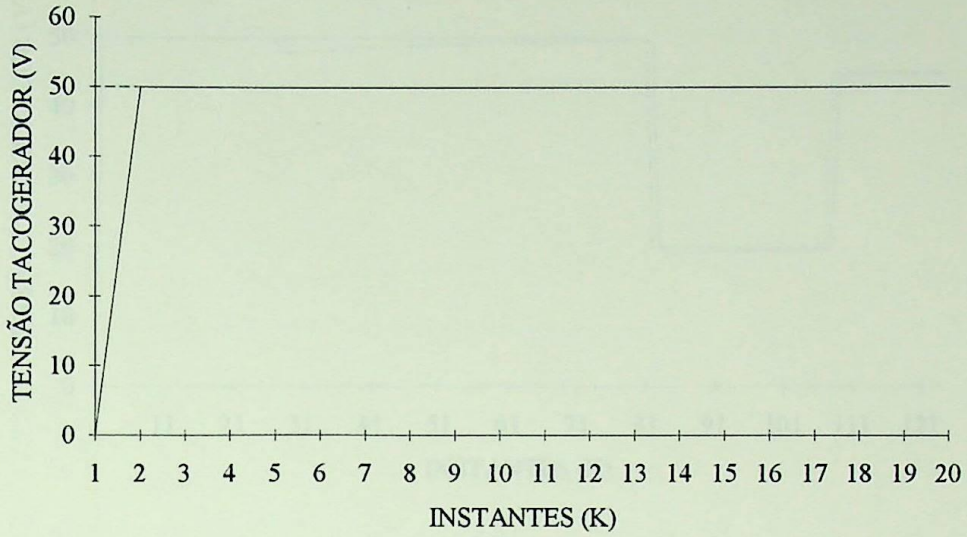


Figura 5.14 - Resposta ao degrau.

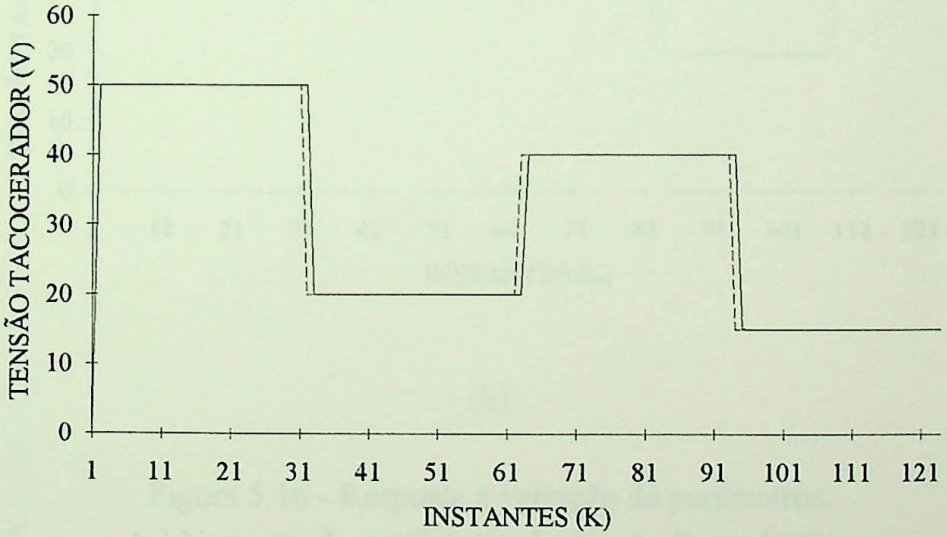
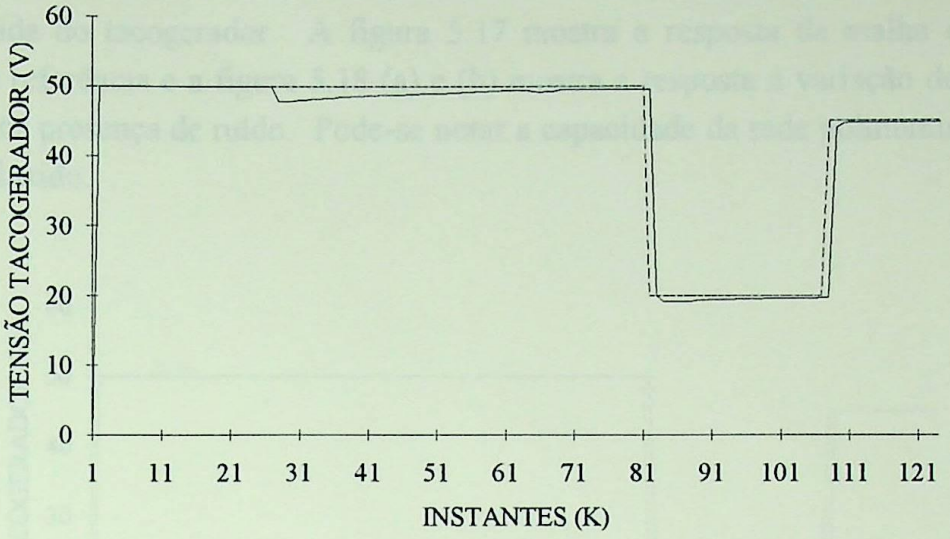
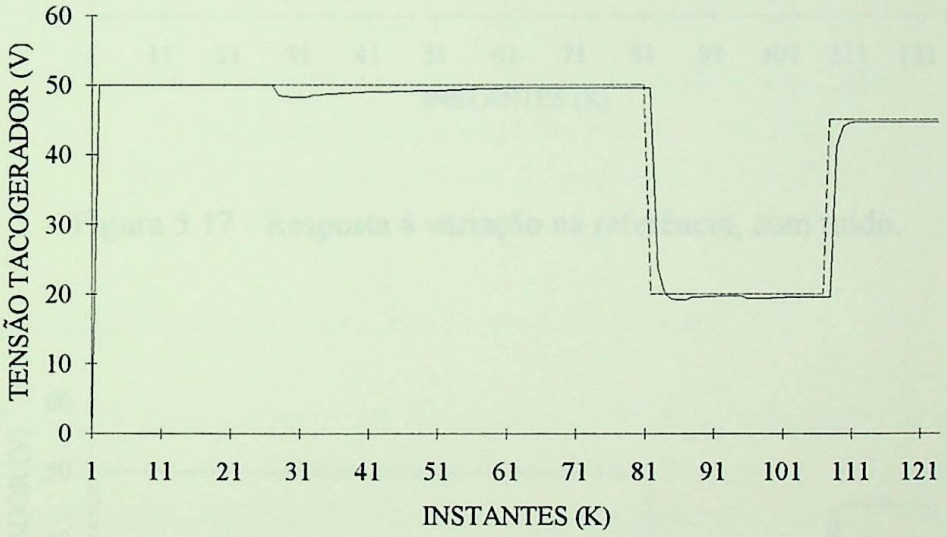


Figura 5.15 - Resposta à variação da referência.



(a)



(b)

Figura 5.16 - Resposta à variação de parâmetros.
(a) Variação do coeficiente de fricção B em 100%;
(b) Variação da resistência de armadura Ra em 80%.

Para verificar como o sistema reage a uma variação de parâmetro e referência, foi simulada uma variação no coeficiente de fricção e uma variação na resistência de armadura do motor, os resultados podem ser vistos na figura 5.16 (a) e (b), onde nota-se uma boa velocidade de recuperação do sistema.

Para testar a imunidade ao ruído, foi introduzido um sinal aleatório variando de -4 a 4 volts na saída do tacogerador. A figura 5.17 mostra a resposta da malha de controle à variação na referência e a figura 5.18 (a) e (b) mostra a resposta à variação de parâmetro e referência, na presença de ruído. Pode-se notar a capacidade da rede polinomial em filtrar o ruído introduzido.

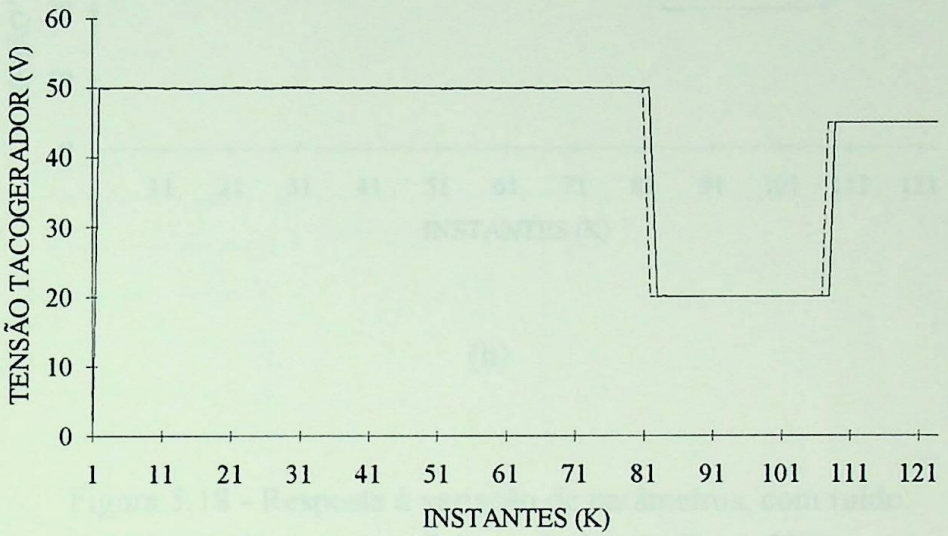
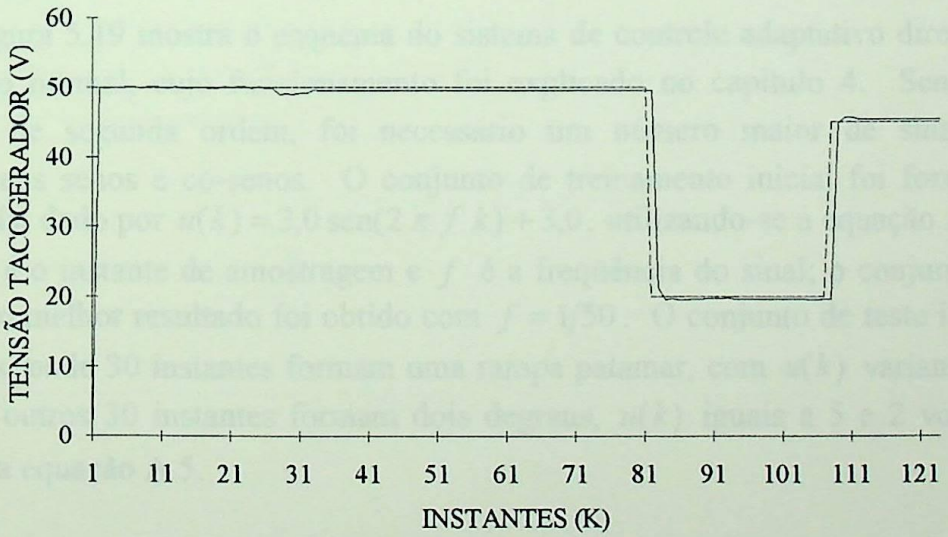
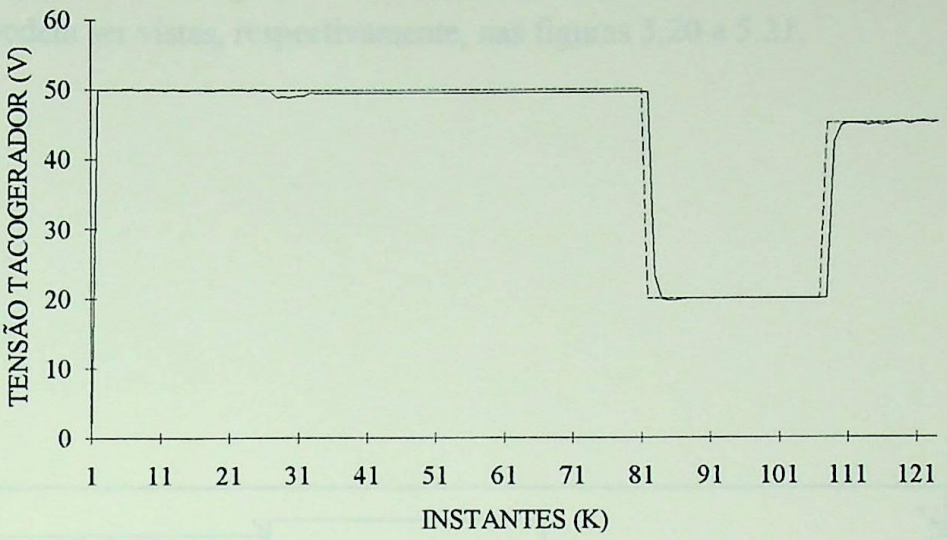


Figura 5.17 - Resposta à variação na referência, com ruído.



(a)



(b)

Figura 5.18 - Resposta à variação de parâmetros, com ruído.
(a) Variação do coeficiente de fricção B em 50%;
(b) Variação da resistência de armadura Ra em 50%.

5.3.1.2 - Simulações envolvendo o modelo de segunda ordem

A figura 5.19 mostra o esquema do sistema de controle adaptativo direto em regime de operação normal, cujo funcionamento foi explicado no capítulo 4. Sendo o sistema controlado de segunda ordem, foi necessário um número maior de sinais atrasados, incluindo seus senos e co-senos. O conjunto de treinamento inicial foi formado por um sinal senoidal dado por $u(k) = 3{,}0\,\mathrm{sen}(2\,\pi\,f\,k) + 3{,}0$, utilizando-se a equação A.5, apêndice A, onde k é o instante de amostragem e f é a freqüência do sinal; o conjunto contém 60 instantes e o melhor resultado foi obtido com $f = 1/50$. O conjunto de teste inicial contém 60 instantes, onde 30 instantes formam uma rampa patamar, com $u(k)$ variando de 0 até 6 volts, e os outros 30 instantes formam dois degraus, $u(k)$ iguais a 5 e 2 volts; para isso utilizou-se a equação A.5.

Foi estabelecido como valor limite de erro de controle o valor de 1%, ou seja, se o erro relativo percentual E_C, calculado em um determinado instante k, estiver acima de 1%, uma nova rede controladora deverá ser criada.

A resposta a um degrau de 50 volts e a resposta a quatro valores diferentes de referência podem ser vistas, respectivamente, nas figuras 5.20 e 5.21.

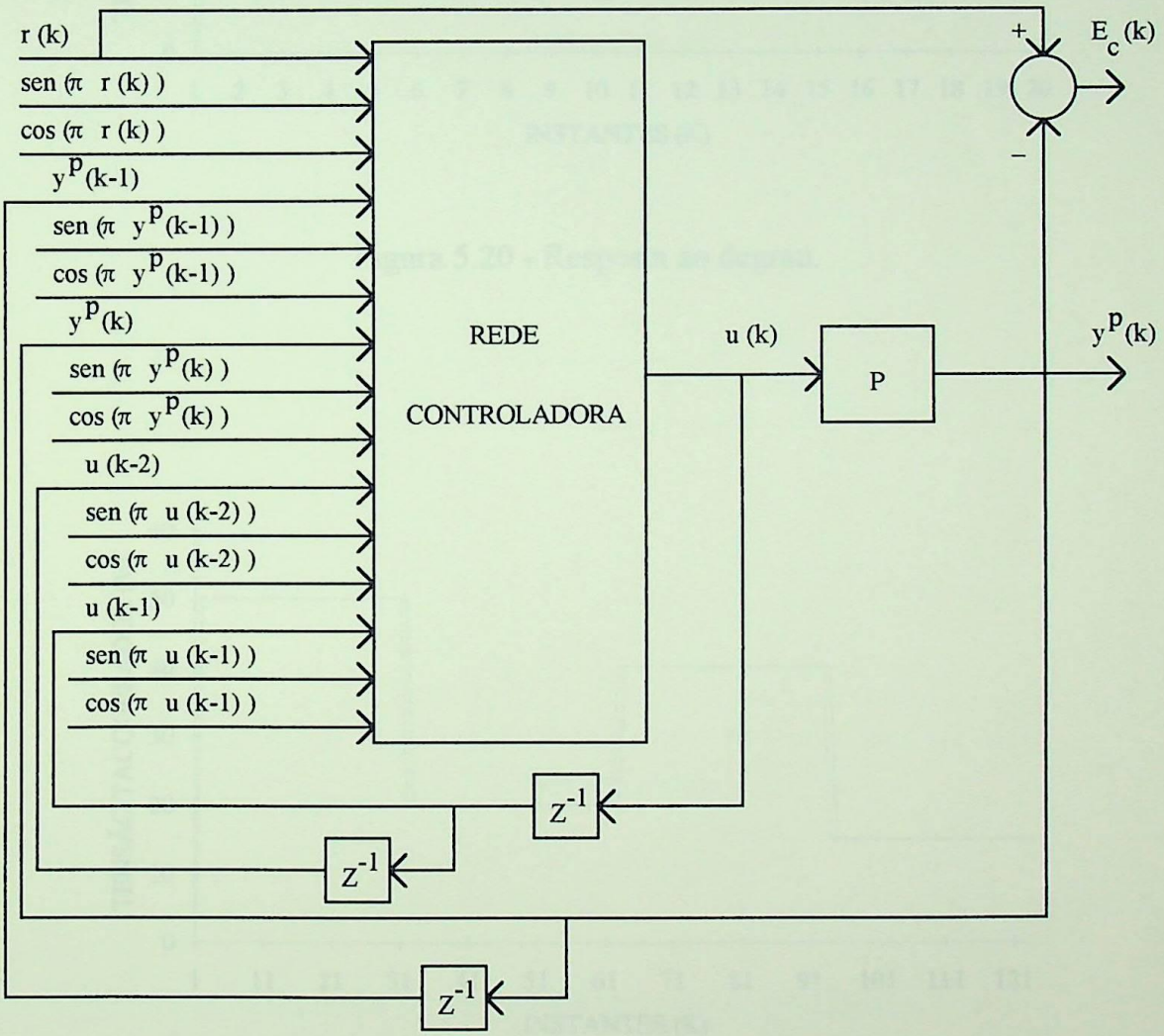


Figura 5.19 - Esquema do sistema de controle

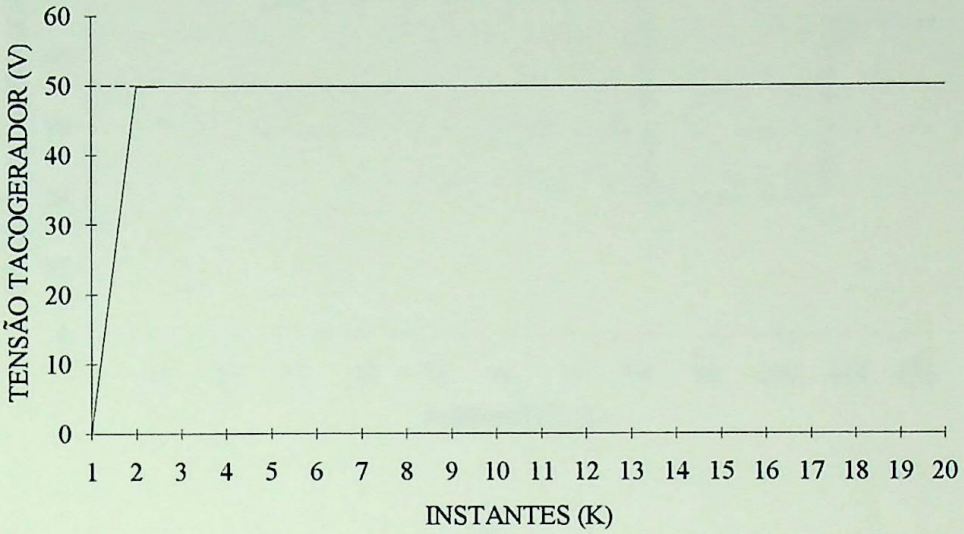


Figura 5.20 - Resposta ao degrau.

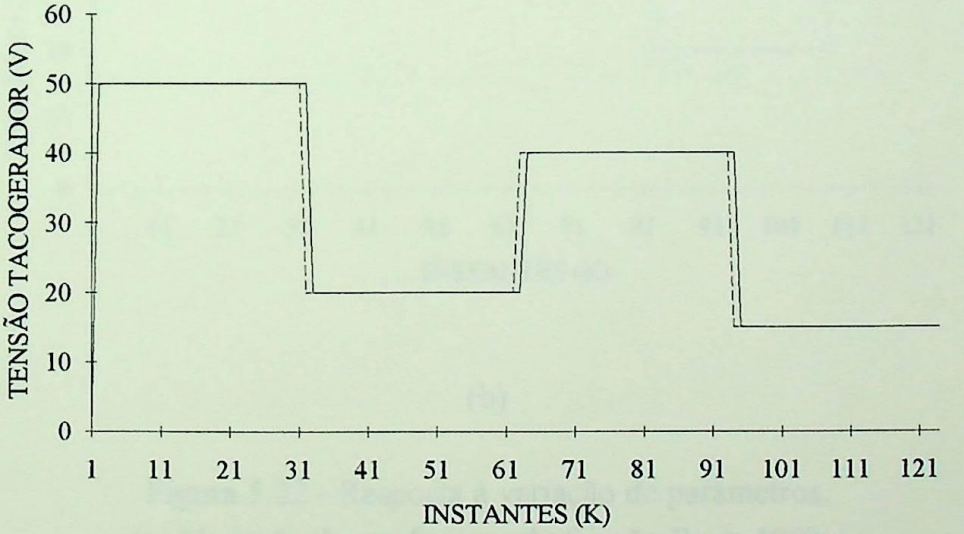
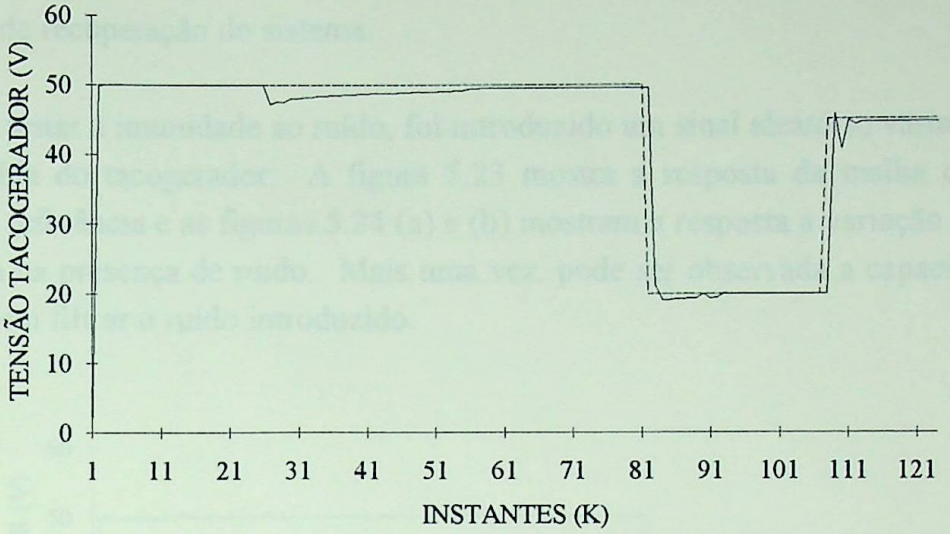
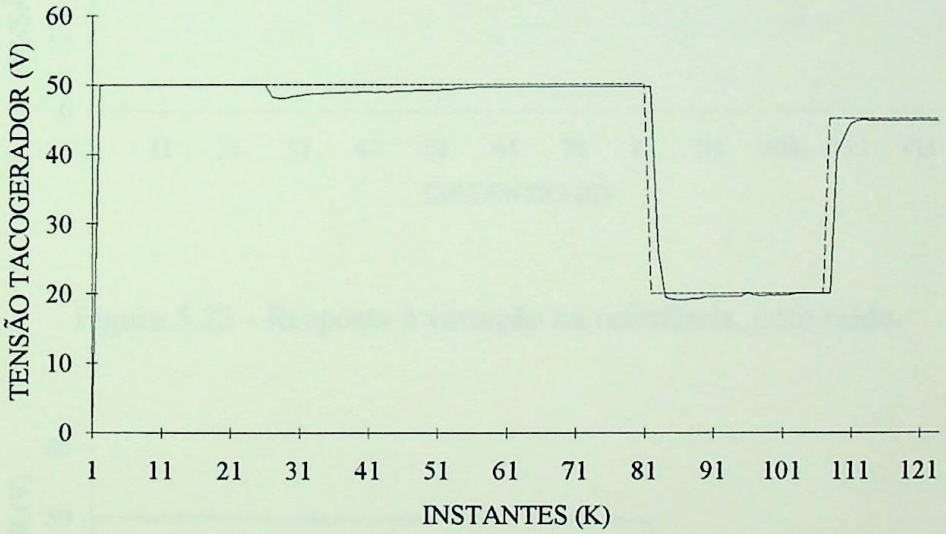


Figura 5.21 - Resposta à variação da referência.



(a)



(b)

Figura 5.22 - Resposta à variação de parâmetros.
(a) Variação do coeficiente de fricção B em 100%;
(b) Variação da resistência de armadura Ra em 80%.

Para verificar como o sistema reage a uma variação de parâmetro e referência, foi simulada uma variação no coeficiente de fricção e uma variação na resistência de armadura

do motor, os resultados podem ser vistos nas figuras 5.22 (a) e (b), onde nota-se uma boa velocidade de recuperação do sistema.

Para testar a imunidade ao ruído, foi introduzido um sinal aleatório variando de -4 a 4 volts na saída do tacogerador. A figura 5.23 mostra a resposta da malha de controle à variação na referência e as figuras 5.24 (a) e (b) mostram a resposta à variação de parâmetro e referência na presença de ruído. Mais uma vez, pode ser observada a capacidade da rede polinomial em filtrar o ruído introduzido.

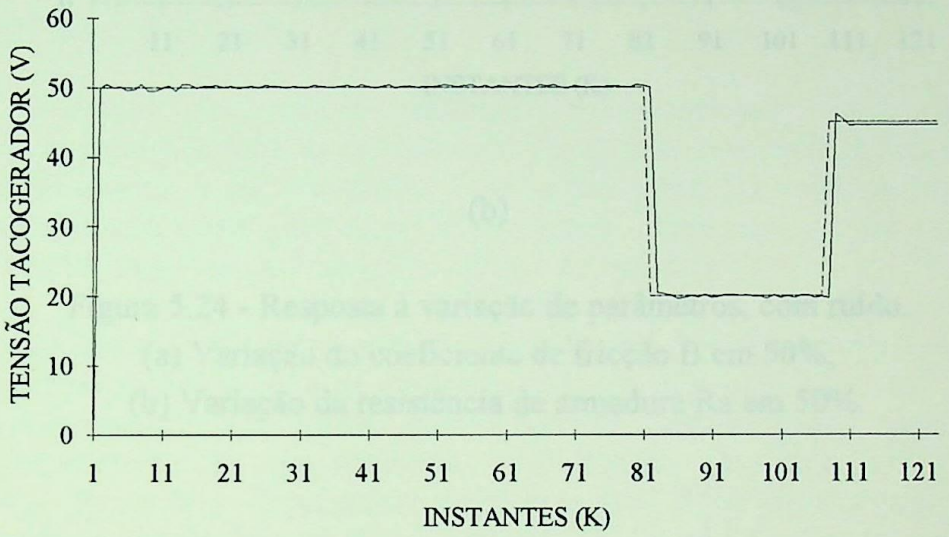
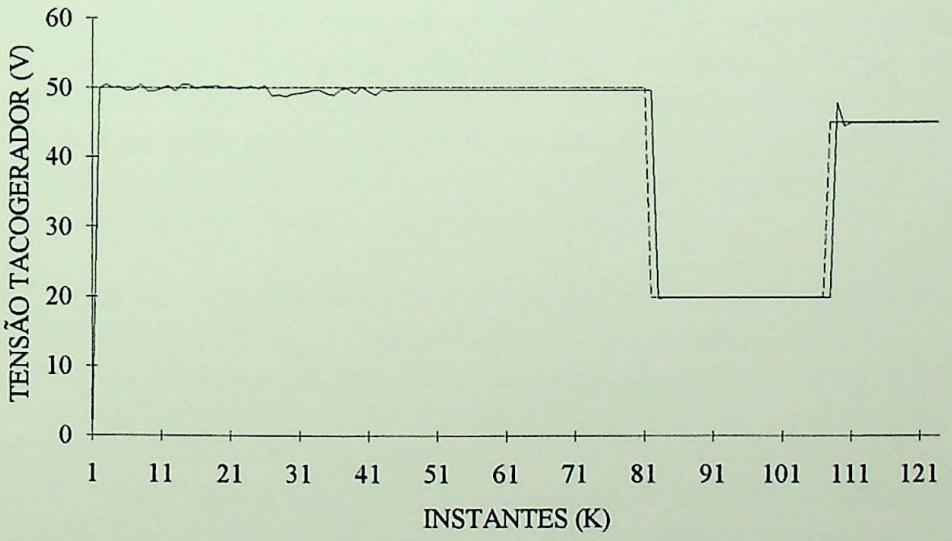
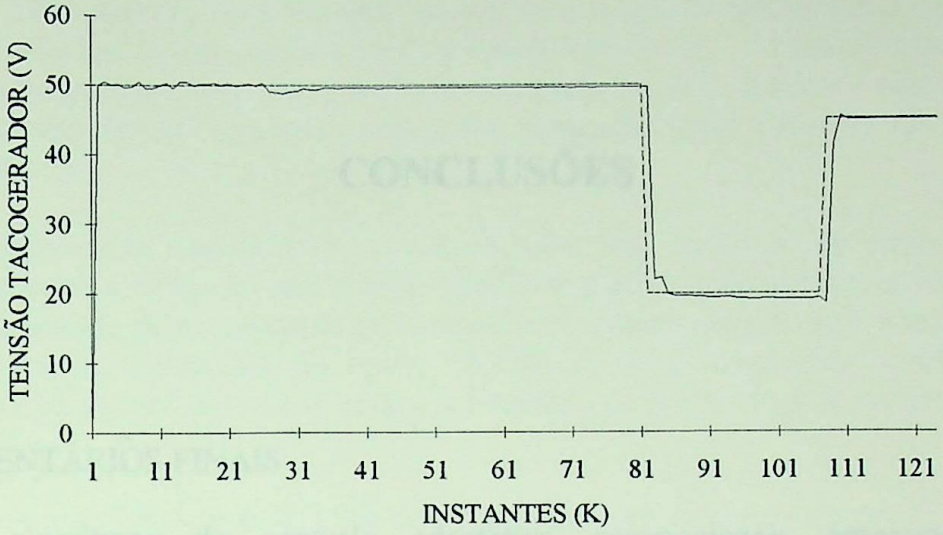


Figura 5.23 - Resposta à variação na referência, com ruído.



(a)



(b)

Figura 5.24 - Resposta à variação de parâmetros, com ruído.
(a) Variação do coeficiente de fricção B em 50%;
(b) Variação da resistência de armadura Ra em 50%.

CAPÍTULO 6

CONCLUSÕES

6.1 - COMENTÁRIOS FINAIS

Os algoritmos de controle adaptativo convencionais apresentam algumas desvantagens, entre elas, a falta de flexibilidade e a forte dependência de um conhecimento prévio sobre o sistema a ser controlado. As técnicas baseadas em controle clássico e moderno não são suficientes para resolver problemas complexos, que envolvem sistemas não-lineares variantes no tempo com parâmetros desconhecidos.

As redes neurais artificiais, apesar de ainda não terem um embasamento teórico sólido e apresentarem dificuldades de implementação em relação a "hardware" e "software", estão sendo aplicadas em controle com relativo sucesso. Suas características de paralelismo intrínseco, capacidade de aprendizagem, mapeamento não-linear e capacidade de generalização, são fatores importantes no tratamento direto de sistemas não-lineares variantes no tempo com parâmetros desconhecidos. No entanto, é essencial que se tenha um bom conhecimento das técnicas convencionais de controle adaptativo, pois as estruturas de controle que utilizam redes se baseiam nas estruturas convencionais.

A rede perceptron multicamadas, treinada com o algoritmo retro-propagação de erro "back-propagation", é a arquitetura de rede neural mais utilizada em controle apesar de apresentar várias imperfeições, tais como, dificuldade no ajuste dos parâmetros de aprendizagem, arquitetura pré-especificada, convergência lenta e falhas no treinamento devido aos mínimos locais. Ainda não existe um método eficiente e seguro para o treinamento deste tipo de rede. Foi analisado o comportamento deste tipo de rede em um esquema de controle adaptativo indireto, onde são necessárias duas redes, que funcionam respectivamente como identificador e controlador do sistema.

A arquitetura de rede apresentada e proposta para resolver problemas de controle adaptativo possui algumas vantagens sobre a rede perceptron multicamadas, tais como, a sua arquitetura é automaticamente definida durante o treinamento, não apresenta problemas de mínimos locais, não há parâmetro de aprendizagem para ser ajustado, além da rápida convergência. Foi analisado o comportamento da rede proposta em um esquema de controle adaptativo direto, onde é usada apenas uma rede que funciona como controlador do sistema. Não há a possibilidade da rede proposta ser usada em controle adaptativo indireto, pois, o erro de saída não pode ser retro-propagado através da mesma.

O acionamento do motor de corrente contínua foi escolhido como sistema a ser controlado, pois o mesmo apresenta um comportamento não-linear; cabe aqui salientar que o uso de redes neurais já torna o sistema de controle não-linear. Foram simuladas condições de operação normais, ou seja, sem a variação dos parâmetros do sistema e sem a presença de ruído. Também foram simuladas condições anormais, com variação de parâmetros e presença de ruído.

Comparando os resultados das simulações das duas estruturas de controle adaptativo, direto e indireto, conclui-se que a rede proposta produziu os melhores resultados com relação à velocidade de recuperação do sistema sem comprometer a estabilidade, capacidade de adaptabilidade e imunidade ao ruído. A rede proposta apresentou problemas quando foram simuladas variações bruscas, muito acentuadas, de parâmetros do sistema na presença de ruído.

Para que aplicações práticas de redes neurais artificiais na solução de problemas de controle adaptativo em tempo-real sejam viáveis, ainda é necessário aprimorar os algoritmos de treinamento, deixando-os mais rápidos e robustos. Bem como desenvolver técnicas que possibilitem a implementação das redes em forma de circuitos integrados, explorando assim toda a capacidade de paralelismo e processamento das mesmas.

6.2 - PROPOSTA PARA FUTUROS TRABALHOS

Há a necessidade de se fazer um estudo teórico mais aprofundado sobre a rede proposta, a fim de se obter um algoritmo mais rápido. Com relação a segurança na operação, é essencial que se tenha um sistema de supervisão do sistema controlado, bem como do próprio algoritmo, capaz de detectar falhas e tomar decisões. Aprimorado o sistema de controle adaptativo, seria a vez de realizar simulações mais detalhadas, envolvendo não só o acionamento do motor de corrente contínua, mas também outros sistemas não-lineares com parâmetros pouco conhecidos.

Provada a eficiência e a segurança do sistema de controle através de simulações, seria a vez de implementar o mesmo em "hardware". Para isso é necessário que se use uma linguagem de programação mais adequada como a linguagem C em conjunto com algumas rotinas em "assembly", a fim de viabilizar o controle digital em tempo-real. Como se sabe, hoje o mercado oferece placas microprocessadas cada vez mais rápidas e baratas, o que implica em uma redução considerável no custo total do projeto do sistema de controle.

BIBLIOGRAFIA

[1] T. Fukuda and T. Shibata : "Theory and applications of neural networks for industrial control systems", IEEE Transactions on industrial electronics, Vol. 39, No. 6, pp. 472 - 489, December 1992.

[2] K. Narendra and K. Parthasaraty : "Identification and control of dynamical systems using neural networks", IEEE Transactions on neural networks, Vol. 1, No. 1, pp. 4-27, March 1990.

[3] K. J. Hunt, D. Sbarbaro, R. Zbikowski and P. J. Gawthrop : "Neural networks for control systems - a survey", Automatica, Vol. 28, No. 6, pp. 1083-1112, 1992.

[4] B. Soucek and the IRIS Group : Neural and Intelligent Systems Integration, John Wiley & Sons, Inc., Chapter 9, pp. 235-279, 1991.

[5] K. Narendra and S. Mukhopadhyay : "Intelligent control using neural networks", IEEE Control Systems, pp. 11-18, April 1992.

[6] T. Kohonen : "The self-organizing map", Proceedings of the IEEE, Vol. 78, No. 9, pp. 1464-1480, September 1990.

[7] W. T. Miller, F. H. Glanz and L. G. Kraft : "CMAC: An associative neural network alternative to backpropagation", Proceedings of the IEEE, Vol. 78, No. 10, pp. 1561-1567, October 1990.

[8] P. J. Werbos : "Backpropagation through time: what it does and how to do it", Proceedings of the IEEE, Vol. 78, No. 10, pp. 1550-1560, October 1990.

[9] L. -W. Chan and F. Fallside : "An adaptive training algorithm for backpropagation networks", Computer Speech and Language, Vol. 2, pp. 205-218, 1987.

[10] E. A. C. Saturno : Aplicação de Redes Neuronais em Controle de Processos, Dissertação (mestrado em ciências em engenharia elétrica), Escola Federal de Engenharia de Itajubá, Itajubá, 1992.

[11] C. R. Dorneles : Aplicação de Redes Neurais em Controle Adaptativo, Dissertação (mestrado em ciências em engenharia elétrica), Instituto Militar de Engenharia, Rio de Janeiro, 1993.

[12] K. M. Passino : "Bridging the gap between conventional and intelligent control", IEEE Control Systems, pp. 12-18, June 1993.

[13] S. J. Farlow : Self-Organizing Methods in Modeling, Marcel Dekker, Inc., 1984.

[14] Y. -H. Pao : Adaptive Pattern Recognition and Neural Networks, Addison-Wesley Publishing Company, Inc., Chapter 8, pp. 197-222, 1989.

[15] A. P. Alves da Silva, A. M. Leite da Silva, J. C. S. de Souza and M. B. do Coutto Filho : "State Forecasting Based on Artificial Neural Networks", 11th Power Systems Computation Conference, PSCC, Avignon, pp. 461-467, August 1993.

[16] M. F. Tenorio and W. -T. Lee : "Self-Organizing Network for Optimum Supervised Learning", IEEE Transactions on Neural Networks, Vol. 1, No. 1, March 1990.

[17] J. Theocharis and V. Petridis : "Neural network observer for induction motor control", IEEE Control Systems, pp. 26-37, April 1994.

[18] K. J. Äström and B. Wittenmark : Adaptive Control, Addison-Wesley Publishing Company, New York, 1989.

[19] W. T. Miller, R. S. Sutton, and P. J. Werbos, Eds. : Neural Networks for Control, Cambridge, MA: The Mit Press, 1990.

[20] Alexandre P. Alves da Silva, Paulo C. Nascimento & Germano Lambert Torres : "An Alternative Approach for Adaptive Real-Time Control Using a Nonparametric Neural Network", aceito para publicação nos anais do IEEE Industry Applications Society Annual Meeting, Orlando, USA, October, 1995.

[21] Alexandre P. Alves da Silva, Paulo C. Nascimento & Germano Lambert Torres : "A Nonparametric Neural Network for Adaptive Real-Time Control", aceito para publicação nos anais do IEEE International Conference on Systems, Man and Cybernetics, Vancouver, Canada, October, 1995.

[22] A. P. Alves da Silva : Pattern Analysis and Parallel Distributed Processing in Power System State Estimation, Ph. D. Thesis, University of Waterloo, Canada, 1991.

[23] N. Winer : Cybernetics, 2nd ed., John Wiley, 1961.

[24] D. O. Hebb : The Organization of Behavior, John Wiley, 1949.

[25] F. Rosenblatt : "The Perceptron : a probabilistic model for information storage and organization in the brain", Psychological Review, Vol. 65, pp. 386-408, 1958.

[26] M. Minsky and S. Papert : Perceptrons, MIT Press, 1969.

[27] P. J. Werbos : Beyond regression: New tools for prediction and analysis in the behavioral sciences, Ph. D. Thesis, Harvard University, Cambridge, MA, 1974.

[28] D. B. Parker : "Learning-logic: Casting the cortex of the brain in silicon", Technical Report TR-47, Center of Computation Research in Economics and Management Science, MIT, Cambridge, MA, 1985.

[29] D. E. Rumelhart, G. E. Hinton and R. J. Williams : "Learning internal representations by error propagation.", In Parallel Distributed Processing: Explorations in the Microstructure of Cognition (D. E. Rumelhart and J. L. Mc Clelland, eds.), Vol. 1, Chapter 8, MA: MIT Press, 1986.

[30] J. J. Hopfield : "Neural Networks and physical systems with emergent collective computational abilities", Proceedings of The National Academy of Sciences, Vol. 79, pp. 2554-2558, 1982.

[31] H. White : "Neural-Network Learning and Statistics", AI Expert, pp. 48-52, December 1989.

[32] H. Kargupta and R. E. Smith : "System Identification with Evolving Polynomial Networks", Proceedings of the fourth International Conference on Genetic Algorithms, University of California, San Diego, pp. 370-376, July 13-16, 1991.

[33] S. A. Harp, T. Samad and A. Guha : "Towards the Genetic Synthesis of Neural Networks", Proceedings of the third International Conference on Genetic Algorithms, George Mason University, pp. 360-369, June 4-7, 1989.

APÊNDICE A

A figura A.1 mostra o sistema utilizado na simulação, o mesmo é composto basicamente de uma ponte retificadora e um motor de corrente contínua. Este trabalho se ocupa apenas com a resposta do sistema diante de variações na tensão da armadura, portanto, perturbações no torque da carga não serão levadas em conta.

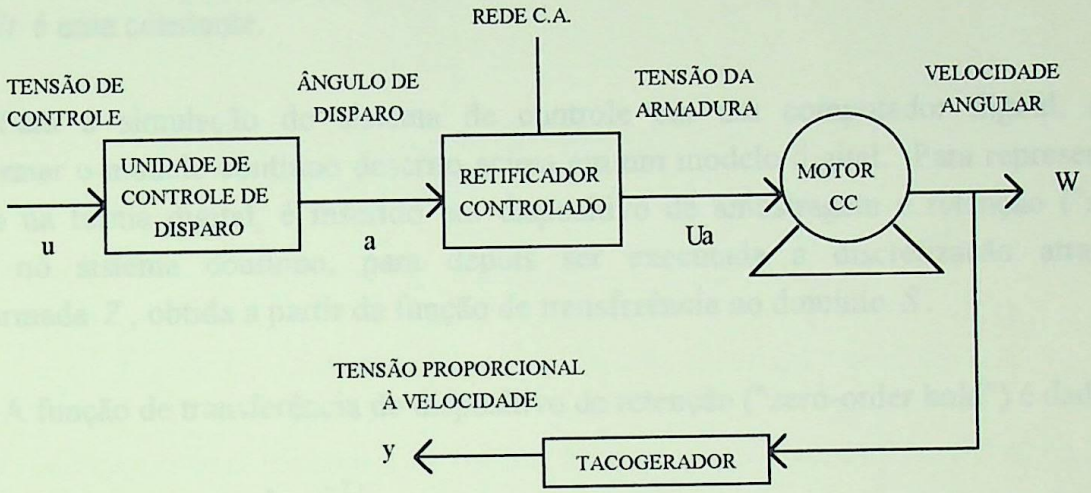


Figura A.1 - Diagrama de blocos representativo do sistema.

A função de transferência do conjunto unidade de controle de disparo e retificador controlado é dada por

$$\frac{Ua}{u}(s) = Kc \tag{A.1}$$

onde Kc é uma constante.

A função de transferência do motor é dada por

$$\frac{W}{Ua}(s) = \frac{Ka\phi}{Ka\phi^2 + Ra\,B\,(1+s\,\tau e)\,(1+s\,\tau m)} \tag{A.2}$$

a constante de torque $Kt\phi$ e a constante de força contra-eletromotriz $Ke\phi$ são numericamente iguais, assim, para simplificar a análise foi feito $Kt\phi = Ke\phi = Ka\phi$. Ra é a resistência da armadura, B é o coeficiente de fricção do conjunto motor-carga, τe é a

constante de tempo elétrica do motor, e τm é a constante de tempo mecânica do motor; $\tau e = La/Ra$, onde La é a indutância da armadura, e $\tau m = J/B$, onde J é a inércia do conjunto motor-carga.

A função de transferência do tacogerador é dada por

$$\frac{y}{W}(s) = Kt \tag{A.3}$$

onde Kt é uma constante.

Para a simulação do sistema de controle em um computador digital, deve-se transformar o modelo contínuo descrito acima em um modelo digital. Para representar um modelo na forma digital, é inserido um dispositivo de amostragem e retenção ("sample-hold") no sistema contínuo, para depois ser executada a discretização através da transformada Z, obtida a partir da função de transferência no domínio S.

A função de transferência do dispositivo de retenção ("zero-order hold") é dada por

$$G_{ZOH}(s) = \frac{1 - e^{-Ts}}{s} \tag{A.4}$$

onde T é o período de amostragem, que, para o caso específico de um sistema de controle de velocidade de um motor CC a escolha deve ser feita levando em consideração, basicamente, a constante de tempo mecânica do motor, que representa o fator mais crítico.

Neste trabalho foram adotadas as seguintes especificações:

Motor CC:	$1{,}2\,[KW]$	$110\,[V]$	$1500\,[rpm]$
	$Ra = 1{,}16\,[\Omega]$	$La = 60\,[mH]$	$\tau e = 0{,}052\,[s]$
	$J = 0{,}023\,[Kg\,m]$	$B = 0{,}018\,[Nms/rad]$	
	$\tau m = 1{,}28\,[s]$	$Ka\phi = 0{,}66\,[Vs/rad]$	
Tacogerador:	$Kt = 0{,}29\,[Vs/rad]$		

Conjunto controle de disparo e retificador: $Kc = 25$, ou seja, uma tensão de alimentação de $110\,[V]$ e uma tensão de controle máxima de $6\,[V]$.

O período de amostragem adotado foi de 200 milisegundos.

Vê-se que a rede neural M é colocada em paralelo com o sistema P a ser identificado e o erro entre a saída do sistema e a saída da rede é usado como sinal de treinamento para a rede, portanto, é um caso de aprendizagem supervisionada.

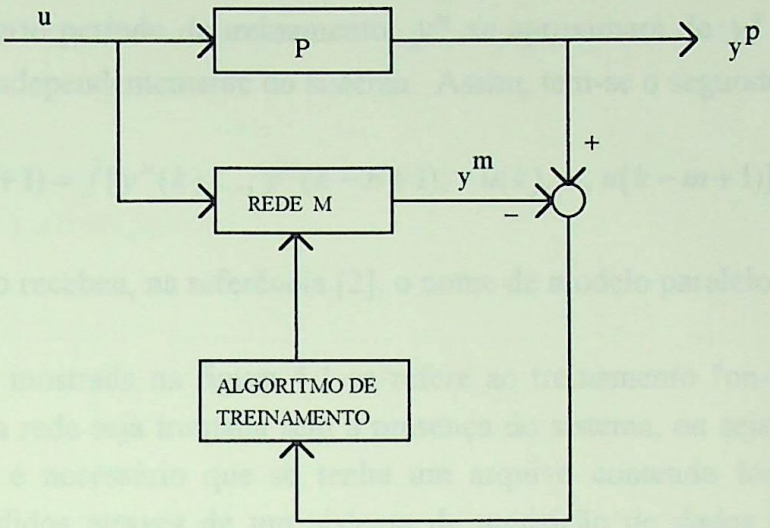


Figura 4.1 - Estrutura para identificação "forward modelling".

Devido à natureza dinâmica do sistema, a rede neural a ser utilizada deveria ser uma rede com realimentação, ou teria-se que introduzir um comportamento dinâmico dentro dos neurônios. Mas, por questão de simplicidade, amplia-se a entrada da rede estática com sinais correspondentes aos valores passados da entrada e saída [2, 3, 8, 10, 11, 17]. Neste capítulo, as figuras estão simplificadas, portanto, não mostram as linhas de atraso dos sinais utilizados na entrada das redes. Assume-se que o sistema é governado pela seguinte equação a diferenças não-linear discreta no tempo.

$$y^p(k+1) = f[y^p(k), \ldots, y^p(k-n+1); \quad u(k), \ldots, u(k-m+1)] \tag{4.1}$$

onde f é a função de mapeamento não-linear do sistema. Portanto, a saída y^p do sistema no instante de tempo $k+1$ depende de n valores anteriores de saída e de m valores anteriores de entrada u. Baseando-se na equação anterior, pode-se chegar a dois modelos. Para o primeiro modelo, as estruturas de entrada-saída da rede e do sistema serão idênticas. Portanto, a saída da rede será:

$$y^m(k+1) = \hat{f}[y^p(k), \ldots, y^p(k-n+1); \quad u(k), \ldots, u(k-m+1)] \tag{4.2}$$

onde $\hat{f}$ representa o mapeamento não-linear de entrada-saída da rede, ou seja, a aproximação de f. Este modelo recebeu, na referência [2], o nome de modelo série-paralelo.

Após um certo período de treinamento, y^m se aproximará de y^p e portanto, a rede poderá ser usada independentemente do sistema. Assim, tem-se o segundo modelo dado por

$$y^m(k+1) = \hat{f}[y^m(k), \ldots, y^m(k-n+1); \quad u(k), \ldots, u(k-m+1)] \tag{4.3}$$

Este modelo recebeu, na referência [2], o nome de modelo paralelo.

A estrutura mostrada na figura 4.1 se refere ao treinamento "on-line" da rede; mas nada impede que a rede seja treinada sem a presença do sistema, ou seja, treinamento "off-line". Para este, é necessário que se tenha um arquivo contendo todos os padrões de entrada-saída, medidos através de um sistema de aquisição de dados ligado ao sistema, utilizando um período de amostragem adequado. Pode-se formar os padrões da seguinte maneira; para um instante de amostragem genérico k, a saída padrão será $y^p(k+1)$ e as entradas padrões serão $y^p(k), \ldots, y^p(k-n+1); \quad u(k), \ldots, u(k-m+1)$.

4.2.2 - Modelagem inversa

Neste tipo de modelagem, o objetivo é gerar o modelo inverso do sistema dinâmico. Existem dois métodos para a identificação inversa; o primeiro é mostrado esquematicamente na figura 4.2 e se chama método direto.

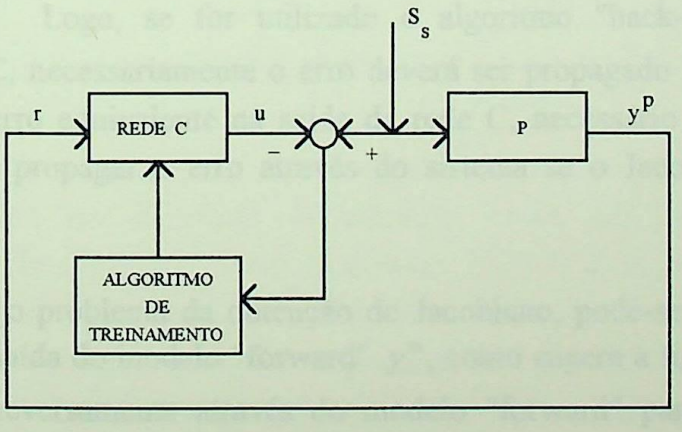


Figura 4.2 - Método direto.

S_s é um sinal sintético de treinamento. A saída do sistema P é usada como entrada para a rede C, a saída da rede é comparada com o sinal de treinamento e o erro resultante da comparação é usado para treinar a mesma. Nota-se que este método força a rede a representar a função inversa do sistema. No entanto, é mais conveniente usar a referência r como sinal de treinamento, pois a mesma corresponde ao comportamento desejado do sistema.

Para se eliminar o sinal sintético de treinamento, pode-se utilizar o método especializado, mostrado na figura 4.3.

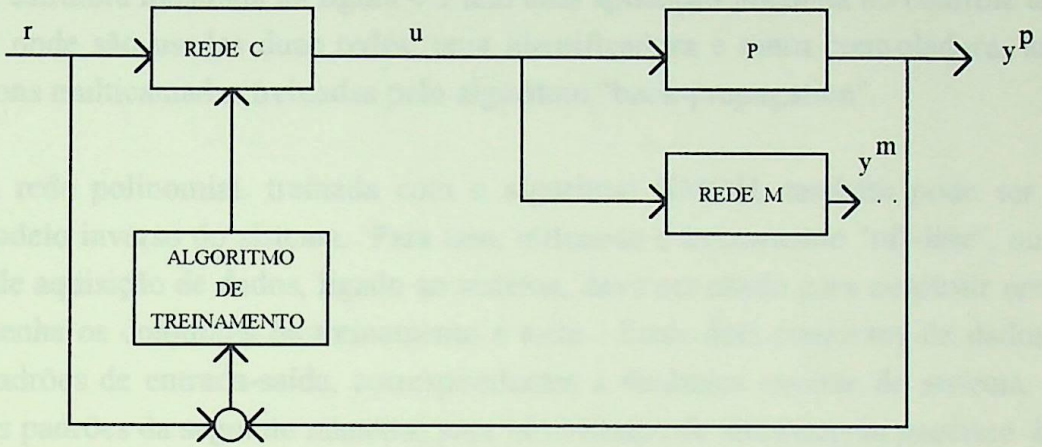


Figura 4.3 - Método especializado.

Neste método, a rede C, que representa o modelo inverso do sistema, precede o sistema e recebe como entrada um sinal externo de treinamento, r, o qual cobre todo o espaço operacional de saída do sistema controlado. O sinal de erro, usado no algoritmo de treinamento "back-propagation", é igual à diferença entre o sinal de treinamento r e a saída y^p do sistema real. Logo, se for utilizado o algoritmo "back-propagation" para o treinamento da rede C, necessariamente o erro deverá ser propagado através do sistema P. Com isso, tem-se o erro equivalente na saída da rede C, necessário ao ajuste dos pesos. Porém, só é possível propagar o erro através do sistema se o Jacobiano do mesmo for conhecido [1, 11].

Para contornar o problema da obtenção do Jacobiano, pode-se substituir a saída do sistema real y^p pela saída do modelo "forward" y^m, como sugere a figura 4.3. Neste caso, o erro é propagado reversamente através do modelo "forward" para o modelo inverso; durante este procedimento, somente os pesos da rede C, que representa o modelo inverso, serão ajustados.

A relação entre entrada-saída para a rede do modelo inverso, durante o treinamento, tem a seguinte forma

$$u(k) = \widehat{f^{-1}}[y^m(k), ..., y^m(k-n+1); r(k), \quad u(k-1), ..., u(k-m+1)] \qquad (4.4)$$

o valor futuro de y^m, ou seja, $y^m(k+1)$, na equação 4.3, foi substituído por $r(k)$, que é a referência no instante k.

A estrutura mostrada na figura 4.3 tem uma aplicação imediata no controle adaptativo indireto, onde são usadas duas redes, uma identificadora e outra controladora, ambas são perceptrons multicamadas treinadas pelo algoritmo "back-propagation".

A rede polinomial, treinada com o algoritmo GMDH, também pode ser utilizada como modelo inverso do sistema. Para isso, utiliza-se o treinamento "off-line", ou seja, um sistema de aquisição de dados, ligado ao sistema, deve ser usado para construir um arquivo, que contenha os conjuntos de treinamento e teste. Estes dois conjuntos de dados deverão conter padrões de entrada-saída, correspondentes à dinâmica inversa do sistema. Pode-se formar os padrões da seguinte maneira; para um instante de amostragem genérico k, a saída padrão será $u(k)$ e as entradas padrões serão $y^p(k+1)$, $y^p(k)$, ..., $y^p(k-n+1)$; $u(k-1)$, $u(k-2)$, ..., $u(k-m+1)$. A rede, após o treinamento, pode ser usada no controle adaptativo direto; para que isso seja possível, é proposto nesta dissertação um método de ajuste direto dos coeficientes dos neurônios, que será explicado logo mais adiante.

4.3 - ESTRUTURAS DE CONTROLE DE SISTEMAS

Pelo que foi visto anteriormente, pode-se notar que os modelos "forward" e inverso têm utilização imediata em estruturas de controle, como por exemplo, em controle adaptativo e controle inteligente. Serão mostradas adiante duas estruturas de controle adaptativo, baseadas nos dois tipos de modelagem analisados. Antes disso, pode-se citar duas aplicações diretas das redes em controle; uma delas é o controle supervisionado, onde uma rede, baseada na estrutura de identificação "forward modelling", é treinada com o objetivo de imitar habilidades humanas, úteis quando a modelagem do sistema é impossível.

A outra aplicação é chamada de controle direto, onde é utilizada uma rede representando o modelo inverso do sistema e a mesma age diretamente como controlador. As figuras 4.1 e 4.3 dão uma visão dessas duas aplicações diretas.

4.3.1 - Controle adaptativo direto

A figura 4.4 mostra o esquema simplificado do controle adaptativo direto utilizando rede neural.

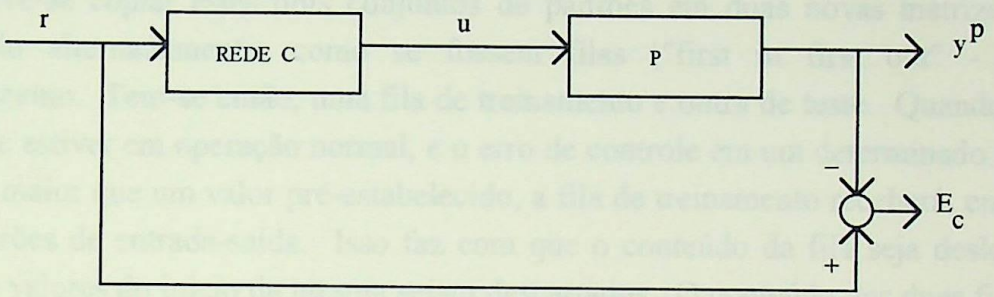


Figura 4.4 - Controle adaptativo direto.

A rede C atua como controlador adaptativo do sistema P, sendo este, variante no tempo. Portanto, o controlador terá que se auto-ajustar quando ocorrer variação nos parâmetros físicos do sistema, pois, tal variação provocará uma mudança no comportamento do mesmo. Pode-se detectar tal mudança calculando o erro de controle E_c, a cada instante de tempo.

Baseando-se no que foi explicado no item 4.2.2, conclui-se que é impraticável o uso da rede do tipo perceptron multicamadas, treinada pelo "back-propagation", em controle adaptativo direto [2]. Pois, não se pode retro-propagar o erro E_C através do sistema P.

Após analisar o algoritmo GMDH e a rede funcional ("functional-link net"), chegou-se à conclusão de que era possível desenvolver um método de treinamento adaptativo para a rede polinomial, e portanto, utilizar este tipo de rede em controle adaptativo direto.

Baseado no modelo expansão funcional, utilizado na "functional-link net", expande-se o número de entradas da rede polinomial utilizando-se as funções ortogonais seno e co-seno. Por exemplo, uma rede controladora que normalmente utilizaria como entradas os valores $r(k)$, $y^p(k)$, $y^p(k-1)$, ..., $y^p(k-n+1)$, $u(k-1)$, $u(k-2)$, ..., $u(k-m+1)$;

com a expansão funcional, as entradas ficam da seguinte forma: $r(k)$, $\text{sen}(\pi\, r(k))$, $\cos(\pi\, r(k))$, $y^p(k)$, $\text{sen}(\pi\, y^p(k))$, $\cos(\pi\, y^p(k))$, ..., $y^p(k-n+1)$, $\text{sen}(\pi\, y^p(k-n+1))$, $\cos(\pi\, y^p(k-n+1))$, $u(k-1)$, $\text{sen}(\pi\, u(k-1))$, $\cos(\pi\, u(k-1))$, ..., $u(k-m+1)$, $\text{sen}(\pi\, u(k-m+1))$, $\cos(\pi\, u(k-m+1))$. Cada variável, y^p e u, utilizadas nas funções seno e co-seno, devem ser normalizadas entre 0 e 1; com isso, as funções trabalham com variáveis entre 0 e π. Deve-se salientar que, nenhuma informação nova é adicionada à rede, somente expande-se a dimensão do espaço de representação dos padrões de entrada.

A rede controladora inicial deve ser treinada utilizando-se padrões de entrada-saída, divididos em dois conjuntos, um de treinamento e outro de teste. Antes de criar a rede inicial, deve-se copiar estes dois conjuntos de padrões em duas novas matrizes, e estas, funcionarão alternadamente, como se fossem filas ("first in first out" - FIFO) de armazenamento. Tem-se então, uma fila de treinamento e outra de teste. Quando o sistema de controle estiver em operação normal, e o erro de controle em um determinado instante de tempo for maior que um valor pré-estabelecido, a fila de treinamento receberá, em seu final, novos padrões de entrada-saída. Isso faz com que o conteúdo da fila seja deslocado para frente e os valores do início da mesma sejam descartados. O conteúdo das duas filas é então copiado nas matrizes X e Y, utilizadas pelo GMDH, e uma nova rede controladora é criada. Se, em outro instante de tempo o erro persistir, o processo descrito é repetido, porém, alterando-se agora, a fila de teste. Esta alternância na atualização das filas ocorre toda vez que uma nova rede tenha que ser criada. Com esse procedimento, a rede controladora aprenderá a nova dinâmica inversa do sistema controlado, quando for necessário.

Para ilustrar o funcionamento do esquema de controle, em regime de operação normal, supõe-se que a rede controladora inicial foi treinada e tem como entradas as variáveis $r(k)$, $y^p(k)$, $u(k-1)$, $\text{sen}(\pi\, r(k))$, $\cos(\pi\, r(k))$, $\text{sen}(\pi\, y^p(k))$, $\cos(\pi\, y^p(k))$, $\text{sen}(\pi\, u(k-1))$ e $\cos(\pi\, u(k-1))$; a saída é a variável $u(k)$. Onde k é um instante de amostragem qualquer.

A rede controladora recebe, no instante k, o sinal de referência $r(k)$ dado pelo operador e gera na saída o sinal de controle adequado $u(k)$, de acordo com o padrão constituído na sua entrada. O sinal de controle alimentará o sistema, que por sua vez, irá gerar a saída $y^p(k+1)$ e esta deverá acompanhar sempre a referência aplicada. Ainda no instante k, é calculado o erro de controle relativo percentual dado por $\left|(r(k)-y^p(k))/r(k)\right|\cdot 100\%$. Se o mesmo for maior que um valor pré-estabelecido, uma nova rede controladora deve ser criada para que o sistema funcione adequadamente. Para isso, todo o conteúdo da fila de treinamento será deslocado para frente e o conteúdo do

início da mesma é descartado; em seu final são armazenados os novos padrões de entrada-saída, dados por $y^p(k)$, $y^p(k-1)$, $u(k-2)$, $\text{sen}(\pi\, y^p(k))$, $\cos(\pi\, y^p(k))$, $\text{sen}(\pi\, y^p(k-1))$, $\cos(\pi\, y^p(k-1))$, $\text{sen}(\pi\, u(k-2))$, $\cos(\pi\, u(k-2))$ e $u(k-1)$ respectivamente, que correspondem a uma nova dinâmica inversa do sistema. As matrizes X e Y, utilizadas pelo GMDH, receberão os conteúdos das filas, de treinamento e teste, e uma nova rede é então criada. Toda esta seqüência deve ser executada dentro do intervalo de amostragem, ou seja, no instante $(k+1)$ já se deve ter disponível uma nova rede controladora, pronta para receber os novos valores de entrada: $r(k+1)$, $y^p(k+1)$, $u(k)$, $\text{sen}(\pi\, r(k+1))$, $\cos(\pi\, r(k+1))$, $\text{sen}(\pi\, y^p(k+1))$, $\cos(\pi\, y^p(k+1))$, $\text{sen}(\pi\, u(k))$ e $\cos(\pi\, u(k))$.

Para o instante $(k+1)$, a seqüência descrita acima é repetida. Se o erro de controle ainda for maior que o valor limite, deve-se lembrar da alternância na atualização das filas. Portanto, a fila de teste deverá agora ser deslocada para frente e em seu final serão armazenados novos dados.

4.3.2 - Controle adaptativo indireto

A figura 4.5 mostra o esquema simplificado do controle adaptativo indireto utilizando redes neurais.

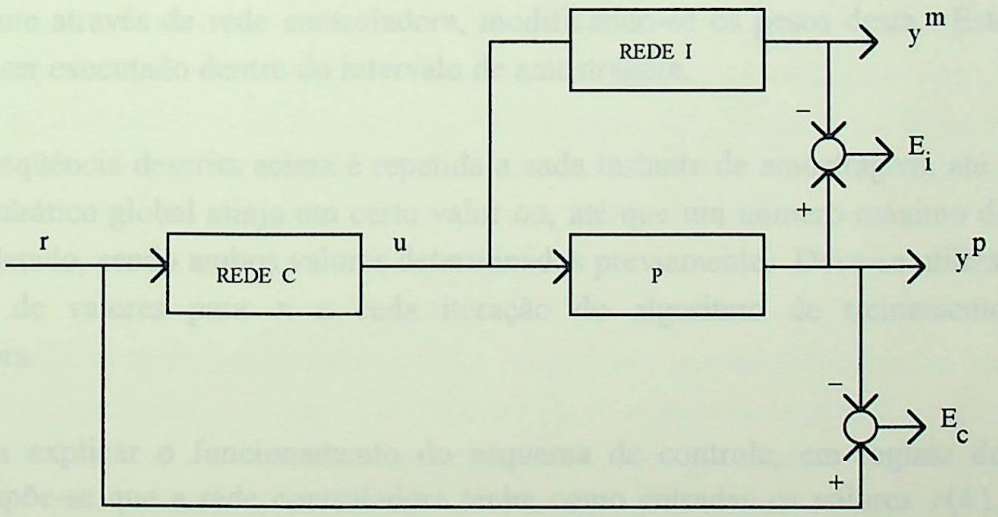


Figura 4.5 - Controle adaptativo indireto.

As duas redes, C e I, atuam como controlador e identificador respectivamente, e o sistema P é variante no tempo. O controlador terá que ser reajustado quando ocorrer

variação nos parâmetros físicos do sistema P, pois, tal variação provocará uma mudança no comportamento do mesmo. Pode-se detectar tal mudança calculando-se o erro de controle E_C, a cada instante de tempo.

As redes são perceptrons multicamadas, treinadas pelo algoritmo retro-propagação de erro "back-propagation". Para dar a partida no sistema, é preciso que se tenha as redes iniciais já treinadas. A primeira rede a ser treinada é a rede identificadora, para isso, foi utilizado neste trabalho o treinamento "off-line". Complementando o que já foi dito no item 4.2.1, os padrões de entrada-saída são apresentados ao algoritmo de forma cíclica e contínua, até que o erro médio quadrático global atinja um certo valor ou, até que um número máximo de iterações seja completado; ambos valores são determinados previamente.

O método utilizado para o treinamento inicial da rede controladora foi explicado no item 4.2.2; para complementar a explicação, o ajuste da rede obedece à seguinte seqüência: No instante de amostragem k, supõe-se que a rede controladora receba como entradas os valores $r(k)$, $y^m(k)$ e $u(k-1)$, e produza na saída o valor $u(k)$. A rede identificadora então, recebe como entradas os valores $u(k)$, $u(k-1)$, $y^m(k)$ e $y^m(k-1)$, produzindo em sua saída o valor $y^m(k+1)$. Ainda no instante k, é calculado o erro $r(k)-y^m(k)$ e este é então propagado reversamente através da rede de identificação, porém, sem modificar os pesos desta. Considera-se o erro na saída do neurônio na camada de entrada da rede de identificação, que recebe o sinal de controle, como sendo o erro equivalente na saída do neurônio da camada de saída da rede controladora. O erro equivalente é propagado reversamente através da rede controladora, modificando-se os pesos desta. Este processo todo deve ser executado dentro do intervalo de amostragem.

A seqüência descrita acima é repetida a cada instante de amostragem, até que o erro médio quadrático global atinja um certo valor ou, até que um número máximo de iterações seja completado, sendo ambos valores determinados previamente. Deve-se utilizar a mesma seqüência de valores para r a cada iteração do algoritmo de treinamento da rede controladora.

Para explicar o funcionamento do esquema de controle, em regime de operação normal, supõe-se que a rede controladora tenha como entradas os valores $r(k)$, $y^p(k)$ e $u(k-1)$, e produza na saída o valor $u(k)$. A rede identificadora tem como entradas os valores $u(k)$, $u(k-1)$, $y^p(k)$ e $y^p(k-1)$, produzindo em sua saída o valor $y^m(k+1)$.

A rede controladora recebe, no instante k, o sinal de referência $r(k)$ dado pelo operador, e gera na saída o sinal de controle adequado $u(k)$, de acordo com o padrão

constituído na sua entrada. O sinal de controle alimentará o sistema P e a rede identificadora, gerando respectivamente em suas saídas, os valores $y^p(k+1)$ e $y^m(k+1)$. Estes valores deverão acompanhar sempre a referência aplicada. Ainda no instante k, é calculado o erro de controle $E_C(k)$; se o mesmo for maior que um valor pré-estabelecido, a rede controladora deve ser retreinada para que o sistema P funcione adequadamente. A diferença entre $r(k)$ e $y^p(k)$ pode ser provocada por uma variação nos parâmetros físicos do sistema P ou, por influências do meio. Independente da causa da variação, não basta apenas retreinar a rede controladora, deve-se retreinar primeiro a rede identificadora, a fim de que esta continue representando o comportamento do sistema. Portanto, o erro de identificação $E_i(k)$, definido pelos sinais $y^p(k)$ e $y^m(k)$, deve ser calculado e retro-propagado para o reajuste dos pesos. A seguir, o erro de controle $E_C(k)$ é retro-propagado através da rede identificadora para a obtenção do erro equivalente, sem modificar os pesos desta; este erro então é retro-propagado através da rede controladora para o reajuste dos pesos. Toda esta seqüência deve ser executada dentro do intervalo de amostragem e repetida a cada instante, ao longo da operação normal.

4.3.3 - Controle inteligente

Pode-se dizer que as duas estruturas de controle adaptativo utilizando redes neurais, analisadas neste trabalho, exibem um certo grau de autonomia. Porém, ainda falta muito para se atingir um alto grau de autonomia ou "inteligência". O controlador adaptativo não tem, por exemplo, memória e portanto, não se lembra dos parâmetros mais adequados utilizados por ele, correspondentes às diferentes configurações assumidas pelo sistema ao longo do funcionamento. Além disso, não foi implementado um sistema de diagnóstico de falhas e tomada de decisões; com isso conclui-se que existe uma forte dependência do fator humano. Na referência [5] é proposto um sistema de controle inteligente, capaz de detectar falhas e tomar decisões, utilizando apenas redes neurais.

Nos últimos anos, pesquisadores estão propondo o uso de redes neurais em conjunto com sistemas especialistas e conjuntos difusos [1]. Um sistema especialista, por exemplo, seria capaz de produzir uma estratégia de controle, fornecendo a estrutura e os pesos iniciais para a rede controladora, baseando-se no conhecimento armazenado, em regras, nas condições atuais do sistema controlado e nas condições do ambiente. Em se tratando de sistemas baseados em conhecimento, como é o caso do sistema especialista, muitas vezes o conhecimento está associado a termos difusos e incertezas, porém regras convencionais como "if-then" não conseguem modelar tal conhecimento. Para resolver tal problema, conjuntos difusos podem ser utilizados.

Redes neurais e conjuntos difusos trabalhando em conjunto também vem sendo proposto. Por exemplo, a taxa de aprendizagem utilizada na rede pode ser determinada através de regras difusas, melhorando-se assim, a convergência do treinamento.

Cabe agora ressaltar que ainda existem muitas dificuldades a serem vencidas em relação a "hardware" e "software" [4], e muitas questões teóricas, principalmente sobre redes neurais, permanecem sem resposta.

CAPÍTULO 5

RESULTADOS DAS SIMULAÇÕES

5.1 - INTRODUÇÃO

Este capítulo tem como objetivo mostrar os resultados das simulações digitais do controle de velocidade de um motor de corrente contínua. O sistema a ser controlado é constituído de uma unidade de controle de disparo de tiristores, de um retificador controlado, do motor cc e de um tacogerador que irá gerar uma tensão proporcional à velocidade angular do motor. Este sistema pode apresentar um comportamento altamente não-linear, dificultando assim, a modelagem correta do mesmo e o projeto de um controlador adaptativo convencional. Além disso, os parâmetros físicos do sistema poderão variar inesperadamente devido às variações da temperatura ambiente, contaminação do tacogerador por ruído, desgaste, etc.

Baseado no que foi mostrado em capítulos anteriores, as redes neurais artificiais possuem características suficientes para justificar o seu uso em sistemas de controle adaptativo. Foram simuladas as estruturas de controle adaptativo indireto e direto utilizando-se, respectivamente, redes perceptrons multicamadas e rede polinomial. O objetivo das simulações é comparar a capacidade de adaptabilidade, a imunidade ao ruído, a velocidade de recuperação e a estabilidade do controle nas duas estruturas utilizadas, e conseqüentemente, verificar qual tipo de rede é mais adequada. Os diagramas de blocos neste capítulo estão detalhados, mostrando as linhas de atrasos dos sinais utilizados pelas redes.

Para avaliar o desempenho de cada estrutura de controle, foram utilizados sinais de referência em forma de degrau, variação de parâmetros físicos do motor e introdução de ruído no sinal de saída do tacogerador.

5.2 - SIMULAÇÃO DO CONTROLE ADAPTATIVO INDIRETO

Como já foi dito anteriormente, para a simulação do controle adaptativo indireto serão utilizadas duas redes perceptrons multicamadas, treinadas por uma variante do algoritmo "back-propagation".

O apêndice A contém toda especificação do sistema a ser controlado. Na simulação do controle adaptativo indireto será utilizado o modelo do sistema de primeira ordem dado pela equação A.6.

Os próximos itens mostrarão o resultado do treinamento das redes identificadora e controladora, e a simulação do funcionamento do sistema de controle em regime de operação normal. O algoritmo de treinamento "back-propagation" foi implementado baseado nas referências [9, 29] . No entanto, na tentativa de se implementar a idéia contida na referência [9], onde é proposto um algoritmo "back-propagation" adaptativo que utiliza valores variáveis para a taxa de aprendizagem e de momento, somente a adaptação dos valores da taxa de momento foi implementada com sucesso e portanto, para a taxa de aprendizagem foi utilizado um valor fixo. Mesmo assim, atingiu-se uma melhora significativa no treinamento em comparação ao algoritmo convencional contido em [29].

5.2.1 - Treinamento da rede identificadora

Para realizar o treinamento da rede é necessário definir a arquitetura da mesma, seus parâmetros de treinamento e o conjunto de padrões de entrada-saída devidamente normalizado, utilizado no treinamento "off-line". A definição da arquitetura da rede apresenta um grande problema, pois, a mesma deve possuir um número adequado de neurônios para uma modelagem correta, "forward" e inversa, do sistema. Se o número de neurônios for insuficiente, a rede não terá os graus de liberdade necessários, e, não importa o tipo de treinamento, o erro residual será alto. Com um número excessivo de neurônios, a rede terá graus de liberdade desnecessários, geralmente perdendo a capacidade de generalização. O conhecimento ou a estimação da ordem do sistema também é importante, pois, este dado é necessário para a determinação do número de atrasos nos pares entrada-saída do sistema que formam o vetor de entrada da rede. Se a ordem for subestimada, haverá um grande erro residual no final do treinamento e, se a ordem for superestimada, a rede perderá a capacidade de generalização [11].

Para gerar o conjunto de padrões de entrada-saída foi usada a equação A.6, que representa o modelo do sistema de primeira ordem, e uma seqüência de sinais de controle variando de 0 a 6 volts formando uma onda em forma de rampa de subida, patamar e rampa de descida. O sinal máximo de controle, 6 volts, corresponde a uma saída do tacogerador de 62,9 volts; portanto, pode-se utilizar as seguintes faixas para a normalização dos valores operados pela rede: {0,...,6} para a entrada e suas amostras anteriores, e {0,...,70} para a saída e suas amostras anteriores. O procedimento de treinamento foi explicado no capítulo 4. O melhor resultado foi obtido utilizando-se uma rede com 4 entradas e uma camada interna com 4 neurônios. A figura 5.1 mostra o esquema de treinamento "off-line" da rede identificadora.

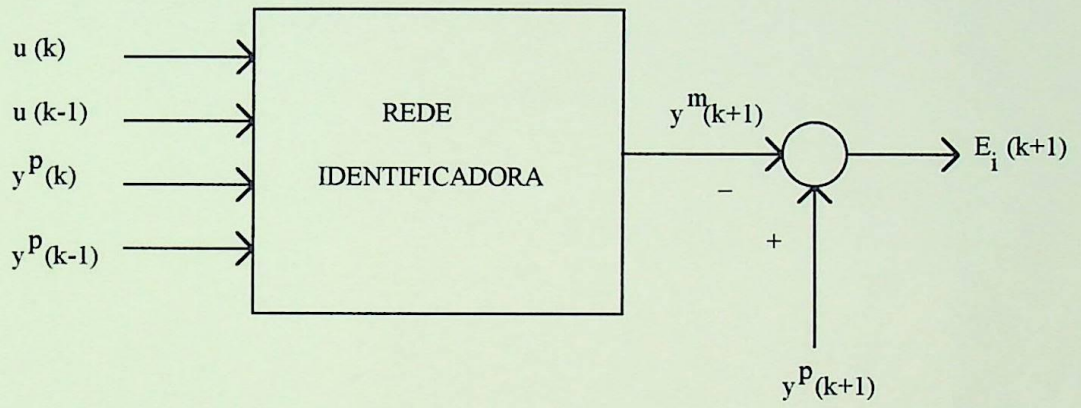


Figura 5.1 - Esquema de treinamento da rede identificadora.

Foi utilizada uma taxa de aprendizagem com valor fixo igual a 0,8 e uma taxa de momento com valor inicial igual a 0,5. Após várias simulações, concluiu-se que 15.000 iterações eram suficientes para se evitar o super ajustamento dos pesos da rede sendo obtido um erro médio quadrático igual a $3{,}062 \times 10^{-6}$. As figuras 5.2 e 5.3 mostram, respectivamente, um gráfico com a resposta da rede ao sinal de treinamento e um gráfico com a resposta da rede a um sinal de referência de teste formado por dois degraus, de 5 e 2 volts respectivamente. A linha tracejada representa a saída padrão e a linha contínua representa a saída da rede.

Após os devidos cálculos, pode-se chegar à seguinte equação a diferenças:

$$y(k+1) = 11{,}7770\,u(k) + 1{,}7097\,u(k-1) - 0{,}2684\,y(k) - 0{,}0183\,y(k-1) \quad \text{(A.5)}$$

obtendo-se assim, um sistema de segunda ordem. Como $\tau e << \tau m$, pode-se desprezar τe, obtendo-se a seguinte equação a diferenças:

$$y(k+1) = 0{,}0328\,y(k) + 10{,}1379\,u(k) \quad \text{(A.6)}$$

ou seja, um sistema de primeira ordem.