



**UNIVERSIDADE FEDERAL DA PARAÍBA  
CENTRO DE CIÊNCIAS E TECNOLOGIA  
COORDENAÇÃO DOS CURSOS DE  
PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA**

**JOSÉ HOMERO FEITOSA CAVALCANTI**

**CONTROLADORES NEURAIIS ADAPTATIVOS**

Tese apresentada à Coordenação de Pós-Graduação em Engenharia Elétrica - COPELE - da Universidade Federal da Paraíba - UFPB, como parte dos requisitos necessários à obtenção do grau de Doutor em Engenharia Elétrica.

**ÁREA DE CONCENTRAÇÃO  
PROCESSAMENTO DA INFORMAÇÃO**

**ORIENTADORES:**

**GURDIP SINGH DEEP  
ANTÔNIO MARCUS NOGUEIRA LIMA**

**CAMPINA GRANDE - PARAÍBA  
OUTUBRO - 1994**



C376c Cavalcanti, José Homero Feitosa.  
Controladores neurais adaptativos / José Homero Feitosa  
Cavalcanti. - Campina Grande, 1994.  
122 f.

Tese (Doutorado em Engenharia Elétrica) - Universidade  
Federal da Paraíba, Centro de Ciências e Tecnologia, 1994.  
"Orientação : Prof. Dr. Gurdip Singh Deep, Prof. Dr.  
Antonio Marcus Nogueira Lima".  
Referências.

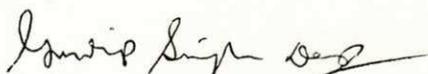
1. Redes Neurais - Controladores Neurais. 2.  
Inteligência Artificial. 3. Controladores Neurais  
Adaptativos. 4. Tese - Engenharia Elétrica. I. Deep, Gurdip  
Singh. II. Lima, Antônio Marcus Nogueira. III. Universidade  
Federal da Paraíba - Campina Grande (PB). IV. Título

CDU 004.8(043)

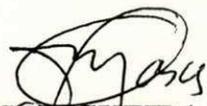
**CONTROLADORES NEURAIIS ADAPTATIVOS**

**JOSÉ HOMERO FEITOSA CAVALCANTI**

Tese Aprovada em 17.10.1994



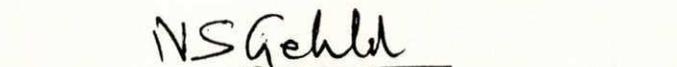
**GURDIP SINGH DEEP, Ph.D., UFPB**  
Orientador



**ANTONIO MARCUS NOGUEIRA LIMA, Dr., UFPB**  
Orientador



**EDILBERTO PEREIRA TEIXEIRA, Dr., UFU**  
Componente da Banca



**NARPAT SINGH GEHLOT, Dr.Engg., UFPB**  
Componente da Banca



**EDSON NASCIMENTO, Dr., UFMA**  
Componente da Banca

CAMPINA GRANDE - PB  
Outubro - 1994

## AGRADECIMENTOS

Aos orientadores, professores Gurdip Singh Deep e Antonio Marcus Nogueira Lima pela análise, persistência e incentivos demonstrados, que tornaram possível a realização deste trabalho.

Ao companheiro professor, José Sérgio da Rocha Neto (DEE-UFPB), pelas constantes discussões que ajudaram a nortear este trabalho.

Aos demais professores, técnicos e funcionários do DEE, pelo apoio e estímulo durante a realização deste trabalho.

O principal objetivo desta Tese é demonstrar que, para alguma classe de sistemas não lineares cujo modelo se conhece parcialmente, é possível empregar redes neurais artificiais multi-camadas (RNMC) para implementar uma estratégia de controle adaptativa, sem que seja necessário o treinamento prévio "off line" da rede.

Inicialmente, são apresentados resultados experimentais e resultados obtidos com simuladores, no controle da velocidade e do posicionamento de um motor CC. Os resultados experimentais foram obtidos com controladores convencionais, representados por controladores tipo PID e controladores adaptativos baseados em modelos de referência, e por controladores não convencionais, representado por controladores neurais. Os resultados obtidos com os simuladores e com a comparação entre os controladores convencionais e não convencionais, foram usados para o projeto da arquitetura da rede neural artificial do controlador.

Usando os principais controladores neurais descritos na literatura especializada, especificamente os controladores neurais direto, indireto e baseado em funções não lineares, foram feitos estudos experimentais e em tempo real com esses controladores. Usando o conhecimento do jacobiano do motor CC, mostrou-se que esses controladores podem se tornar controladores adaptativos. Mostrou-se também que para o motor CC, é possível transformar o controlador neural direto num controlador adaptativo. Usando-se o jacobiano da planta, desenvolveu-se o conceito de estado passivo que permite o treinamento "on line" da RNMC, sem o seu prévio treinamento "off line", com relativa segurança, evitando um treinamento a priori prolongado da RNMC. Esse conceito também possibilitou a sintonia fina do controlador em tempo real.

Algumas considerações sobre a arquitetura da RNMC foram verificadas usando o controlador neural adaptativo. Apresentou-se um método para se calcular o número ótimo dos neurônios na camada oculta da RNMC. Definiu-se o fator de adaptação para o controlador neural direto e se apresentou um método para determinar o seu valor ótimo. Para mostrar experimentalmente a capacidade de generalização do controlador neural adaptativo direto, associou-o a regras fuzzy e se implementou um sistema controlador neural adaptativo para posicionamento do braço de um pêndulo invertido acoplado ao eixo de um motor CC.

## SUMMARY

---

The principal objective of this Thesis is to demonstrate that for a class of non-linear systems with partially known models, it is possible to employ a multi-layer artificial neural network (MLANN), for an adaptative control strategy, without the need for off-line training of the neural network.

Initially, some experimental results, as well as simulation results, related with the position and speed control of a d.c. motor are presented. The experimental results were obtained both with the conventional controllers like the PID controller and reference model adaptive controller and the non-conventional controller like a neural network controller. Based on the simulations studies and comparisons of the performance of the conventional and neural controllers, the configuration of a neural network controller is proposed.

Using the principal neural controller strategies proposed in the literature, and more specifically the direct and indirect controllers and the ones based on non-linear functions, experimental studies were carried out for a d.c. motor drive system with these controllers. These investigations revealed that if the jacobian of a plant is known, it is possible to transform the direct neural controller into an adaptive neural controller. Based on the plant jacobian, the concept of a passive state for the plant under control is introduced. The use of this concept enables the on-line training of the multi-layer artificial neural network in real time with a fair degree of reliability and the necessity for a prolonged off-line training of the MLANN is thus avoided. This also affords a possibility for on-line fine-tuning of the neural controller.

Some architectural aspects and other characteristics of the MLANN, as an adaptive controller, have been verified. A method to determine the optimum number of neurons in the hidden layer is presented. The adaptation factor for a direct adaptive neural controller is defined and an experimental procedure to determine its value, is presented.

With a view to experimentally demonstrate the possibility of the generalization of the direct neural adaptive controller, concepts of fuzzy control are combined with those of the neural control to implement a position controller for a rigid inverted pendulum arm mounted on the d.c. motor shaft.

<b>INTRODUÇÃO - Controladores Neurais Adaptativos.</b>	<b>1</b>
<b>CAPÍTULO 1 - O Sistema de Acionamento do Motor CC.</b>	
1.1 - Introdução	7
1.2 - Características do Sistema Motor de Corrente Contínua	7
1.3 - Identificação da constante elétrica de tempo e de acionamento do motor CC	11
1.4 - Conclusão	15
<b>CAPÍTULO 2 - Controladores Convencionais.</b>	
2.1 - Introdução	17
2.2 - Controladores Proporcionais e Integrais (PI)	18
2.3 - Controladores Proporcionais e Derivativos (PD)	20
2.4 - Sistema Adaptativo com Modelo Referência (MRAS)	22
2.5 - Projeto de MRAS Usando a Abordagem do Gradiente	24
2.5.1 - Regra MIT	24
2.5.2 - Controle da Velocidade do Motor CC usando a Regra MIT	26
2.6 - Controle Adaptativo Usando funções de Lyapunov	29
2.6.1 - Projeto de MRAS Usando funções de Lyapunov	29
2.6.2 - Controle da Velocidade do Motor CC usando Funções de Lyapunov	30
2.7 - Conclusão	31
<b>CAPÍTULO 3 - Controladores Neurais Artificiais.</b>	
3.1 - Introdução	33

3.2 - Redes neurais artificiais	33
3.3 - Arquitetura da Redes Neural Multi-camadas	35
3.3.1 - Número de camadas da RNMC	36
3.3.2 - Número de neurônios na camada oculta da RNMC	36
3.3.3 - Modelos dos neurônios da RNMC	36
3.3.4 - Número de entradas e saídas da RNMC	37
3.3.5 - Sinais usados no treinamento da RNMC	37
3.3.6 - Número de iterações usadas no treinamento e as condições iniciais	38
3.4 - Treinamento "off line" da RNMC com a dinâmica da planta	38
3.5 - Análise do treinamento "off line" da RNMC com a dinâmica do motor CC	40
3.5 - Conclusão	42
<b>CAPÍTULO 4 - Controlador Neural Adaptativo.</b>	
4.1 - Introdução	43
4.2 - Controladores neurais usando a dinâmica inversa da planta	44
4.3 - Dados experimentais do motor CC	44
4.4 - Controlador neural direto inverso	46
4.5 - Treinamento "on line" da RNMC do controlador neural direto inverso	48
4.6 - Controlador neural adaptativo direto	51
4.7 - Controlador neural adaptativo indireto	52
4.8 - Controle neural adaptativo baseado em funções não lineares	56
4.9 - Controle neural adaptativo direto	61
4.10 - Conclusão	62
<b>CAPÍTULO 5 - Comparação entre os controladores neurais e convencionais.</b>	
5.1 - Introdução	65
5.2 - Comparação entre controladores convencionais e neurais	65
5.3 - Comparação entre os controladores neurais	69
<b>CAPÍTULO 6 - Controle neural adaptativo em tempo real.</b>	
6.1 - Introdução	73
6.2 - Treinamento em tempo real do controlador neural	74
6.3 - Cálculo do valor ótimo do fator de adaptação do controlador neural	78
6.4 - Cálculo do número ótimo de neurônios na camada oculta da RNMC	80
6.5 - Controle neural de um pêndulo invertido	83
6.5.1 - Controlador neural de posição	83
6.5.2 - Controle neural de posição do pêndulo	85
6.5.3 - Posicionamento do pêndulo invertido	86

	Sumário
6.5.4 - Posicionamento do pêndulo invertido instável na posição para cima	92
6.6 - Conclusão	94
<b>CAPÍTULO 7 - Conclusões e Perspectivas Futuras</b>	95
<b>REFERÊNCIAS BIBLIOGRÁFICAS</b>	99
<b>ANEXO A - Redes Neurais Multi-Camadas</b>	
A.1 - Introdução	107
<b>ANEXO B - Controladores Fuzzy</b>	
B.1 - Introdução	115
B.2 - Controlador fuzzy tradicional	116
B.3 - Controle fuzzy inverso	120

## LISTA DAS FIGURAS

**Capítulo 1.**

1.1 - Sistema de acionamento experimental do motor CC	8
1.2 - Geração do sinal PWM de tensão do chaveador	9
1.3 - Diagrama de blocos do sistema motor CC	9
1.4 - Curvas experimentais da velocidade angular, tensão e corrente de armadura	13

**Capítulo 2.**

2.1 - Diagrama de blocos do controlador PI de velocidade	18
2.2 - Resultados experimentais obtidos com o controlador PI	20
2.3 - Diagrama de blocos do controlador PD de posição	20
2.4 - Resultados experimentais obtidos com o controlador PD	22
2.5 - Controlador robusto de alto ganho	23
2.6 - Sistema adaptativo com modelo referência	23
2.7 - Adaptação dos parâmetro $t_0$ e $s_0$	28
2.8 - Resultados experimentais obtidos com o controlador MIT	28

**Capítulo 3.**

3.1 - Determinação da dinâmica de uma planta usando RNMC	35
3.2 - RNMC usada como EMULADOR	35
3.3 - RNMC usada como EMULADOR para o motor CC	38
3.4 - Treinamento da RNMC com excitação senoidal em $U(t)$	39
3.5 - Treinamento da RNMC com excitação aleatória em $U(t)$	40
3.6 - Índice de desempenho durante o treinamento da RNMC	42

**Capítulo 4.**

4.1 - Resultados experimentais do motor CC	45
4.2 - Fase de controle do controlador neural direto inverso	46
4.3 - Fase de treinamento "off line" do controlador neural direto inverso	47
4.4 - Resultados experimentais obtidos com o controlador neural direto inverso	48
4.5 - Configuração do controlado neural direto inverso	49

4.6 - Aprendizagem "on line" da dinâmica do motor CC	50
4.7 - Controlador neural adaptativo direto	52
4.8 - Esquema de treinamento das RNMC	53
4.9 - Controlador neural adaptativo indireto	54
4.10 - Resultados experimentais obtidos com o controlador neural indireto	55
4.11 - Controle neural adaptativo baseado em funções não lineares	56
4.12 - Arquitetura da RNMC usada no controlador de Chen	59
4.13 - Adaptação do controlador neural para sistemas não lineares	60
4.14 - Adaptação em tempo real usando o controlador neural direto	62
 <b>Capítulo 5.</b>	
5.1 - Superfície de controle gerada pelo controlador "fuzzy"	66
5.2 - Superfície de controle gerada pelo controlador neural	67
5.3a - Resultados experimentais usando controlador PI - excitação senoidal	68
5.3b - Resultados experimentais usando controlador PI - referência constante	68
5.4 - Adaptação do controlador MIT	69
 <b>Capítulo 6.</b>	
6.1 - Controlador neural adaptativo direto	74
6.2 - Curvas experimentais obtidas na partida do motor CC	76
6.3 - Treinamento "on line" da RNMC com diferentes valores de $\chi$	79
6.4 - Tempo de adaptação da RNMC versus fator de adaptação da RNMC	79
6.5 - Fase de Treinamento da RNMC	81
6.6a - Tempo de transição da RNMC versus número de neurônios - experimental	82
6.6b - Tempo de transição da RNMC versus número de neurônios - simulado	82
6.7 - Esquema geral do pêndulo simples	84
6.8 - Posicionamento e adaptação de um peso ao eixo do motor CC	85
6.9 - Controlador neural de posição do eixo do motor CC	86
6.10 - Círculo de torque do braço do pêndulo invertido	89
6.11 - Posições de referência do braço do pêndulo para treinamento da RNMC	90
6.12 - Movimento do braço do pêndulo próximo de $\theta = \pi$ rd	91
6.13 - Posicionamento em $\theta = 0$ do pêndulo invertido instável	93
6.14 - Posicionamento do braço do pêndulo invertido instável	93
 <b>Anexo A</b>	
A.1 - Rede neural multi-camadas	107

**Anexo B**

B.1 - Controlador fuzzy	117
B.2 - Função de pertinência de $E_f$	118
B.3 - Superfície de controle gerada pelo controlador fuzzy	119
B.4 - Resultados experimentais do controlador fuzzy	120
B.5 - Resultados experimentais do controlador fuzzy a nível móvel	121
B.5 - Resultados experimentais do controlador fuzzy adaptativo a nível móvel	122

TABELAS

---

**Capítulo 1.**

1.1 - Parâmetros do motor CC	10
------------------------------	----

**Anexo B.**

B.1 - Tabela das funções de pertinência do controlador fuzzy	118
B.2 - Tabela dos valores discretos de $U(t)$ do controlador fuzzy tradicional	119
B.3 - Tabela das funções de pertinência do controlador fuzzy inverso	120

REGRAS

---

**Anexo B.**

B.1 - Regras do controlador fuzzy convencional

118

B.1 - Regras do controlador fuzzy a nível móvel

122

---

**ALGORITMOS**

---

**Capítulo 2.**

- 2.1 - Controle da velocidade do motor CC usando a regra MIT 27
- 2.2 - Controle da velocidade do motor CC usando funções de Lyapunov 30

**Capítulo 4.**

- 4.1 - Treinamento em tempo real da RNMC com a nova dinâmica da planta 50
- 4.2 - Treinamento das RNMCs do controlador neural adaptativo indireto 55
- 4.3 - Adaptação da RNMC do controlador usando funções não lineares 60

**Capítulo 6.**

- 6.1 - Treinamento seguido de controle da RNMC 75
- 6.2 - Controle neural "on line" usando o conceito de estado passivo 77
- 6.3 - Controle neural de posição usando o conceito de estado passivo 91
- 6.1 - Adaptação em tempo real da RNMC do controlador neuronal direto inverso 54

**Anexo A**

- A.1 - Funções usadas para calcular a saída e treinar a RNMC 111
- A.2 - Simulação do motor CC 113

## SIMBOLOGIA

Nas equações apresentadas neste trabalho, o significado de cada termo é o seguinte:

**Caracteres latinos:**

$A$ ,  $B$ ,  $C$ ,  $D$ ,  $Y$  e  $X$  são matrizes.

$A_p$  é a amplitude do pulso a ser adicionado a  $U(t)$ ;

$a_1$ ,  $a_2$ ,  $b_1$  e  $b_2$  são elementos da matriz do modelo discreto de estado da planta;

$B_m$  é o Coeficiente de atrito viscoso;

$d/dt$  é o operador da derivada em relação ao tempo;

$e_a(t)$  é a tensão elétrica aplicada nos enrolamentos de armadura do motor CC;

$Ea(s)$  é a transformada de Laplace de  $e_a(t)$ ;

$Ea(k)$  é o valor discreto de  $e_a(t)$ ;

$E_b$  é a força contra-eletromotriz (fcem);

$E(t)$  é a diferença entre as velocidades referência e a velocidade atual do motor CC;

$ERRO$  é a diferença entre a velocidade atual e a velocidade desejada do motor CC;

$F(.)$  e  $G(.)$  são funções não lineares;

$F'(.)$  e  $G'(.)$  são os valores de saída das redes neurais;

$G(s)$  é a função de transferência da planta;

$G_c(s)$  é a função de transferência do controlador;

$G_m(s)$  é a função de transferência do modelo de referência da planta;

$ia(t)$  é a corrente de armadura do motor CC;

$J_m$  é a inércia do rotor do motor CC;

$KK_i$  é a constante de torque do motor CC;

$K_b$  é a constante de fcem do motor CC;

$K(t)$  e  $P(t)$  são matrizes auxiliares do algoritmo LMS recursivo;

$K_p$  é o ganho proporcional do algoritmo PID de controle;

$K_i$  é o ganho integral do algoritmo PID de controle;

$K_d$  é o ganho derivativo do algoritmo PID de controle;

$k$  é a variável tempo (valor discreto);  
 $La$  é a indutância dos enrolamentos de armadura do motor CC;  
 $ln$  é a função logaritmo neperiano;  
 $Np$  é o número de pulsos;  
 $p$  é o operador da derivada em relação ao tempo;  
 $Ra$  é a resistência elétrica dos enrolamentos de armadura do motor CC;  
 $s$  é a representação do tempo em segundos;  
 $s$  é o operador da transformada de Laplace;  
 $s_0$  é uma variável que pode ser adaptada;  
 $T$  é a transposta de uma matriz;  
 $T1, T2, Tt$  e  $T_s$  são períodos de amostragem;  
 $Tl(t)$  é o torque da carga do motor CC;  
 $Tm(t)$  é o torque do motor CC;  
 $Tm$  é a constante mecânica do motor CC;  
 $Te$  é a constante elétrica do motor CC;  
 $t$  é a variável tempo (valores contínuo e discreto);  
 $t_0$  é uma variável que pode ser adaptada;  
 $t1, t1, t2, t3...t14$  são os tempos de transição;  
 $U(t)$  é o sinal de controle da tensão de armadura do motor CC;  
 $u$  é a variável de entrada da planta;  
 $u_c$  é a variável de controle usada em MRAS;  
 $V(t)$  é o valor da saída da RNMC usada como controlador;  
 $x$  é uma variável;  
 $Y(t)$  é o sinal de saída da planta;  
 $Ym(t)$  é o sinal de saída do modelo de referência da planta;  
 $z$  é o operador da transformada Z;  
 $\#neuronios$  é o número de neurônios da camada oculta da RNMC.

### Caracteres gregos:

$\alpha$  é um polo da função de transferência desejada para o motor CC;  
 $\beta$  é um polo da função de transferência do motor CC;  
 $\theta(t)$  é o vetor de parâmetros do motor CC;  
 $\theta(t)$  é o deslocamento angular do rotor do motor CC;  
 $\Omega(t)$  é a velocidade angular do rotor do motor CC;  
 $\Omega(t)$  é o valor de saída da rede neural EMULADOR da velocidade angular do rotor do motor CC;  
 $\Omega_m(t)$  é a velocidade angular de referência do rotor do motor CC;

$\tau$  é a constante elétrica de tempo do motor CC;  
 $\kappa$  é a constante de acionamento do motor CC;  
 $\psi$  é o vetor dos coeficientes desconhecidos;  
 $\psi'(t)$  é a estimação recursiva de  $\psi$  no tempo discreto  $t$ ;  
 $\phi(t)$  é o vetor de regressão do motor CC;  
 $\lambda$  é o fator de esquecimento do estimador recursivo;  
 $\gamma$  é o fator de ajuste da regra MIT e das funções de Lyapunov;  
 $\eta$  é o fator de adaptação do controlador neural;  
 $\chi$  é o fator de adaptação do controlador neural adaptativo;  
 $\mu$  é o fator de convergência usado no algoritmo de propagação retroativa;  
 $\mathfrak{R}$  é o conjunto dos números reais;  
 $\Theta$ ,  $\beta$  e  $T$  são os parâmetros dos neurônios.

**Abreviações:**

APR é o algoritmo de propagação retroativa;  
SCI é o sistema de controle inteligente;  
LMS é o algoritmo dos mínimos quadrados ("Least Mean Square");  
MIT é uma regra usada num algoritmo de controle adaptativo;  
MRAS é o Sistema Adaptativo com Modelo Referência;  
PID é o algoritmo de controle usando um termo proporcional ao erro, um outro proporcional à integral do erro e um termo proporcional a derivada do erro;  
RNMC é a rede neural multi-camadas;  
SISO é o sistema com única entrada e única saída.

### CONTROLADORES NEURAIIS ADAPTATIVOS

O projeto de um controlador requer o conhecimento de um modelo matemático para a descrição do comportamento dinâmico do processo a ser controlado e a aplicação de técnicas analíticas para a dedução da lei de controle. Tal metodologia é empregada, de modo geral, para resolver problemas a partir de objetivos fixos e pré-definidos.

O desenvolvimento tecnológico atual permite a implementação de sistemas de controle com autonomia de operação capazes de estabelecer seus próprios objetivos baseados no reconhecimento da situação (percepção), e podendo utilizar conhecimentos adquiridos anteriormente (experiência). Esses sistemas devem ser capazes de operar com incertezas e tomar decisões em ambientes não estruturados. Esse conceito foi desenvolvido para utilização de técnicas de inteligência artificial no controle de processos.

Durante a década de 50, K. S. Fu introduziu o termo "Controle Inteligente" [Shoureshi, 1991a,b] ou "Sistema de Controle Inteligente" (SCI). SCI é um sistema que possui a habilidade de sentir o seu ambiente, processar as informações para reduzir as incertezas nos parâmetros do processo, planejar, gerar e executar ações de controle. Um SCI deve ser capaz de distribuir as tarefas de decisão entre um conjunto de executores de tarefas, usando intensivamente os computadores disponíveis para inferir o estado atual do sistema e detectar mudanças no estado interno e circunvizinho do sistema. Um SCI se diferencia de sistemas convencionais devido a sua habilidade de tomar decisão e capacidade de aprendizagem. Isto permite ao SCI inferir sobre a dinâmica do processo de uma forma adaptativa e preditiva. Os SCI são necessários em sistemas

com variações na sua circunvizinhança, mudanças nos modelos de referência, diferentes critérios de desempenho, e podendo ser sujeito a falhas de componentes.

Um SCI pode ser implementado com técnicas usuais de controle (controladores PID e controladores adaptativos) acoplado a técnicas de inteligência artificial (neurônios artificiais, conjuntos nebulosos e sistemas especialistas) e utilizando microcomputadores para o controle digital e a gerência do sistema [Porter, 1989].

Recentemente, alguns pesquisadores tem estudado SCI. Dentre esses pesquisadores pode-se citar Narendra [1991] e Narendra & Mukhopadhyay [1991]. Eles mostram que redes neurais artificiais podem ser usadas no controle inteligente de sistemas dinâmicos não lineares. Åström [1991] questiona sobre a "inteligência" de controle inteligente e sugere a combinação entre algoritmos de controle com realimentação e sistemas especialistas para aumentar a capacidade de adaptação dos sistemas de controle.

Atualmente existem diferentes algoritmos de controle com realimentação. A escolha do tipo de controlador depende da dinâmica da planta e das especificações de projeto. Se o índice de desempenho do controlador for modesto, serão adequados controladores com parâmetros constantes, sintonizados para o pior caso. Esses controladores podem ser representados pelos controladores padrão do tipo PID.

No caso da dinâmica da planta variar de uma forma previsível, o controlador deve compensar essa variação ajustando os seus parâmetros. Essas variações são adequadamente manipuladas por controladores do tipo escalonador de ganho que podem ser vistos como um conjunto de controladores convencionais do tipo PID, sintonizados para diferentes dinâmicas da planta [Hägglund & Åström, 1991].

No caso da dinâmica da planta variar de uma forma imprevisível, possivelmente causadas por variações não mensuráveis na sua circunvizinhança (p.ex., causadas por mudanças de temperatura), deve-se usar controladores adaptativos para garantir que ele compense essas variações. Algumas vezes pode ser adequado o uso de um auto-sintonizador ou uma combinação de controladores adaptativos e um auto-sintonizador.

Alguns pesquisadores têm integrado diferentes tipos de controladores para controle de plantas não lineares. Devido a complexidade exigida nessa integração eles usam ferramentas oriundas da área de inteligência artificial. Por exemplo, Handelman [1990] integrou sistemas baseados no conhecimento (regras implementadas em LISP) e escalonador de ganho para controle robótico. Porter [1989] desenvolveu um sistema especialista em tempo real para sintonização de escalonador de ganho inteligente e de controladores PI.

A partir de 1983 [Rumelhart et alii, 1986] e com o ressurgimento do interesse por redes neurais artificiais, diversos autores consideraram redes neurais artificiais como uma alternativa no desenvolvimento de SCI. Dentre os quais podem ser citados: Fukuda & Shibata [1992], Narendra & Parthasaraty [1990,1991], Werbos [1991], Bavarian [1988] e Shoureshi [1991b]. Werbos [1991] criticou alguns autores que consideram redes neurais como uma alternativa mágica na área de controle. Narendra & Parthasaraty [1990] mostraram a viabilidade de se utilizar redes neurais artificiais em controle adaptativo usando modelo referência. No Brasil, diversas equipes das Universidades se interessaram na aplicação de redes neurais em controle adaptativo de sistemas não lineares [Dorneles, 1993]. A partir desse levantamento bibliográfico observou-se a necessidade de se estudar alguns aspectos da utilização prática de controladores neurais adaptativos em um SCI.

Alguns autores, como por exemplo Dote [1988] e Tanomaru & Omatu [1991], têm questionado a adaptação em tempo real de controladores neurais sem o treinamento prévio da dinâmica da planta ou o uso de uma rede neural como emulador.

Nesta Tese será mostrado experimentalmente que, sob certas condições de operação, uma única rede neural artificial, com arquitetura multi-camadas e treinada com o algoritmo de propagação retroativa, pode ser usada como controlador neural com adaptação em tempo real, sem a necessidade do seu treinamento prévio "off line". Inicialmente escolheu-se um sistema motor CC como plataforma de testes experimentais dos controladores a serem estudados. O motor CC foi escolhido devido a sua disponibilidade no nosso laboratório. Após o projeto e a implementação de um novo controlador, geralmente é necessário verificar o seu desempenho e compará-lo com o desempenho de controladores convencionais. Foram feitas simulações e obtidos resultados experimentais usando controladores convencionais do tipo PID, controladores adaptativos do tipo regra MIT e baseado em funções de Lyapunov. Baseado nas publicações técnicas disponíveis sobre controladores neurais, escolheu-se a rede neural multi-camada (RNMC) como arquitetura do controlador neural. Usou-se o algoritmo de propagação retroativa para treinamento da RNMC. A partir de experimentos e de simulações, foi determinada a estratégia de treinamento "off line" e "on line" da RNMC. Após a escolha da RNMC iniciou-se a implementação e o estudo das diferentes estruturas e estratégias de controladores neurais. Usando os resultados experimentais obtidos com os diferentes tipos de controladores neurais, verificou-se o desempenho desses controladores, comparando-os entre si e com os controladores convencionais. Observou-se que o controlador neural direto, sob certas condições na dinâmica da planta e no sinal de referência, pode ser treinado em tempo real com a nova dinâmica da planta. A partir dessa característica da RNMC e da definição de ponto de equilíbrio de um sistema, desenvolveu-se o conceito de estado passivo do sistema integrado neural e planta. Usando o conceito de estado passivo foi possível desenvolver algoritmos em tempo real para controle

adaptativo de plantas não lineares. O mesmo conceito de estado passivo permitiu a determinação do número ótimo de neurônios da camada oculta da RNMC, e a estimação do valor do fator de convergência usado na adaptação da rede neural. Por último mostrou-se experimentalmente que, usando o conceito de estado passivo e uma única RNMC, é possível o posicionamento de um pêndulo invertido.

Este trabalho está organizado em 7 Capítulos cujos conteúdos são sumarizados a seguir.

No Capítulo 1 apresenta-se um sistema controlador digital, baseado em microcomputadores, desenvolvido para o controle de um sistema motor CC com excitação independente. Após a apresentação do modelo discreto em espaço de estado, usando os parâmetros do motor CC identificados "off line", são apresentados os resultados experimentais do acionamento do motor CC usando esses parâmetros. Por último, apresenta-se a identificação recursiva "on line" dos parâmetros do motor CC.

No Capítulo 2, inicialmente mostra-se o projeto e implementação de um controlador PI convencional para controle da velocidade e de um controlador PD convencional para posicionamento do eixo do rotor do motor CC. A seguir, implementa-se controladores adaptativos para controle da velocidade do motor CC. Inicia-se com o controlador adaptativo usando a regra MIT e a seguir, estuda-se o controlador adaptativo baseado em funções de Lyapunov. Conclui-se o Capítulo com uma análise das características principais desses tipos de controladores especificamente em relação ao controle da velocidade do eixo do motor CC.

No Capítulo 3 mostra-se a implementação de um controlador neural usando RNMC e com algoritmo de treinamento de propagação retroativa. A seguir apresentam-se sucintamente detalhes do simulador do sistema motor CC. A partir da análise dos resultados obtidos com o controle neural do sistema motor CC simulado, sugere-se que para aprendizagem da dinâmica direta ou inversa de uma planta pela RNMC, a simetria da planta deve ser considerada.

No Capítulo 4 estuda-se e implementa-se um controlador neural direto usando a inversa da dinâmica de velocidade do motor CC. Sugere-se um algoritmo para treinamento "on line" da nova dinâmica inversa do motor CC após variação nos seus parâmetros. A seguir, após a verificação de que se necessita do jacobiano da planta para treinamento do controlador neural direto, mas como não se conhece esse jacobiano, constata-se que se pode usar redes neurais para emular a planta e, usando o algoritmo de propagação retroativa, se pode calcular indiretamente o jacobiano da planta usando uma RNMC EMULADOR ou, no caso do motor CC, adaptar diretamente a rede neural controladora usando as características da dinâmica da velocidade do motor CC. A seguir, a partir da análise das características de um controlador neural desenvolvido para controle de

plantas não lineares, sugere-se um algoritmo capaz de transformar o controlador neural direto num controlador neural adaptativo direto.

No Capítulo 5, baseado em resultados experimentais obtidos no controle da velocidade do motor CC, faz-se uma comparação sucinta entre os desempenhos dos controladores convencionais e controladores não convencionais. A seguir, compara-se o desempenho entre os controladores neurais e sugere-se o tipo de controlador neural a ser usado nesta Tese.

No Capítulo 6 define-se estado passivo do sistema neural e motor CC. A seguir, mostra-se que, usando o conceito de estado passivo, é possível o treinamento e adaptação em tempo real de uma única RNMC usada como controlador neural direto, sem o prévio treinamento da RNMC "off line". Ainda usando o conceito de estado passivo, mostra-se que é possível a determinação do número ótimo de neurônios na camada oculta da RNMC, e a estimação do fator de convergência usado na adaptação da rede neural. Por último, apresentam-se resultados experimentais obtidos, usando o controlador neural direto e o conceito de estado passivo, no posicionamento de um pêndulo invertido.

No Capítulo 7, após considerações sucintas sobre os resultados experimentais obtidos usando o controlador neural direto, conclui-se este trabalho com sugestões de trabalhos futuros usando controladores neurais adaptativos.

## O SISTEMA DE ACIONAMENTO DO MOTOR CC

### 1.1 - Introdução

Neste Capítulo apresenta-se o sistema motor CC usado como plataforma de testes dos algoritmos de controle estudados nesta Tese.

Inicialmente, apresentam-se as características de um sistema motor CC com excitação independente. A seguir, apresenta-se o modelo discreto em espaço de estados e calcula-se a matriz do modelo discreto usando os parâmetros do motor CC calculados "off line". Na última Seção, mostra-se como identificar, em tempo real, os parâmetros do motor CC usando o método dos mínimos quadrados recursivos. Conclui-se o Capítulo com comentários sobre a não linearidade observadas nas características do sistema motor CC.

### 1.2 - Características do Sistema Motor de Corrente Contínua

O sistema motor CC, mostrado na Figura 1.1, é composto de um motor CC acoplado a um gerador CC, processadores digitais de sinais, e microcomputador pessoal. Usa uma placa multifunção, compatível com o microcomputador pessoal IBM PC, contendo dois canais para conversão A/D de 10bit-25 $\mu$ S, um temporizador de intervalos programáveis de 16bit (PIT-8254) e um circuito de interface de entrada e saída digital de 24bit (PPI-8255). As CPUs, V20 do IBM PC e TMS320C25 do processador digital de sinal, são usadas no modo mestre e escravo. Para

PC e TMS320C25 do processador digital de sinal, são usadas no modo mestre e escravo. Para eles Cavalcanti et alii [1988][1992c] desenvolveram um executivo em tempo real para supervisionar as tarefas críticas no tempo de aquisição de dados e de geração de comandos de controle necessários ao sistema de controle de velocidade do motor CC.

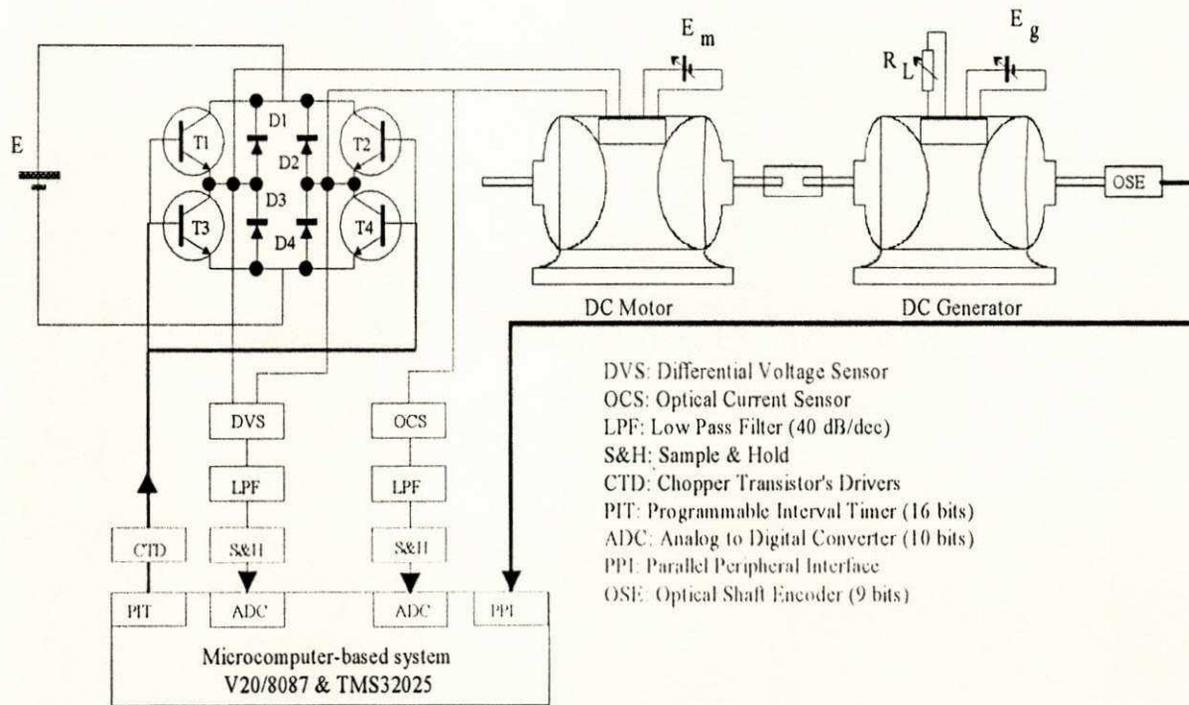


Figura 1.1 Sistema de acionamento experimental do motor CC.

A velocidade do rotor é calculada, no TMS320C25, a partir da leitura de 9 bits de um codificador ótico, acoplado ao rotor, usando a PPI. O temporizador de intervalos programáveis opera sob o controle da CPU V20 para gerar sinais de comandos aos quatro transistores darlington de potência do conversor estático.

A análise e o projeto de um sistema de acionamento típico empregando motores de corrente contínua, de modo geral, pode ser feito usando modelos lineares invariantes no tempo. Na montagem experimental da Figura 1.1, existem algumas fontes que podem forçar o sistema de acionamento exibir características não lineares. A queda de tensão sobre os transistores darlington do chaveador é bastante elevada e depende da corrente do motor. A tensão sobre os diodos de roda livre durante a condução é muito menor que a tensão sobre a chave darlington. Essas tensões introduzem variações na tensão da armadura do motor. Outras não linearidades são introduzidas por saturação magnética da máquina, atrito estático, e efeito de quantização nos conversores A/D [Kuo, 1985]. Essas não linearidades não foram modeladas nos estudos de simulações mas estão presentes nos testes experimentais do sistema de acionamento do motor CC.

O método PWM utilizado é ilustrado na Figura 1.2 [Lima et alii, 1993]. Na parte esquerda da Figura 1.2 é mostrada a curva da tensão de referência  $U(t)$  em função do tempo  $t$ . No instante  $k$  é mostrada a tensão de referência  $e_a^*(t)$  que deve ser aplicada à carga através do circuito chaveador. Na parte direita da Figura 1.2 é mostrado o sinal do comando PWM, equivalente a tensão de referência  $e_a^*(t)$ , que deve ser imposto ao chaveador no instante  $t$  durante o período de amostragem  $T_I$ . Usou-se a seguinte equação para calcular o período de amostragem:  $T_I = \frac{T_t}{2} + \frac{T_t e_a(t)}{2E}$

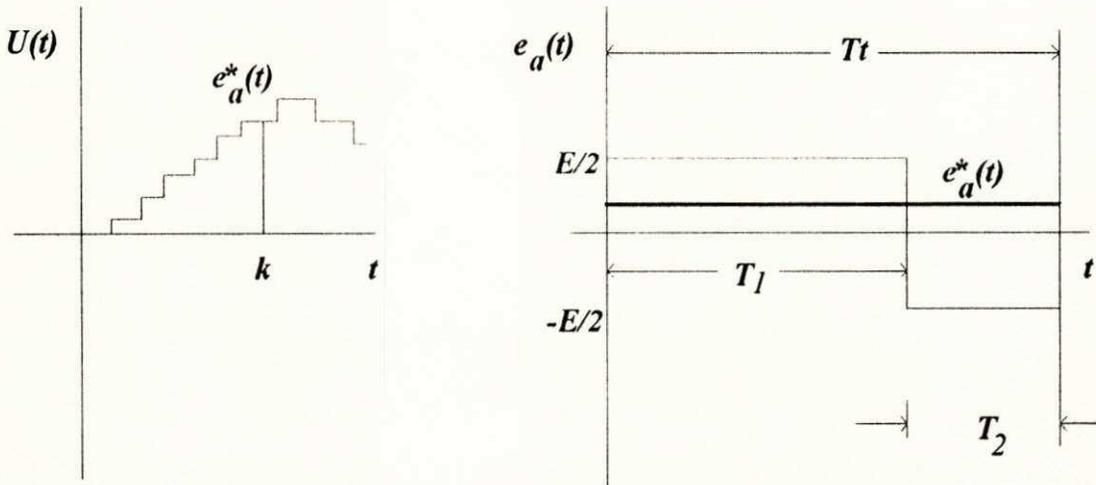


Figura 1.2 Geração do sinal PWM de tensão do chaveador.

Na Figura 1.3 é mostrado o diagrama de blocos do motor CC. As funções de transferência dos diferentes componentes do motor CC são representadas com o operador  $s$  da transformada de Laplace [D'azzo, 1988][Kuo, 1985].

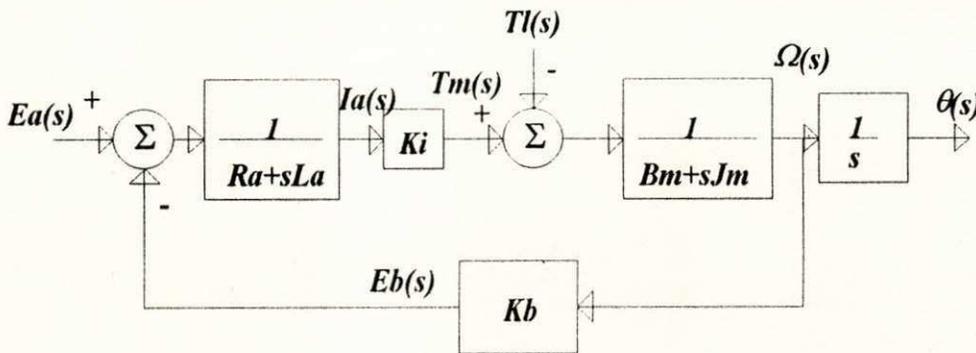


Figura 1.3 Diagrama de blocos do sistema motor CC.

As variáveis de estado do sistema motor CC podem ser definidas como  $\theta(t)$ ,  $\Omega(t)$  e  $Ia(t)$ . O modelo de estado na forma contínua do motor CC é mostrado na equação 1.1.

$$\begin{bmatrix} dI_a(t)/dt \\ d\Omega(t)/dt \\ d\theta(t)/dt \end{bmatrix} = \begin{bmatrix} -Ra/La & -Kb/La & 0 \\ Ki/Jm & -Bm/Jm & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} I_a(t) \\ \Omega(t) \\ \theta(t) \end{bmatrix} + \begin{bmatrix} 1/La & 0 \\ 0 & -1/Jm \\ 0 & 0 \end{bmatrix} \begin{bmatrix} e_a(t) \\ Tl(t) \end{bmatrix} \quad (1.1)$$

Os parâmetros do motor CC, determinados por Jacobina et alii [1992], usando o método dos mínimos quadrados, são mostrados na Tabela 1.1.

Tabela 1.1 - Parâmetros do motor CC

Resistência dos enrolamentos de armadura	$Ra=0.8756\Omega$
Indutância de armadura	$La=0.0292H$
Inércia do rotor do motor	$Jm=0.00063078 \text{ Kg m}^2$
Coefficiente de atrito viscoso	$Bm=0.0023 \text{ Kg m}^2/(\text{rad S})$
Constante de torque	$Ki=0.0869 \text{ N m/A}$
Constante de fcm	$Kb=0.0869 \text{ V}/(\text{rad/S})$

O polinômio característico do modelo do sistema motor CC, descrito pela equação 1.1, em função da sua posição angular  $\theta(s)$ , desprezando o torque  $Tl(t)$ , é mostrado na equação 1.2.

$$\frac{LaJm}{Ki} s^3 \theta(s) + \frac{LaBm + RaJm}{Ki} s^2 \theta(s) + \frac{RaBm + KiKb}{Ki} s \theta(s) = Ea(s) \quad (1.2)$$

Sendo a indutância da armadura muito pequena [D'azzo, 1988][Ogata, 1982], a constante de tempo elétrica pode ser dada por  $\tau = (RaJm)/(RaBm + KbKi)$ , e a constante de acionamento do motor CC pode ser dada por  $\kappa = Ki/(RaBm + KbKi)$  [Dessaint et alii, 1990]. O modelo contínuo de segunda ordem da função de transferência do motor CC é representado pela equação 1.3.

$$G(s) = \frac{\theta(s)}{Ea(s)} = \frac{\kappa}{s(\tau s + 1)} \quad (1.3)$$

O modelo discreto em espaço de estado do motor CC, com  $T_s$  representando tempo de amostragem e desprezando o torque  $Tl(t)$ , é representado na forma matricial da equação 1.4a ou na forma simplificada mostrada na equação 1.4b.

$$\begin{bmatrix} \theta(t+1) \\ \Omega(t+1) \end{bmatrix} = \begin{bmatrix} 1 & a_1 \\ 0 & a_2 \end{bmatrix} \begin{bmatrix} \theta(t) \\ \Omega(t) \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} Ea(t) \quad (1.4a)$$

$$Y(t+1) = AY(t) + BEa(t) \quad (1.4b)$$

Os parâmetros das matrizes  $A$  e  $B$ , mostrados na equação 1.4a, são descritos nas equações 1.5a, 1.5b, 1.5c e 1.5d.

$$a_1 = T_s(1 - e^{-T_s/\tau}) \quad (1.5a)$$

$$a_2 = e^{-T_s/\tau} \quad (1.5b)$$

$$b_1 = \kappa[T_s - \tau(1 - e^{-T_s/\tau})] \quad (1.5c)$$

$$b_2 = \kappa[1 - e^{-T_s/\tau}] \quad (1.5d)$$

### 1.3 - Identificação das constantes elétrica de tempo e de acionamento do motor CC

Usando os valores dos parâmetros do motor CC mostrados na Tabela 1.1 obtêm-se os valores mostrados na equação 1.6, para as constantes de acionamento e constante de tempo elétrica do motor CC. A partir da constante de tempo elétrica do motor CC e das características do seu sistema de acionamento, escolheu-se  $T_s = 0.012s$  como tempo de amostragem.

$$G(s) = \frac{9.0847}{s(0.0577s + 1)} \quad (1.6)$$

Os elementos da matriz do modelo discreto da planta, mostrados na equação 1.7, foram calculados com os valores de  $T_s$ ,  $\kappa$  e de  $\tau$  definidos na equação acima.

$$\begin{bmatrix} \theta(t+1) \\ \Omega(t+1) \end{bmatrix} = \begin{bmatrix} 1 & 0.0108 \\ 0 & 0.8121 \end{bmatrix} \begin{bmatrix} \theta(t) \\ \Omega(t) \end{bmatrix} + \begin{bmatrix} 0.0106 \\ 1.7067 \end{bmatrix} Ea(t) \quad (1.7)$$

Uma das formas de se identificar os parâmetros do motor CC é usando a função de transferência chaveada do motor CC mostrada na equação 1.8.

$$G(z) = \frac{\theta(z)}{Ea(z)} = \frac{b_1z + b_2}{z^2 + a_1z + a_2} \quad (1.8)$$

Na equação 1.9 são mostrados os parâmetros de  $G(z)$  que foram calculados com os valores de  $T_s$ ,  $\kappa$  e de  $\tau$  definidos na equação 1.6.

$$G(z) = \frac{0.0106z + 0.0099}{z^2 - 1.8128z + 0.8128} \quad (1.9)$$

Quando não se conhecem os parâmetros do motor CC, os parâmetros de  $G(z)$  devem ser identificados. A estimação dos parâmetros da equação 1.8 pode ser feita usando o método dos mínimos quadrados recursivos. Da equação 1.8 obtém-se a equação de diferença mostrada na equação 1.10.

$$\theta(t) - \theta(t-1) = a_2[\theta(t-1) - \theta(t-2)] + b_1Ea(t-1) + b_2Ea(t-2) \quad (1.10)$$

Definindo-se  $e(t) = \theta(t) - \theta(t-1)$ , obtêm-se a equação 1.11.

$$e(t) = a_2e(t-1) + b_1Ea(t-1) + b_2Ea(t-2) \quad (1.11)$$

Definindo-se o vetor de parâmetros do motor CC como:

$$\psi'(t) = [a_2(t) \quad b_1(t) \quad b_2(t)]^T \quad (1.12)$$

Obtêm-se a equação 1.13 dos mínimos quadrados [Åström & Wittenmark, 1989].

$$\psi'(t) = \psi'(t-1) + K(t-1)[e(t) - \phi^T(t)\psi'(t-1)] \quad (1.13)$$

Onde  $\phi(t)$ , mostrado na equação 1.14, é o vetor de medição de entrada e saída do motor CC.

$$\phi(t) = [e(t-1) \quad Ea(t-1) \quad Ea(t-2)]^T \quad (1.14)$$

Os componentes do vetor  $K(t)$  são calculados pelas equações 1.15a e 1.15b [Ljung & Söderström, 1983].

$$K(t-1) = \frac{P(t-1)\phi(t)}{(\lambda + \phi^T(t)P(t-1)\phi(t))} \quad (1.15a)$$

$$P(t-1) = \frac{(1 - K(t-1)\phi^T(t))P(t-1)}{\lambda} \quad (1.15b)$$

Usando-se os parâmetros  $a_2$ ,  $b_1$  e  $b_2$  determinados usando o método dos mínimos quadrados recursivos, obtêm-se as constantes de acionamento e constante de tempo elétrica do motor CC mostrados nas equações 1.16a e 1.16b, respectivamente.

$$\tau = -\frac{T_s}{\ln(a_2)} \quad (1.16a)$$

$$\kappa = \frac{b_1 + b_2}{T_s(1 - a_2)} \quad (1.16b)$$

Na Figura 1.4 são mostradas as formas de ondas da velocidade angular, tensão e corrente de armadura do motor CC obtidas experimentalmente com o sistema mostrado na Figura 1.1. Essas formas de ondas, representadas em p.u., estão em função do tempo com  $T_s = 0.012s$ . Elas foram geradas a partir do sinal de controle do chaveador da tensão de armadura, representado por  $U(t)$  (ver a Figura 1.2) com valor de base de 12Volts, uma corrente de armadura  $I_a(t)$  com valor de base de 4.8A, e a velocidade angular  $\Omega(t)$  com valor de base de 120rpm.

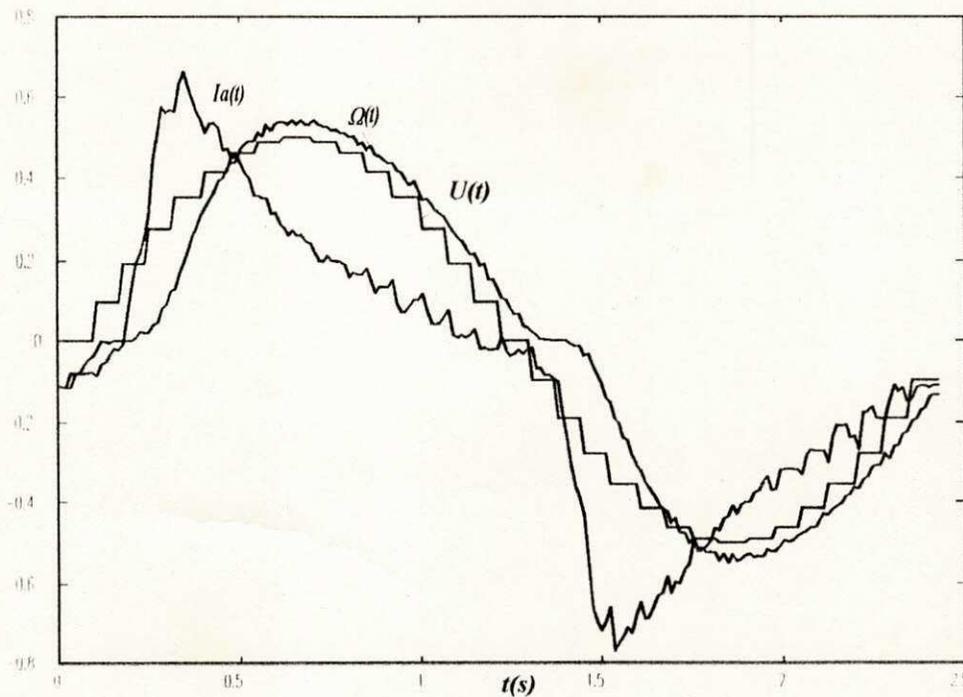


Figura 1.4 - Curvas experimentais da velocidade angular, tensão e corrente de armadura.

Observe-se, na Figura 1.4, a não linearidade evidente no sinal da velocidade angular  $\Omega(t)$ . Uma das causas dessa não linearidade é o ruído gerado pelo algoritmo usado para calcular a velocidade angular a partir da posição angular do eixo do motor CC. Uma outra causa da não linearidade é a zona morta gerada pelos transistores do conversor estático que pode ser vista próximo do ponto de velocidade zero. Além dessas causas, podemos citar os erros de quantização que ocorrem na conversão das variáveis analógicas em variáveis digitais.

A identificação dos parâmetros de  $G(z)$ , usando a equação 1.11, necessita de  $e(t) = \theta(t) - \theta(t-1)$ . Devido ao ruído de quantização existente em  $\Omega(t)$ , não é adequada a utilização da equação 1.11 para identificação recursiva das constantes de acionamento e elétrica de tempo do motor CC.

Uma outra forma de se calcular  $\tau$  e  $\kappa$  é usando o modelo contínuo de primeira ordem do motor CC, mostrado na equação 1.17. A função de transferência chaveada da velocidade é mostrada na equação 1.18a, e a equação de diferença do motor CC é mostrada na equação 1.18b.

$$G(s) = \frac{\Omega(s)}{Ea(s)} = \frac{\kappa}{(\tau s + 1)} \quad (1.17)$$

$$G(z) = \frac{\Omega(z)}{Ea(z)} = \frac{b_2}{(z - a_2)} \quad (1.18a)$$

$$\Omega(t) = a_2 \Omega(t-1) + b_2 Ea(t-1) \quad (1.18b)$$

Definindo-se o vetor de parâmetros do motor CC como:

$$\theta(t) = [-a_2 \quad b_2]^T \quad (1.19)$$

Usando-se a equação 1.13 dos mínimos quadrados, onde  $\phi(t)$ , mostrado na equação 1.20, é o vetor de regressão do motor CC.

$$\phi(t) = [\Omega(t-1) \quad Ea(t-1)]^T \quad (1.20)$$

Os componentes do vetor  $K$  são calculados pelas equações 1.15a e 1.15b. Usando-se os parâmetros  $a_2$  e  $b_2$  identificados usando o método dos mínimos quadrados recursivos, pode-se calcular  $\tau$  e  $\kappa$  como mostrado nas equações 1.21a e 1.21b.

$$\tau = -\frac{T_s}{\ln(a_2)} \quad (1.21a)$$

$$\kappa = \frac{b_2}{(1 - a_2)} \quad (1.21b)$$

Usando-se os valores de  $Ea(t)$  e  $\Omega(t)$  mostrados nas formas de ondas da Figura 1.4 e o método dos mínimos quadrados recursivos, calculou-se  $a_2 = 0.9401$  e  $b_2 = 0.3494$ . Usando as equações 1.21a e 1.21b, calculou-se  $\tau = 0.1943$  e  $\kappa = 5.83$ .

#### 1.4 - Conclusão

Neste Capítulo, inicialmente foram apresentadas as características principais do sistema de acionamento do motor CC usado como plataforma de testes experimentais para os controladores convencionais e não convencionais a serem estudados neste trabalho. A seguir, usando os parâmetros do motor CC calculados "off line", apresentaram-se os resultados experimentais do acionamento do motor CC. Foi mostrado como se identificar em tempo real, usando o algoritmo dos mínimos quadrados recursivos, os parâmetros do modelo contínuo de primeira e de segunda ordem do motor CC.

Devido aos ruídos de medição de  $\theta(t)$ , observou-se que a identificação dos parâmetros do motor CC mostrado na Figura 1.1, usando o algoritmo dos mínimos quadrados recursivos, deve ser feita usando o modelo contínuo de primeira ordem do motor CC.

## CONTROLADORES CONVENCIONAIS

### 2.1 - Introdução

Neste Capítulo apresentam-se os resultados experimentais obtidos do controle da velocidade do motor CC usando diferentes tipos de controladores convencionais. Esses resultados, no Capítulo 5, serão comparados com os resultados experimentais obtidos usando controladores neurais.

Inicialmente, apresentam-se os projetos (usando alocação e imposição de pólos) e resultados experimentais de um controlador proporcional integral (PI), para controle da velocidade do rotor do motor CC, e de um controlador proporcional derivativo (PD), para posicionamento do eixo do rotor do motor CC. A seguir, descreve-se controlador adaptativo e se escolhem dois tipos de controladores adaptativos. Analisa-se o sistema de controle adaptativo com modelo de referência (MRAS) para sistemas contínuos usando a regra MIT e baseado na teoria da estabilidade de Lyapunov, mostram-se a implementação e resultados experimentais obtidos com o sistema de controle de velocidade do motor CC usando a regra MIT e as funções de Lyapunov.

## 2.2 - Controladores Proporcionais Integrais (PI)

O projeto do controlador PI segue o procedimento de projeto de Guillemin-Truxal [D'AZZO, 1988]. Considere-se o motor CC como uma planta com a função de transferência mostrada na equação 2.1 e com os parâmetros  $\kappa$  e  $\tau$  da equação 1.6 (ver Seção 1.3).

$$G(s) = \frac{\Omega(s)}{Ea(s)} = \frac{\kappa}{(\tau s + 1)} = \frac{5.83}{(0.1943s + 1)} \quad (2.1)$$

Definindo-se para o controlador PI  $E(s) = \Omega_r(s) - \Omega(s)$ , e com a função de transferência  $Gc(s) = U(s) E(s) = K_p + K_i/s$  (ver Figura 2.1), calculou-se  $K_p$  e  $K_i$  seguindo o projeto de cancelamento e alocação de pólos de Guillemin-Truxall que gera um  $Gc(s)$  que conduz a uma função de transferência total do sistema dentro das especificações desejadas [D'Azzo, 1988].

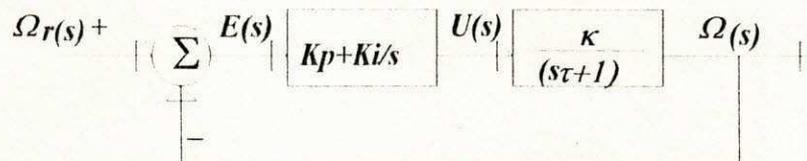


Figura 2.1 Diagrama de blocos do controlador PI de velocidade.

Definindo-se as variáveis auxiliares  $p3 = \kappa/\tau$  e  $p4 = 1/\tau$ , especificando  $D(s) = K'(s + \alpha)$  como a função de transferência  $\Omega(s) / \Omega_r(s)$  desejada para o sistema motor CC, obtém-se a equação 2.2 em malha fechada do sistema motor CC.

$$\frac{\Omega(s)}{\Omega_r(s)} = \frac{K_p p3 (s + K_i / K_p) / s (s + p4)}{1 + K_p p3 (s + K_i / K_p) / s (s + p4)} = \frac{K'}{(s + \alpha)} \quad (2.2)$$

Na primeira etapa cancela-se um polo da função de transferência. O polo restante é usado como polo desejado para resposta do sistema motor CC. Definindo-se  $K_i / K_p = p4$ , obtém-se a equação 2.3 em malha fechada do sistema motor CC.

$$\frac{K_p p3 / s}{1 + K_p p3 / s} = \frac{K_p p3}{(s + K_p p3)} = \frac{K'}{(s + \alpha)} \quad (2.3)$$

O valor do polo  $\alpha$  é definido na equação 2.4.

$$\alpha = K_p p3 \quad (2.4)$$

Usando-se os valores dos parâmetros do motor CC da equação 2.1 e alocando o polo  $\alpha$  de acordo com a dinâmica desejada para o sistema motor CC e o controlador PI ( $\alpha = 6p4$ ), obtêm-se os valores para os parâmetros do controlador PI mostrados nas equações 2.5a e 2.5b.

$$K_p = \alpha p3 = 6/\kappa = 1.03 \quad (2.5a)$$

$$Ki = p \cdot Kp = Kp \cdot \tau = 5.3 \quad (2.5b)$$

Usando-se a transformada Z e admitindo que o sinal de controle permanece constante durante o intervalo de amostragem ("zero-order-hold"), o controlador PI digital pode ser descrito pela função transferência mostrada na equação 2.6.

$$G_c(z) = Kp + \frac{KiT_s}{(z-1)} = \frac{Kpz + (KiT_s - Kp)}{(z-1)} \quad (2.6)$$

A partir da equação 2.6 e definindo-se  $E(t) = \Omega_r(t) - \Omega(t)$ , obtém-se a equação de diferença 2.7.

$$U(t+1) = U(t) + KpE(t) + (KiT_s - Kp)E(t-1) \quad (2.7)$$

Usando-se tempo de amostragem  $T_s = 0.012s$ , obtém-se:

$$U(t+1) = U(t) + 1.03E(t) - 0.97E(t-1) \quad (2.8)$$

Na Figura 2.2 são mostradas as curvas da velocidade referência ( $\Omega_r(t)$  representada em rps), curva da velocidade real ( $\Omega(t)$  representada em rps), e curva da corrente de armadura ( $I_a(t)$  representada em Ampères) do motor CC em função do tempo (em segundos), obtidas experimentalmente com o sistema de controle digital mostrado na Figura 2.1. Usaram-se dois perfis de velocidade angular de referência (onda quadrada e senoidal) para caracterização do sistema motor CC com controlador PI.

O índice de desempenho escolhido para comparação dos controladores estudados neste trabalho foi o somatório do quadrado do erro, mostrado na equação 2.9. Define-se o erro como a diferença entre a velocidade de referência e a velocidade real do rotor do motor CC,  $E(t) = \Omega_r(t) - \Omega(t)$ . Os somatórios do quadrado dos erros obtidos com as formas de ondas senoidais (E2S) e quadradas (E2Q) foram  $E2S = 12.5 \text{ rps}^2$  e  $E2Q = 101.4 \text{ rps}^2$ , respectivamente.

$$\sum_1^{300} E_t^2 = \sum_t [\Omega_r(t) - \Omega(t)]^2 \quad (2.9)$$

Na Figura 2.2, pode-se observar que quando a referência é uma onda quadrada houve um sobressinal na velocidade de aproximadamente 15% gerado intencionalmente no projeto do controlador PI.

Na Figura 2.2, quando a referência foi uma onda senoidal e a velocidade referência está em torno de zero, observou-se uma não linearidade na velocidade do motor CC. Essa não linearidade pode ser explicada pela zona morta que ocorre nos transistores do conversor estático.

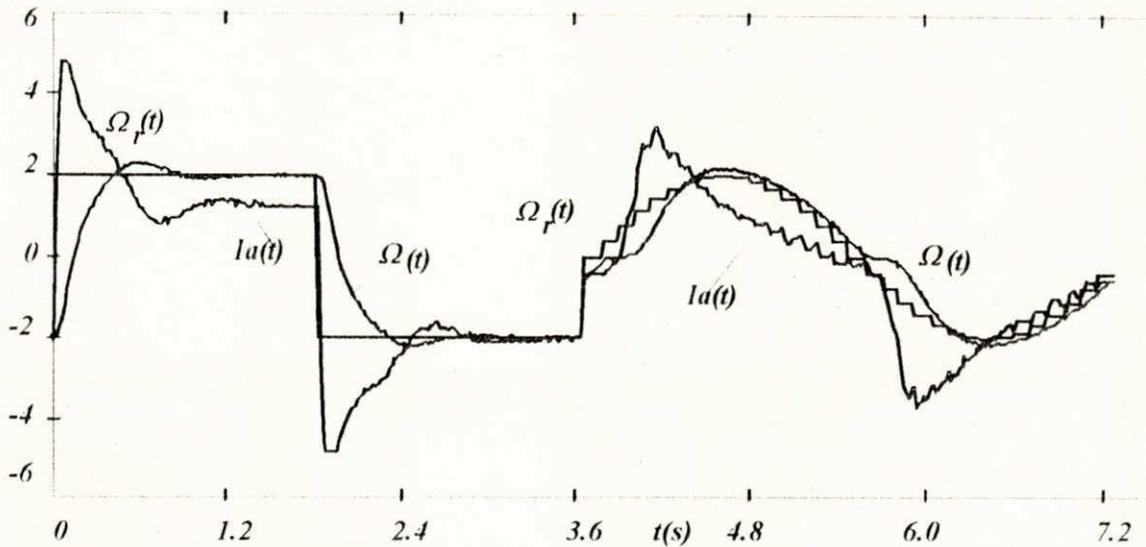


Figura 2.2 Resultados experimentais obtidos com o controlador PI.

### 2.3 - Controladores Proporcionais Derivativos (PD)

O projeto do controlador PD segue o procedimento de projeto de Guillemin-Truxal [D'AZZO, 1988]. Considere o motor CC como uma planta com a função de transferência mostrada na equação 2.10 e com os parâmetros  $\kappa$  e  $\tau$  da equação 1.6 calculados na Seção 1.3.

$$G(s) = \frac{\theta(s)}{Ea(s)} = \frac{\kappa}{s(\tau s + 1)} = \frac{5.83}{s(0.1943s + 1)} \quad (2.10)$$

Para o controlador de posição PD, definindo  $E(s) = \theta_r(s) - \theta(s)$ , e com a função de transferência  $G_c(s) = U(s)/E(s) = K_p + sK_d$  (ver Figura 2.3), foram calculados  $K_p$  e  $K_d$  seguindo o projeto de cancelamento de pólos de Guillemin-Truxall [D'Azzo, 1988].

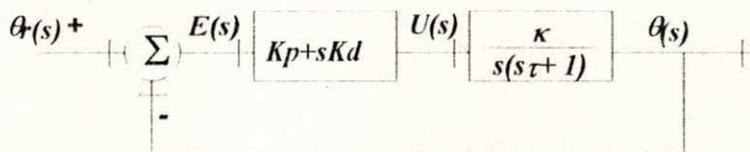


Figura 2.3 Diagrama de blocos do controlador PD de posição.

Especificando  $D(s) = K'(s + \alpha)$  como a função de transferência  $\theta(s)/\theta_r(s)$  desejada para o sistema motor CC, obtém-se a equação 2.11 em malha fechada do sistema motor CC.

$$D(s) = \frac{\Theta(s)}{\theta(s)} = \frac{\frac{\kappa}{s(1+T_s)}(Kp + sKd)}{1 + \frac{\kappa}{s(1+T_s)}(Kp + sKd)} = \frac{K'}{s + \alpha} \quad (2.11)$$

Considere  $\alpha$  e  $\beta$  como polos do sistema da equação 2.11 e faça  $\beta = 1/T_s$ . Na primeira etapa cancela-se um polo da função de transferência. O polo restante é usado como polo desejado para resposta do sistema motor CC. Após algumas simplificações, obtêm-se as equações 2.12a, 2.12b e 2.12c.

$$K' = \kappa K d T_s \quad (2.12a)$$

$$Kp = \alpha \kappa \quad (2.12b)$$

$$Kd = \alpha T_s \kappa \quad (2.12c)$$

Usando-se os valores dos parâmetros do motor CC da equação 2.12 e alocando o polo  $\alpha$  de acordo com a dinâmica desejada para o sistema motor CC e o controlador PD ( $\alpha = 10$ ), obtêm-se os valores para os parâmetros do controlador PD mostrados nas equações 2.13a e 2.13b.

$$Kp = 1.7153 \quad (2.13a)$$

$$Kd = 0.3333 \quad (2.13b)$$

Usando-se a transformada Z e admitindo que o sinal de controle permaneça constante durante o intervalo de amostragem ("zero-order-hold"), o controlador PD digital pode ser descrito pela função transferência mostrada na equação 2.14.

$$G_C(z) = Kp + \frac{Kd(z-1)}{T_s z} = \frac{(T_s Kp + Kd)z - Kd}{T_s z} \quad (2.14)$$

A partir da equação 2.14 obtêm-se a equação de diferença 2.15.

$$U(t+1) = (Kp + Kd T)E(t) - (Kd T)E(t-1) \quad (2.15)$$

Usando-se tempo de amostragem de  $T_s = 0.012s$ , obtêm-se a equação de diferença 2.16.

$$U(t+1) = 3.43E(t) - 1.72E(t-1) \quad (2.16)$$

Na Figura 2.4 são mostradas as curvas da posição referência e a curva da posição real ( $\Theta(t)$  e  $\theta(t)$ ) representados em p.u. com valor de base de  $2\pi$ rad), e curva da tensão de armadura ( $U(t)$ ) representada em p.u. com valor de base de 12V) do motor CC em função do tempo (em milissegundos), obtidas experimentalmente com o sistema de controle digital mostrado na Figura 2.3. Na Figura 2.4, pode-se observar que existe um erro de regime permanente,  $E(t) = \Theta(t) - \theta(t)$ ,

que pode ser explicado pelo atrito não linear do eixo do rotor, pela zona morta que ocorre nos transistores do conversor estático, e pela quantização dos sinais analógicos nos conversores A/D.

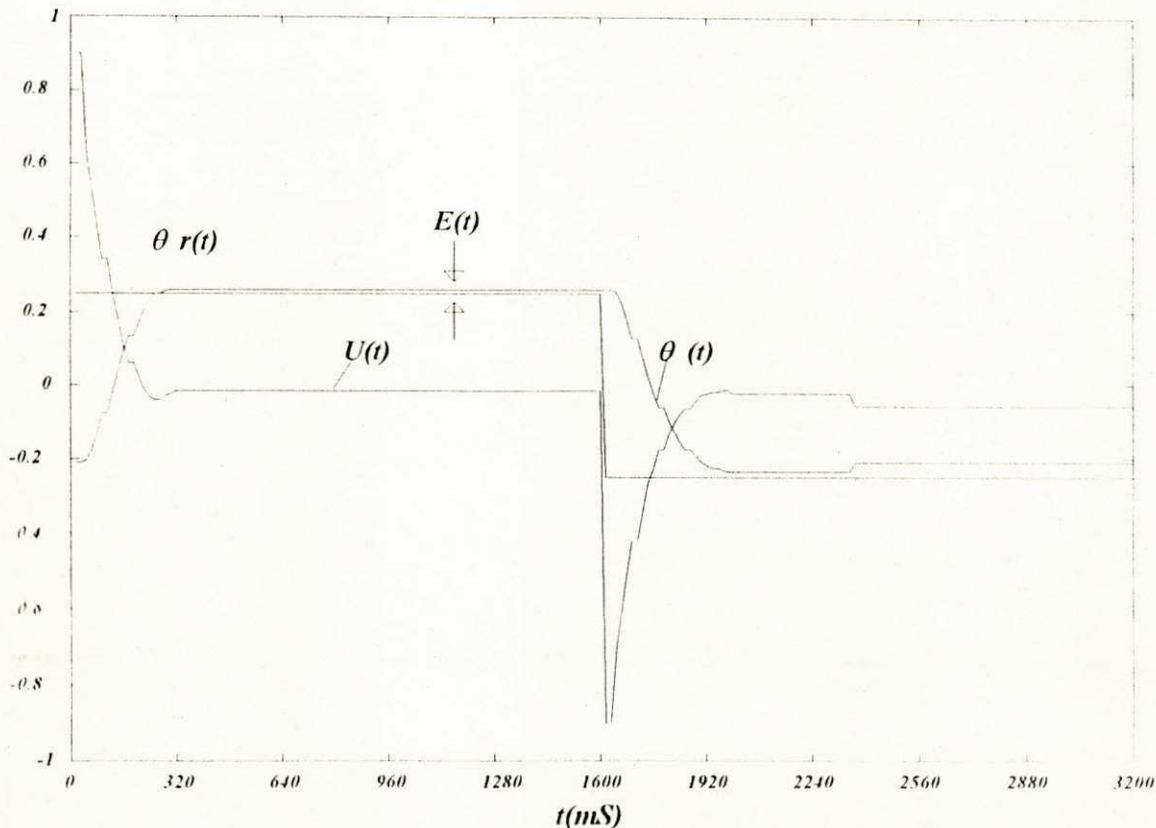


Figura 2.4 Resultados experimentais obtidos com o controlador PD.

## 2.4 - Sistema Adaptativo com Modelo de Referência (MRAS)

Quando somente existe incerteza paramétrica da planta sob controle, a escolha natural é controle adaptativo [Åström & Wittenmark, 1989]. Controladores adaptativos são adequados para plantas que não requerem alguma forma de autonomia de operação. A partir da década de cinquenta, foram desenvolvidos controladores, baseados num modelo de referência, em que os parâmetros da planta podem se modificar durante o controle.

O primeiro sistema de controle com adaptação própria [Parks, 1981] foi proposto, num Simpósio realizado no "Wright Air Development Center" em Dayton, Ohio, USA, na década de cinquenta, para aplicação na área de engenharia aeronáutica. Foi proposto um auto-piloto adaptativo para compensar as mudanças aerodinâmicas causadas por variações na altitude e velocidade dos aviões. Nesse simpósio foram propostos dois esquemas de controladores baseados em modelos de referência em que os modelos ideais das plantas estavam em série com servo-mecanismos de alto ganho, ou em paralelo com sistemas de adaptação.

No primeiro esquema, arquitetura em série, a resposta ideal é fornecida por um modelo de referência através de um sistema de realimentação de alto-ganho (Figura 2.5). O controlador robusto de alto ganho ("Robust High-gain Control") [Åström & Wittenmark, 1989], é um regulador de ganho constante com a característica de forçar a saída  $\Omega$  da planta a seguir a saída  $\Omega_m$  do modelo de referência.

O segundo esquema, proposto pelos pesquisadores Osburn, Whitaker e Kezer, ficou conhecido como "Sistema Adaptativo com Modelo de Referência - MRAS", com arquitetura em paralelo, a resposta atual da planta é comparada com a resposta ideal do modelo gerando um sinal de erro que é usado para ajustar o controlador, compensando assim as mudanças nos parâmetros do sistema (Figura 2.6)[Åström & Wittenmark, 1989].

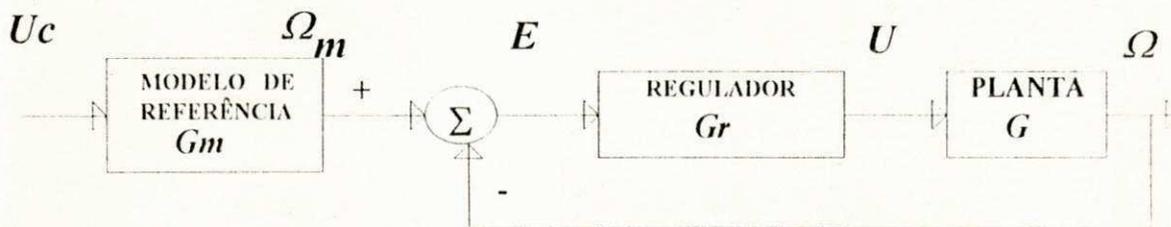


Figura 2.5 Controlador robusto de alto ganho.

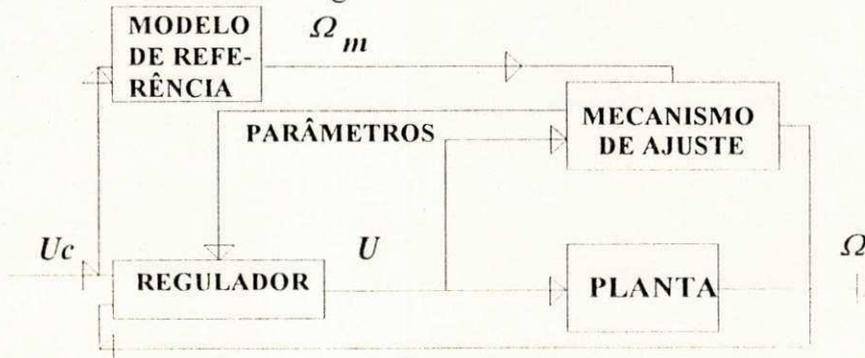


Figura 2.6 Sistema adaptativo com modelo referência.

Diversos pesquisadores definiram sucintamente controle adaptativo. Åström define controle adaptativo como um tipo especial de controle não linear realimentado em que os estados do processo podem ser agrupados em duas categorias que se modificam a diferentes velocidades [Åström & Wittenmark, 1989]. Os estados que se modificam mais lentamente são vistos como parâmetros. Outro pesquisador, Landau [1979] define um sistema adaptativo como um sistema que avalia um determinado índice de desempenho (ID) usando as entradas, os estados, e as saídas do sistema ajustável. A partir da comparação entre o ID calculado e um conjunto de valores de ID pré-definidos, o mecanismo de adaptação modifica os parâmetros do sistema ajustável ou gera uma entrada auxiliar no intuito de manter o ID próximo do conjunto pré-definido.

A estratégia MRAS introduz o conceito de duas escalas de tempo em sistemas de controle: a escala mais rápida para a realimentação normal e a escala mais lenta para a atualização dos parâmetros do regulador [Åstrom & Wittenmark, 1989]. Além disso, MRAS introduziu duas novas idéias: 1) o desempenho de um sistema é especificado por um modelo; 2) os parâmetros do regulador são ajustados baseados no erro existente entre as saídas do modelo de referência e da planta.

A estratégia MRAS foi proposta originalmente para sistemas determinísticos contínuos no tempo, sendo depois estendido para sistemas discretos no tempo e para sistemas com perturbações aleatórias. Existem três abordagens principais para análise e projeto de MRAS: a) abordagem do gradiente; b) funções de Lyapunov; c) teoria da passividade.

A seguir apresentam-se o projeto e implementação de dois sistemas de controle baseados na estratégia MRAS para o controle da velocidade do motor CC. Seguindo a cronologia do desenvolvimento de MRAS, inicia-se com a análise do método chamado de Regra MIT, desenvolvido por Osburn nos laboratórios do MIT, e a seguir, analisa-se o MRAS baseado nas funções de Lyapunov.

## 2.5 - Projeto de MRAS baseado na abordagem do gradiente

### 2.5.1 - Regra MIT

Considere-se o sistema contínuo de primeira ordem descrito pela função de transferência  $G(s) = b/(s+a)$  e pela equação 2.17a. Na equação 2.17a  $u$  é definida como a entrada da planta e  $\Omega$  como a saída medida do sistema [Åstrom & Wittenmark, 1989].

Considere-se o modelo do sistema contínuo de primeira ordem descrito pela função de transferência  $G_m(s) = b_m/(s+a_m)$  e pela equação 2.17b.

$$d\Omega/dt = -a\Omega + bu \quad (2.17a)$$

$$d\Omega_m/dt = -a_m\Omega_m + b_mu_c \quad (2.17b)$$

Na equação 2.17b  $u_c$  é a variável de controle e  $\Omega_m$  a saída calculada pelo modelo. Usando-se o controlador descrito pela equação 2.18 com  $t_0 = b_m/b$  e  $s_0 = (a_m - a)/b$ .

$$u(t) = t_0 u_c(t) - s_0 \Omega(t) \quad (2.18)$$

Define-se o erro do modelo pela equação 2.19.

$$e = \Omega - \Omega_m \quad (2.19)$$

Definindo-se o operador diferencial  $p=d/dt$ , obtem-se a equação 2.20 de saída do sistema.

$$\Omega = \frac{bt_0u_c}{p+a+bs_0} \quad (2.20)$$

Assumindo-se que se deseja modificar os parâmetros  $\theta(t_0, s_0)$  do controlador tal que o erro entre a saída do processo e do modelo de referência seja mínimo, introduz-se o índice de desempenho mostrado na equação 2.21.

$$J(\theta) = \int_0^{\infty} e^2 \quad (2.21)$$

Para tornar  $J$  pequeno é necessário modificar os parâmetros na direção do gradiente negativo de  $J$ , ou:

$$\frac{d\theta}{dt} = -\gamma \frac{\partial J}{\partial \theta} = -\gamma e \frac{\partial e}{\partial \theta} \quad (2.22)$$

Assumindo-se que as variações nos parâmetros são mais lentas que as variações nos parâmetros da planta, pode-se calcular  $\partial e / \partial \theta$  admitindo-se  $\theta$  constante.  $\partial e / \partial \theta$  é chamada de derivada da sensibilidade do erro e a equação 2.23 é chamada de regra MIT.

$$\frac{d\theta}{dt} = -\gamma e \frac{\partial e}{\partial \theta} \quad (2.23)$$

Usando a equação 2.19 e usando a regra MIT, obtem-se as derivadas da sensibilidade do erro mostradas nas equações 2.24 e 2.25.

$$\frac{de}{dt_0} = \frac{bu_c}{p+a+bs_0} \quad (2.24)$$

$$\frac{de}{ds_0} = \frac{-b^2 t_0 u_c}{(p+a+bs_0)^2} = \frac{-b\Omega}{p+a+bs_0} \quad (2.25)$$

Para o cálculo das equações 2.25 necessitam-se dos valores dos parâmetros  $a$  e  $b$  da planta que não são geralmente conhecidos. Considerando-se valores ótimos dos parâmetros do regulador, a equação 2.26 pode ser definida.

$$p+a+bs_0 = p+am \quad (2.26)$$

Usando a regra MIT e definindo duas novas variáveis auxiliares  $x_t$  e  $x_s$ , obtem-se as equações 2.27a e 2.27b.

$$x_I = \frac{u_c}{p + a_m} \quad (2.27a)$$

$$x_S = \frac{\Omega}{p + a_m} \quad (2.27b)$$

Agrupando-se as equações 2.25 e 2.27a e 2.27b, obtêm-se as equações 2.28a e 2.28b para os parâmetros  $t_0$  e  $s_0$ . Conhecendo-se os valores de  $t_0$  e  $s_0$  pode-se usar a equação 2.18 para controle do sistema mostrado na equação 2.17a.

$$dt_0/dt = -\gamma \left[ \frac{u_c}{p + a_m} \right] e = -\gamma \cdot e \cdot x_I \quad (2.28a)$$

$$ds_0/dt = \gamma \left[ \frac{\Omega}{p + a_m} \right] e = \gamma \cdot e \cdot x_S \quad (2.28b)$$

### 2.5.2 - Controle da velocidade do motor CC usando a regra MIT

O ajuste de  $\theta(t_0, s_0)$ , representado pelos parâmetros auxiliares  $t_0$  e  $s_0$ , é implementado pelas equações 2.29a e 2.29b onde o sobrescrito  $T$  indica transposta da matriz.

$$\frac{d\theta}{dt} = \gamma \cdot \phi \cdot e \quad (2.29a)$$

$$\phi = \frac{1}{p + a_m} [-u_c \quad \Omega]^T \quad (2.29b)$$

Pode-se modificar o procedimento de adaptação da regra MIT para que o ajuste independa da magnitude do sinal de comando (normalização). Na equação 2.30 é mostrada a regra MIT normalizada. Na equação 2.30 o parâmetro  $\alpha$  é usado para evitar divisão por zero.

$$\frac{d\theta}{dt} = -\gamma e \frac{\partial e}{\partial \theta} / \left[ \alpha + \left( \frac{\partial e}{\partial \theta} \right)^T \left( \frac{\partial e}{\partial \theta} \right) \right] \quad (2.30)$$

Os parâmetros da função de transferência do motor CC utilizados foram calculados na Seção 1.3 ( $\kappa=5.83$  e  $\tau=0.1943$ ). Para se ter um ganho unitário no modelo de referência, forçou-se  $b_m=a_m$  na equação 2.17b do modelo de referência do motor CC.

No Algoritmo 2.1 [Åström & Wittenmark, 1989] são mostrados detalhes da implementação do programa, escrito na linguagem C, para controle da velocidade do motor CC usando-se a regra MIT normalizada. Usou-se  $\alpha=\text{alfa}=0.0001$ ,  $T_s=0.012s$  e  $\gamma=\text{gama}=0.5$ .

## Algoritmo 2.1

## Controle da velocidade do motor CC usando a regra MIT

a) Calcula a saída do modelo de referência, usa a equação de diferença:

$$\Omega_m = 0.9401 * \Omega_m + 0.0599 * uc;$$

b) Calcula as variáveis auxiliares  $x_t$  e  $x_s$ , usam as equações de diferença:

$$x_t = 0.9401 * x_t + 0.012 * uc;$$

$$x_s = 0.9401 * x_s + 0.012 * \Omega;$$

c) Calcula o erro entre o valor da velocidade do modelo de referência e o valor real da velocidade:

$$e = \Omega_m - \Omega;$$

d) define uma variável auxiliar:

$$den = \alpha + x_t * x_t + x_s * x_s;$$

e) Calcula as variáveis auxiliares  $t_0$  e  $s_0$ , usa as seguintes equações de diferença:

$$t_0 = t_0 - 0.012 * \gamma * e * x_t / den;$$

$$s_0 = s_0 + 0.012 * \gamma * e * x_s / den;$$

f) Calcula o valor a ser usado no controle do motor CC:

$$u = t_0 * uc - s_0 * \Omega;$$

g) Repetem-se as etapas acima.

Na Figura 2.7 são mostradas as curvas de adaptação dos parâmetros  $t_0$  e  $s_0$  obtidas experimentalmente com o controlador MIT. Nessa Figura são mostradas 1200 intervalos de tempo que equivalem a 1200 adaptações dos parâmetros  $t_0$  e  $s_0$ . Cada intervalo de tempo corresponde a 0.012s. A adaptação foi iniciada com  $t_0 = 0$  e  $s_0 = 0$ . Os parâmetros  $t_0$  e  $s_0$  tendem a valores ( $t_0 = 2.1; s_0 = 0.5$ ) que tornam as velocidades de rotação do motor CC igual a velocidade referência ou,  $\Omega(t) = \Omega_m(t)$ .

Na Figura 2.8 são mostrados as curvas de  $\Omega(t)$  e  $\Omega_m(t)$  obtidas durante os 450 intervalos de tempo iniciais da adaptação dos parâmetros  $t_0$  e  $s_0$  da regra MIT. Cada intervalo de tempo corresponde a 0.012s. Observe-se que inicialmente  $\Omega(t)$  é zero. Com a adaptação de  $t_0$  e  $s_0$ , o valor da velocidade de rotação do motor ( $\Omega(t)$ ) tende para a velocidade obtida com o modelo de referência ( $\Omega_m(t)$ ).

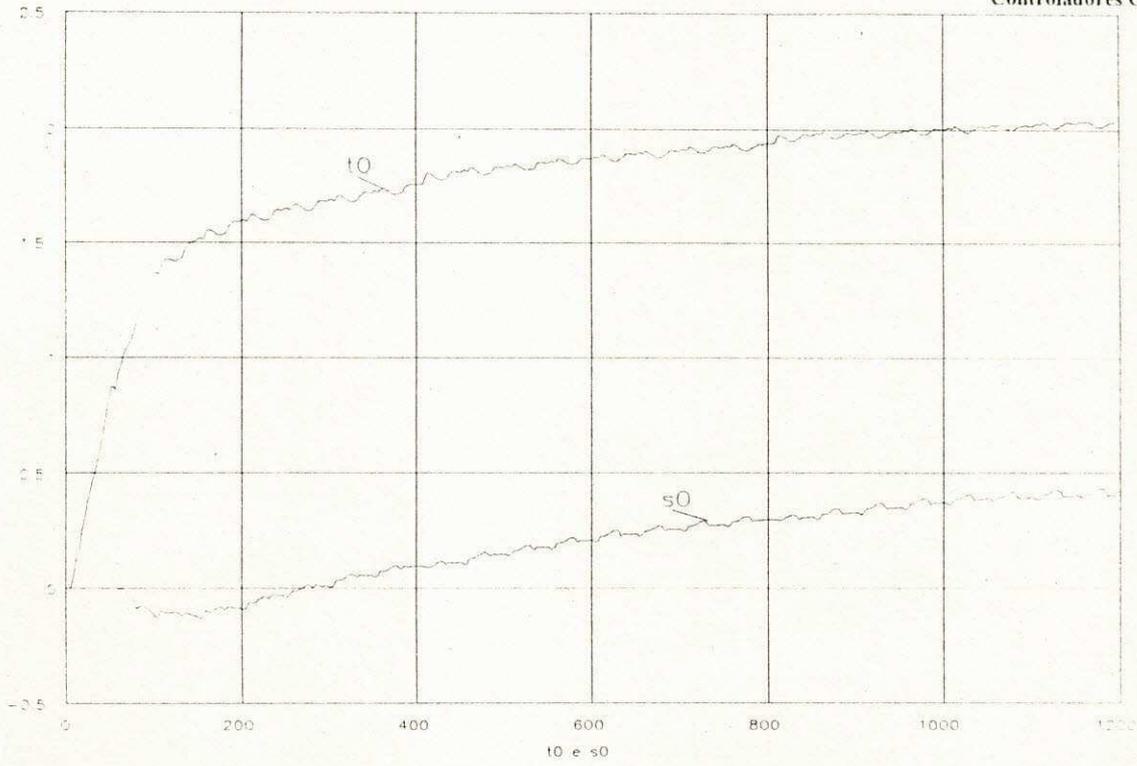


Figura 2.7 Adaptação dos parâmetros  $t_0$  e  $s_0$ .

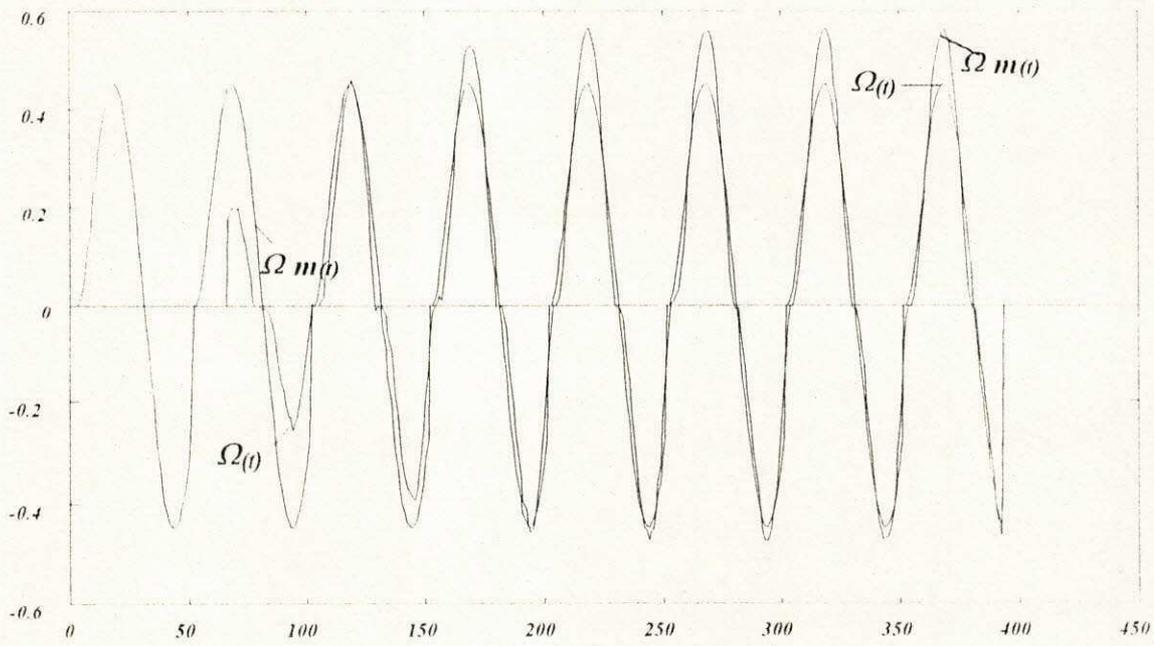


Figura 2.8 - Resultados experimentais obtidos com o controlador MIT.

## 2.6 - Controle adaptativo usando funções de Lyapunov

### 2.6.1 - Projeto de MRAS usando funções de Lyapunov

O projeto de controladores MRAS de primeira ordem utiliza o teorema de estabilidade de Lyapunov que garante a estabilidade do sistema em malha fechada. O projeto do controlador segue a mesma metodologia usada no projeto da regra MIT.

Usando-se a equação do erro do modelo ( $e = \Omega - \Omega_m$ ) e derivando-se obtém-se a equação 2.31.

$$\frac{de}{dt} = \frac{d\Omega}{dt} - \frac{d\Omega_m}{dt} \quad (2.31)$$

Usando-se as equações 2.17a, 2.17b e 2.18, obtém-se a equação 2.32.

$$\frac{de}{dt} = -a_m e + (a_m - a - bs_0)\Omega + (bt_0 - b_m)u_c \quad (2.32)$$

Deve-se construir um mecanismo de ajustamento de parâmetros para conduzir  $t_0$  e  $s_0$  aos valores desejados. Introduce-se a função 2.33 de Lyapunov que é igual a zero quando o erro do modelo é zero.

$$V(e; t_0; s_0) = \frac{1}{2} \left[ e^2 + \frac{1}{b\gamma} (bs_0 + a - a_m)^2 + \frac{1}{b\gamma} (bt_0 - b_m)^2 \right] \quad (2.33)$$

A derivada de V é mostrada na equação 2.34.

$$\frac{dV}{dt} = e \frac{de}{dt} + \frac{1}{\gamma} (bs_0 + a - a_m) \frac{ds_0}{dt} + \frac{1}{\gamma} (bt_0 - b_m) \frac{dt_0}{dt} \quad (2.34)$$

$$\frac{dV}{dt} = -a_m e^2 + \frac{1}{\gamma} (bs_0 + a - a_m) \left( \frac{ds_0}{dt} - \gamma \Omega e \right) + \frac{1}{\gamma} (bt_0 - b_m) \left( \frac{dt_0}{dt} + \gamma u_c e \right) \quad (2.35)$$

Se os parâmetros  $s_0$  e  $t_0$  variam de acordo com as equações 2.36a e 2.36b, obtém-se a equação 2.37 para V que decresce enquanto o erro e for diferente de zero. As equações 2.36a e 2.36b são chamadas de leis de ajustamento e podem ser escritas na forma da equação 2.38.

$$\frac{ds_0}{dt} = \gamma \Omega e \quad (2.36a)$$

$$\frac{dt_0}{dt} = -\gamma u_c e \quad (2.36b)$$

$$\frac{dV}{dt} = -a_m e^2 \quad (2.37)$$

$$\frac{d\theta}{dt} = \gamma \phi \quad (2.38)$$

A equação 2.38 pode ser escrita na forma matricial mostrada na equação 2.39 que pode ser comparada com a do ajustamento dos parâmetros da regra MIT mostrada na equação 2.29.

$$\phi = (-u_c \quad \Omega)^T \quad (2.39)$$

### 2.6.2 - Controle da velocidade do motor CC usando funções de Lyapunov

Os parâmetros da função de transferência do motor CC utilizados foram calculados na Seção 1.3 ( $\kappa=5.83$  e  $\tau=0.1943$ ). Para se ter um ganho unitário no modelo de referência, forçou-se  $b_m=a_m$  na equação do modelo de referência do motor CC.

No Algoritmo 2.2 [Äström & Wittenmark, 1989] são mostrados detalhes da implementação do programa, escrito na linguagem C, para controle da velocidade do motor CC usando funções de Lyapunov. Usou-se  $T_s=0.012s$  e  $\gamma=gama=0.1$ .

#### Algoritmo 2.2

##### Controle da velocidade do motor CC usando funções de Lyapunov

a) Calcula a saída do modelo de referência, usa a equação de diferença:

$$\Omega_n = 0.9401 * \Omega_{n-1} + 0.0599 * u_c;$$

b) Calcula o erro entre o valor da velocidade do modelo de referência e o valor atual da velocidade:

$$e = \Omega_n - \Omega,$$

c) Calcula as variáveis auxiliares  $t_0$  e  $s_0$ , usa as equações de diferença:

$$t_0 = t_0 - 0.012 * gama * e * u_c;$$

$$s_0 = s_0 + 0.012 * gama * e * \Omega,$$

d) Calcula o valor de controle do motor:

$$u = t_0 * u_c - s_0 * \Omega,$$

g) Repetem-se as etapas acima.

Não foram apresentadas as curvas experimentais de adaptação dos parâmetros  $t_0$  e  $s_0$  e das velocidades do motor e referência obtidos com o controlador baseado nas funções de Lyapunov porque elas se assemelham as curvas equivalentes obtidas com o controlador MIT.

## 2.7 - Conclusão

Neste Capítulo foi feita uma descrição sucinta das principais técnicas de controle convencionais, as técnicas do tipo PID e técnicas adaptativas baseadas no modelo de referência. A ênfase foi dada a estratégias de controle adaptativas em que o valor da variável de saída da planta deve seguir um valor de referência.

Inicialmente, apresentamos o projeto e os resultados experimentais obtidos com um controlador convencional do tipo PI e o controle de posição do eixo do motor CC usando um controlador PD. A seguir discutimos os sistemas adaptativos com modelo de referência. Escolheram-se as duas abordagens mais representativas para o desenvolvimento de controladores MRAS: abordagem do gradiente e usando funções de Lyapunov. Usando a abordagem do gradiente, apresentou-se o projeto e os resultados experimentais obtidos com um controlador MRAS denominado de "Regra MIT". Usando funções de Lyapunov, apresentou-se o projeto de um controlador MRAS baseado nessas funções.

No Capítulo 5 será feita uma análise sucinta do desempenho desses controladores convencionais. Serão comparados os desempenhos dos controladores PID com os controladores adaptativos e serão comparados os desempenhos dos controladores convencionais com os controladores neurais. No Capítulo 6 serão analisados os resultados obtidos com o controlador PD de posição do eixo do motor CC aplicado ao controle de um pêndulo invertido.

## CONTROLADORES NEURAIIS ARTIFICIAIS

### 3.1 - Introdução

Neste Capítulo, após uma breve análise dos principais controladores neurais hoje existentes, propõe-se a arquitetura da rede neural artificial a ser usada no controle de processos em tempo real.

Inicialmente apresentam-se sucintamente as características principais do controlador neural e discutem-se alguns detalhes para a implementação desse tipo de controlador. A seguir propõe-se a arquitetura da rede neural como rede neural multi-camadas, com única camada oculta, e usando propagação retroativa como algoritmo de treinamento. Na Seção seguinte, faz-se uma análise, estuda-se e simula-se um sistema para a geração das formas de ondas dos sinais a serem usados no treinamento da rede neural.

### 3.2 - Redes neurais artificiais

Redes neurais artificiais estão sendo empregadas em diferentes áreas científicas. O desenvolvimento das redes neurais remonta ao ano de 1942 como pode ser visto na coletânea de trabalhos comentados e editados por Rosenfeld [1988]. Os modelos de redes neurais

artificiais desenvolvidos recentemente foram aplicados em reconhecimento de padrões, mapeamento de regiões, e controladores. Duas classes de redes neurais artificiais se destacaram nos últimos anos: a) redes neurais multi-camadas, utilizada em problemas de reconhecimento de padrões; b) redes neurais artificiais recorrentes, utilizada em memórias associativas e na resolução de problemas de otimização. As redes neurais artificiais se caracterizam por um conjunto de neurônios interconectados em que os pesos das conexões podem ser modificados baseado num objetivo comum de operação. Isto é, as redes neurais processam informações de uma forma coletiva permitindo uma resposta rápida nas suas saídas, e sob certas condições, as redes neurais podem se adaptar a mudanças nos parâmetros externos à rede. Rumelhart et alii [1986] descreveram um algoritmo para treinamento de RNMC em que o erro na saída é usado retroativamente para a modificação dos pesos da rede (ver Anexo A).

Diversos pesquisadores tem estudado a aplicação de redes neurais multi-camadas na identificação e controle de sistemas não lineares. Cotter [1990] e Hornik [1991], baseados no teorema "Stone-Weierstrass", comprovaram que mesmo com uma única camada oculta de neurônios, redes neurais podem aproximar efetivamente qualquer função contínua. Recentemente Chen & Chen [1993], além de analisarem os resultados obtidos com a aproximação de funções contínuas por redes neurais artificiais multi-camadas, apresentaram teoremas que mostram a capacidade das redes neurais artificiais aproximarem funções contínuas. Baseado na capacidade de aproximação de funções contínuas por RNMC e usando o algoritmo de treinamento retroativo da RNMC, Narendra & Parthasarathy [1990] mostraram, a partir de resultados de simulação, que redes neurais artificiais efetivamente podem ser usadas na identificação e controle de sistemas dinâmicos não lineares.

A operação ou o funcionamento de uma rede neural geralmente passa por duas fases distintas: aprendizagem (ou treinamento) e operação. Na fase de treinamento devem ser conhecidos os padrões de entrada e saída. Na fase de operação, espera-se que a rede neural reconheça os padrões de entrada, isto é, fornecendo-se um padrão de entrada a rede deverá fornecer o respectivo padrão de saída. Além disso, espera-se que a rede neural seja capaz de generalizar, podendo existir uma fase de adaptação durante a fase de operação da rede neural, que no caso do controlador neural, serve para a sua sintonização fina.

Um controlador neural executa uma forma específica de controle adaptativo, tomando a forma de uma rede multi-camadas em que os parâmetros adaptáveis são as intensidades das interconexões entre os neurônios artificiais da rede neural, e os parâmetros  $\theta$ ,  $\beta$  e  $T$  dos neurônios [Rangwala & Dornfeld, 1989]. Diversas estruturas de controladores neurais foram propostas e implementadas [Tanomaru & Omatu, 1991] [Narendra & Parthasarathy, 1990]. Essas estruturas baseiam-se principalmente na determinação da dinâmica (a RNMC é

denominada de EMULADOR) ou inversa da dinâmica (a RNMC é denominada de INVERSA) da planta a ser controlada. Representando a planta como  $Y(t+1)=f(U(t),Y(t))$  onde  $U(t)$  é a variável de controle e  $Y(t)$  é a saída da planta, mostra-se na Figura 3.1 o diagrama de blocos da estrutura de treinamento das redes neurais INVERSA (saída da RNMC  $U_i(t)$ ) treinada com os valores  $U(t)$ , e EMULADOR (saída da RNMC  $Y_e(t)$ ) treinada com os valores  $Y(t)$ . Na Figura 3.1 APR significa algoritmo de propagação retroativa.

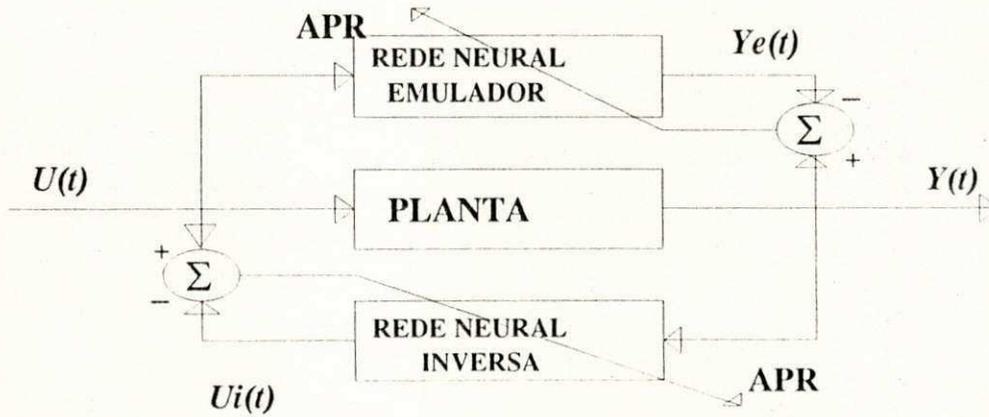


Figura 3.1 Determinação da dinâmica de uma planta usando RNMC.

### 3.3 - Arquitetura da Rede Neural Multi-Camadas

Na Figura 3.2 é mostrado o diagrama em blocos de uma RNMC aqui proposta para a determinação da dinâmica de uma planta. A arquitetura da RNMC da Figura 3.2 é derivada da arquitetura proposta por Yamada & Yabuta [1990]. Neste trabalho, o projeto da RNMC da Figura 3.2 foi feito baseado nas suas características estruturais descritas nas seções a seguir.

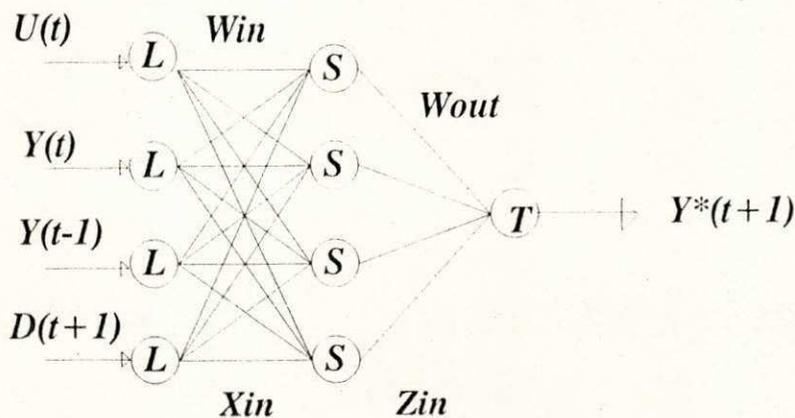


Figura 3.2 RNMC usada como EMULADOR.

### 3.3.1 - Número de camadas da RNMC

Os pesquisadores Villers & Barnard [1993] investigaram o desempenho de uma RNMC, aplicada a classificação de padrões, em função do número das suas camadas ocultas. Eles concluíram que se uma RNMC com um única camada oculta satisfaz às especificações da aplicação, a adição de uma segunda camada oculta implica num pequeno ganho na performance do sistema neural. Sontag [1992], estudando a otimização global de sistemas não lineares, mostrou que o número necessário de camadas ocultas da RNMC depende da aproximação necessária da dinâmica inversa da planta. Neste Trabalho, o número de camadas da rede neural foi escolhido de acordo com o teorema de "Stone-Weierstrass" [Cotter, 1990][Hornick, 1991], isto é, admitiu-se que uma RNMC com única camada oculta pode ser a indicada para o controle.

### 3.3.2 - Número de neurônios na camada oculta da RNMC

A seguir, considerou-se que com uma única camada oculta é possível o controle neural da planta e se procurou calcular o número ótimo de neurônios na camada oculta da RNMC. Diversos pesquisadores obtiveram resultados no cálculo do número ótimo de neurônios na camada oculta da RNMC. Huang & Huang [1991] propuseram um método para calcular o número máximo de neurônios na camada oculta da RNMC. O método baseia-se no número de elementos que determinam o objetivo a ser alcançado com a RNMC. Azimi et alii [1993] empregaram a declividade da curva do erro médio quadrado como um critério para a adição de um novo neurônio na camada oculta da RNMC. Por enquanto, não existe nenhuma maneira de se determinar a priori o número ótimo de elementos da camada oculta da RNMC. Por exemplo, Fogel [1991] propôs um critério, baseado no critério de informação de Akaike (AIC) [Ljung, 1987], para dimensionar o número ótimo de neurônios na camada oculta da RNMC. No Capítulo 6 deste Trabalho será apresentada uma metodologia que permite a determinação do número ótimo de neurônios na camada oculta da RNMC.

### 3.3.3 - Modelos dos neurônios da RNMC

A RNMC da Figura 3.2 é composta de três camadas. O modelo do neurônio da camada de entrada da RNMC é linear (L) e é usado para converter o sinal de entrada para o sistema p.u.. A primeira camada gera sinais normalizados entre -1 e +1 e se constitui basicamente num ponto de distribuição dos sinais de entrada para a camada interna da RNMC. Os neurônios da camada interna são não lineares do tipo sigmóide [ $S(Xin)$ ], mostrado na equação 3.1, e são conectados a cada neurônio da camada de entrada via pesos Win. Alguns pesquisadores usam os parâmetros dos neurônios  $\theta$ ,  $\beta$  e  $T$  constantes. Os pesquisadores Rangwala & Dornfeld [1989] e Cavalcanti et alii [1992b] observaram que a modificação em tempo real desses parâmetros, permite a redução do número de neurônios nas camadas ocultas da RNMC. O neurônio da camada de saída é do

tipo função tangente hiperbólica, mostrado na equação 3.2, com  $T(S(Zin))=2*S(Zin)/\Theta-1$ , e pode variar entre -1 e +1. Os neurônios da camada de saída são conectados aos neurônios da camada interna via os pesos  $Wout$ .

$$S(Xin) = \frac{\Theta}{(1 + e^{-\beta(\sum X_i W_{ij} + T)})} \quad (3.1)$$

$$T[S(Zin)] = \frac{(1 - e^{-\beta(\sum Zin Wout + T)})}{(1 + e^{-\beta(\sum Zin Wout + T)})} \quad (3.2)$$

Usa-se o algoritmo de propagação retroativa para modificação dos pesos  $Win$ ,  $Wout$  e os parâmetros  $\beta$ ,  $T$  e  $\Theta$  dos neurônios da RNMC. No Anexo A são mostradas as equações para ajuste dos parâmetros da RNMC e o programa, escrito na linguagem C, desenvolvido para calcular o algoritmo de propagação retroativa.

### 3.3.4 - Número de entradas e saídas da RNMC

O número de entradas e saídas da rede neural pode ser projetado a partir de uma estimativa da ordem da planta a ser representada pela rede [Yamada & Yabuta, 1990]. Por exemplo, para uma planta SISO (única entrada e única saída), pode-se usar a representação  $AY(t) = BU(t)$  onde  $A$  e  $B$  são polinômios,  $U(t)$  a entrada e  $Y(t)$  a saída da planta. Os números dos neurônios da entrada e da saída da RNMC podem ser determinados a partir da ordem dos polinômios  $A$  e  $B$ .

Observe-se que a RNMC da Figura 3.2 têm os valores  $U(t)$ ,  $Y(t)$ ,  $Y(t-1)$  e  $D(t+1)$  como entradas e, como ela é usada como EMULADOR, ela tem  $Y^*(t+1)$  como valor de saída. A entrada  $D(t+1)$  representa o valor desejado para a saída  $Y(t+1)$  da planta. O valor  $Y^*(t+1)$  fornecido na saída da planta, representa o valor calculado pela RNMC para a saída  $Y(t+1)$  da planta. O treinamento da RNMC é feito no intuito de tornar o menor possível o quadrado do erro definido como  $E^2 = [Y(t+1) - Y^*(t+1)]^2$ .

### 3.3.5 - Sinais usados no treinamento da RNMC

O treinamento da RNMC pode ser feito "off line" e "on line". Para qualquer um dos dois tipos de treinamento deve-se considerar o conjunto dos dados representativos da dinâmica da planta. O conceito de entradas com excitação persistente, que é bem conhecido na área de controle adaptativo [Åström & Wittenmark, 1989], foi definido no campo das redes neurais pelos pesquisadores Polycarpou & Ioannou [1992] para analisar a convergência da aprendizagem de redes neurais. Hoskins et alii [1992] propuseram uma RNMC para um controlador adaptativo usando modelo referência e estudaram a necessidade de um sinal "dither" para determinação em tempo real da dinâmica da planta. Cavalcanti et alii [1992b] também propuseram o emprego de

um sinal "dither" para sintonização de uma rede neural em tempo real com a dinâmica inversa de uma planta (ver a Seção 4.5 deste trabalho).

### 3.3.6 - Número de iterações usadas no treinamento e as condições iniciais

Um grande número de iterações é necessário na fase de treinamento da RNMC. A fase de treinamento da RNMC é geralmente executada "off line". Após a fase de treinamento, durante a fase de operação "on line", a RNMC deve ser treinada para uma sintonização fina com a dinâmica da planta. O treinamento "off line" da RNMC pode ser feita usando um simulador de operação da planta.

A seguir apresentam-se os resultados obtidos, usando simuladores e com a RNMC da Figura 3.3 aplicadas ao controle de um motor CC (modelo linear). Os programas usados na simulação do motor CC e na implementação da rede neural artificial multi-camadas foram desenvolvidos por Cavalcanti et alii [1991a,b][1992] são mostrados resumidamente no Anexo A.

### 3.4 - Treinamento "off line" da RNMC com a dinâmica da planta

Para o treinamento da RNMC, mostrada na Figura 3.3, com a dinâmica da planta, necessita-se do conjunto de dados de entrada e saída do sistema motor CC. Implementou-se um simulador, apresentado no Algoritmo A.2 do Anexo A, usando o algoritmo de Runge-Kutta de quarta ordem como integrador, para o sistema motor CC. Usando o simulador e com os parâmetros do motor CC apresentados na Tabela 1.1, foram calculados dois conjuntos de dados: um conjunto de dados gerado na entrada e outro conjunto de dados obtido na saída do sistema motor CC.

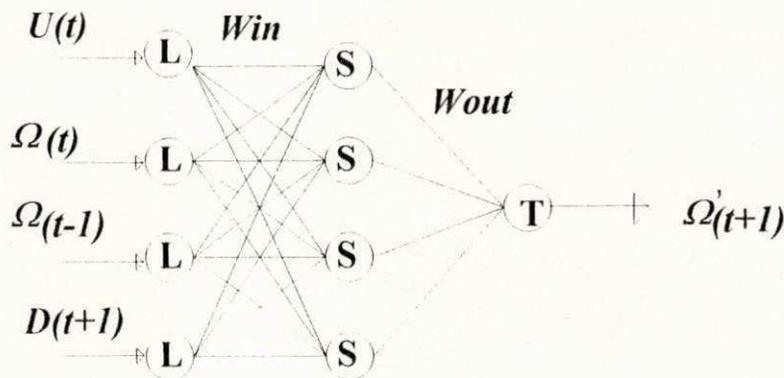


Figura 3.3 RNMC usada como EMULADOR para o motor CC.

A geração dos conjuntos dos dados de entrada da RNMC foi feito diretamente aplicando um sinal na entrada  $U(t)$  da planta. Foram usados dois tipos de sinais como entrada (tensão de armadura  $U(t)$ ) do sistema motor CC. O primeiro tipo usou uma forma de onda senoidal

(denominado excitação senoidal) e o segundo tipo usou sinais pseudo-aleatórios com distribuição uniforme (denominado excitação aleatória) como entradas  $U(t)$  para controle da tensão de armadura do motor CC.

Na Figura 3.4a é mostrado um intervalo das formas de onda obtidas na simulação e no controle do motor CC usando excitação senoidal. Nessa figura são mostradas as formas de onda normalizadas de um intervalo da tensão de armadura  $U(t)$ , corrente de armadura  $I_a(t)$ , velocidade angular  $\Omega(t)$  do motor CC, e a velocidade angular  $\Omega'(t)$  estimada pela RNMC. A forma de onda da tensão de armadura consistiu de 60 sinais PWM gerados de uma forma senoidal a cada intervalo de 12 mS ( $T_c = 8T_\phi$ ). Somente os 60 valores dos sinais PWM e velocidade angular foram utilizados para treinamento da rede neural. Observe-se que a forma senoidal da tensão da armadura gera formas de onda simétricas de corrente e velocidade no motor CC.

A RNMC, mostrada na Figura 3.3, é treinada com a apresentação de 1000 intervalos das formas de onda de  $U(t)$  e  $\Omega(t)$  mostradas na Figura 3.4a. Usou-se  $\eta = 0.1$  como valor do fator de convergência do algoritmo de propagação retroativa. Logo após o treinamento com excitação senoidal, foi feita a verificação da capacidade de generalização da RNMC usando excitação aleatória. Isto é, verifica-se a capacidade de interpolação da RNMC. Na Figura 3.4b são mostradas as curvas de  $\Omega(t)$  e  $\Omega'(t)$  obtidas com excitação aleatória como entradas  $U(t)$  para controle da tensão de armadura do motor CC.

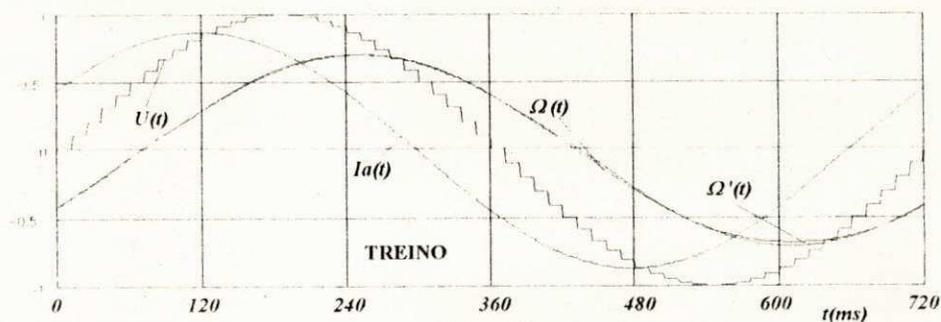


Figura 3.4a

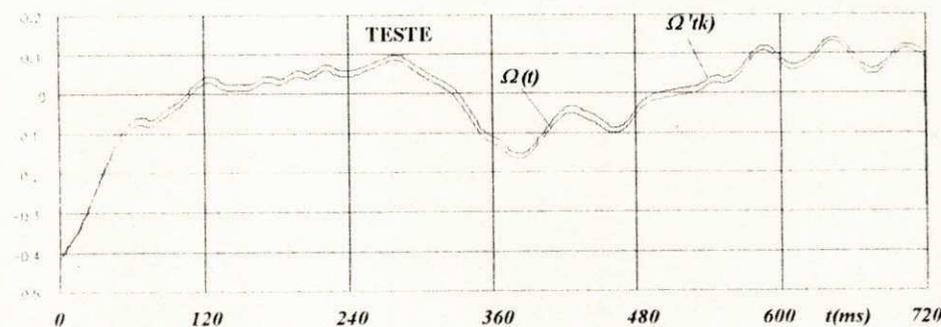


Figura 3.4b

Figura 3.4 Treinamento da RNMC com excitação senoidal em  $U(t)$ .

Na Figura 3.5a é mostrado um intervalo das formas de onda obtidas na simulação do motor CC usando excitação aleatória. Nessa figura são mostradas as formas de onda normalizadas de um intervalo da tensão de armadura  $U(t)$ , corrente de armadura  $Ia(t)$ , velocidade angular  $\Omega(t)$  do motor CC, e a velocidade angular  $\hat{\Omega}(t)$  estimada pela RNMC. A forma de onda da tensão de armadura consistiu de 60 sinais PWM gerados de uma forma aleatória e uniformemente distribuído a cada intervalo de 12 mS ( $T_c = 8T_s$ ).

A RNMC, mostrada na Figura 3.3, é treinada com a apresentação de 1000 intervalos das formas de onda de  $U(t)$  e  $\Omega(t)$  mostradas na Figura 3.5a. Usou-se  $\eta = 0.1$  como valor do fator de convergência do algoritmo de propagação retroativa. Logo após o treinamento com excitação aleatória foi feita a verificação da capacidade de generalização da RNMC usando excitação senoidal. Isto é, verifica-se a capacidade de extrapolação da RNMC. Na Figura 3.5b são mostradas as curvas de  $\Omega(t)$  e  $\hat{\Omega}(t)$  obtidas com excitação senoidal como entradas  $U(t)$  para controle da tensão de armadura do motor CC.

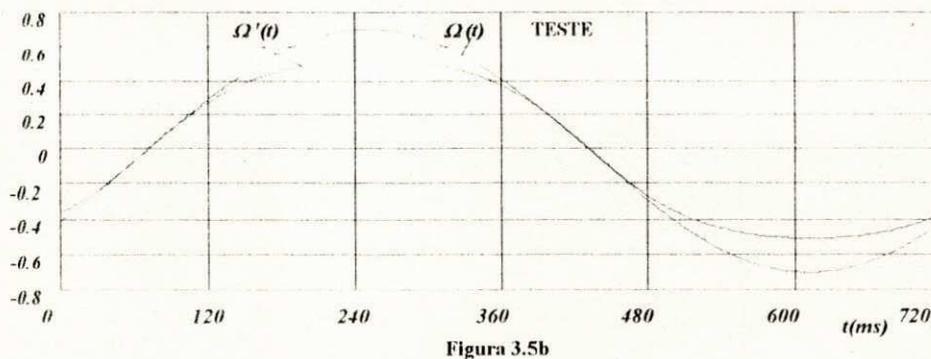
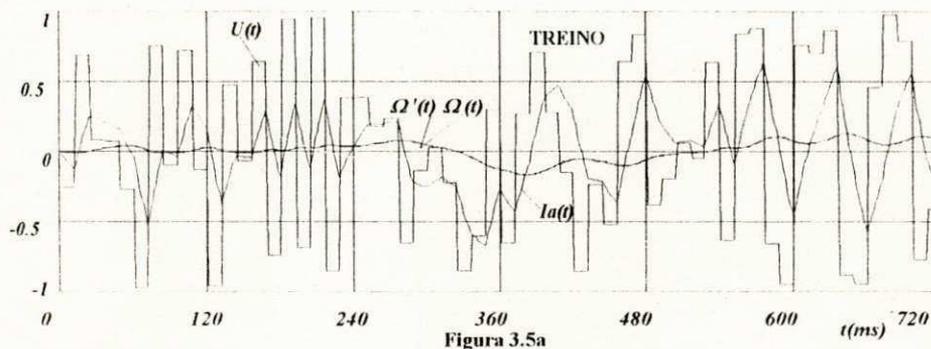


Figura 3.5 Treinamento da RNMC com excitação aleatória em  $U(t)$ .

### 3.5 - Análise do treinamento "off line" da RNMC com a dinâmica do motor CC

A partir dos resultados obtidos no treinamento "off line" da RNMC concluiu-se o seguinte:

a) o treinamento repetitivo da rede neural com formas de onda simétricas obtidas com excitação senoidal, e com a escolha de um valor conveniente para o fator de convergência usado no algoritmo de propagação retroativa, a saída  $\Omega'(t)$  da RNMC será também simétrica (ver a Figura 3.4a);

b) a RNMC treinada com excitação senoidal foi capaz de identificar as saídas geradas com excitação aleatória, isto é, a RNMC foi capaz de identificar valores interpolados aos do treinamento (ver a Figura 3.4b);

c) a RNMC treinada com excitação aleatória não foi capaz de identificar satisfatoriamente as saídas geradas com excitação senoidal. O sinal de excitação aleatória gerou um sinal  $\Omega(t)$  variando no intervalo entre -0.6 e 0.6. Na parte inferior da Figura 3.4 vê-se que a RNMC não foi capaz de extrapolar. Isto é, o erro cresce quando os valores de  $\Omega(t)$  e  $\Omega'(t)$  se aproximam de 1 (ver a Figura 3.5b).

Durante o treinamento da RNMC como EMULADOR, usou-se como índice de desempenho, mostrado na equação 3.3, a somatória do erro ao quadrado ocorrido durante um intervalo do treinamento. O erro foi definido como a diferença entre o valor da velocidade do eixo do motor CC e o valor da saída da RNMC mostrada na Figura 3.3.

$$\sum_1^{60} E_k^2 = \sum_k |\Omega(k) - \Omega'(k)|^2 \quad (3.3)$$

A cada intervalo são aplicados, na entrada da RNMC, os 60 valores de  $U(t)$ ,  $\Omega(t)$ ,  $\Omega(t-1)$  e  $D(t+1)$ , e são usados  $\Omega(t)$  e  $\Omega'(t)$  para cálculo do erro a ser empregado no algoritmo de propagação retroativa descrito no Anexo A. Na Figura 3.6 é mostrada uma curva típica do índice de desempenho obtida no treinamento da RNMC. Nessa figura, a ordenada representa o índice de desempenho mostrado na equação (3.3), e a abscissa representa o número de intervalos utilizados para o treinamento da RNMC. Vê-se nessa Figura que houve uma rápida aprendizagem da dinâmica da planta pela RNMC. Os mesmos testes foram feitos para a determinação da INVERSA pela RNMC e se obteve resultados semelhantes aos aqui descritos para a RNMC EMULADOR.

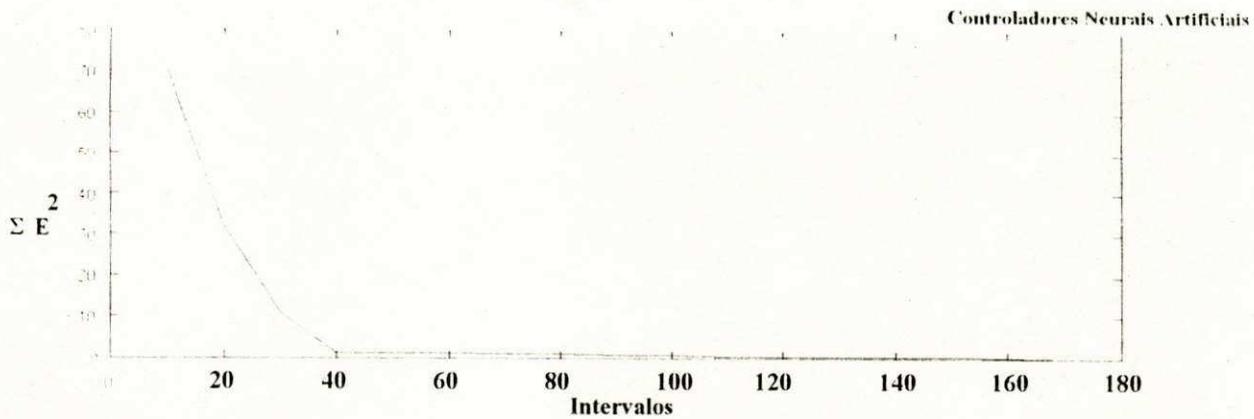


Figura 3.6 Índice de desempenho do treinamento da RNMC.

### 3.6 - Conclusão

Apresentou-se a arquitetura de redes neurais artificiais multi-camadas para controle de plantas lineares. Mostraram-se resultados obtidos com a simulação de sistemas motor CC excitados com sinais senoidais e sinais aleatórios. A partir da observação desses resultados, concluiu-se heurísticamente que a aprendizagem da RNMC depende diretamente da simetria das formas de onda usadas no seu treinamento. Sugere-se que o treinamento "off line" da RNMC deve ser feito inicialmente com excitação aleatória e, para um ajuste fino, deve-se treinar a RNMC com sinais senoidais. Os resultados obtidos neste Capítulo serão usados, no Capítulo 6, para a especificação da RNMC do controlador neural direto adaptativo.



## CONTROLADOR NEURAL ADAPTATIVO

### 4.1 - Introdução

Neste Capítulo, após uma sucinta descrição dos controladores neurais baseados na inversa da dinâmica da planta, apresentam-se a análise e a implementação experimental de diferentes tipos de controladores neurais adaptativos usando a arquitetura da RNMC proposta e estudada no Capítulo 3 desta Tese.

Inicialmente, descrevem-se os controladores neurais usando a dinâmica inversa da planta, mostra-se o procedimento de treinamento "off line" da RNMC INVERSA, e apresenta-se um algoritmo heurístico para sintonização "on line" da RNMC após a variação da dinâmica da planta. A seguir apresenta-se o controlador neural adaptativo direto que necessita do conhecimento do jacobiano da planta para adaptar, em tempo real, a RNMC controladora. Usando uma RNMC para calcular o jacobiano da planta e uma outra RNMC para o controle, apresenta-se o controlador neural adaptativo indireto. Apresentam-se o projeto e resultados experimentais de um controlador neural adaptativo baseado na formulação de funções não lineares, e se mostra que é possível, sob certas condições, desenvolver um controlador neural adaptativo direto para controle de uma planta linear.

## 4.2 - Controladores neurais usando a dinâmica inversa da planta

Recentemente, Hunt et alii [1992] publicaram uma coletânea dos principais trabalhos sobre redes neurais usando uma perspectiva de sistemas de controle. Eles citam que diversos pesquisadores, principalmente Narendra & Parthasarathy [1990] e Tanomaru & Omatu [1991], têm utilizado diferentes estruturas de controladores neurais baseados na inversa da dinâmica da planta que pode ser calculada para plantas lineares e não lineares.

Considerando-se a planta não linear, e que ela pode ser descrita pela equação  $Y(t+1) = f(Y(t), \dots, Y(t-n+1); U(t), \dots, U(t-n+1))$ , com única entrada e única saída, e que  $f(\cdot)$  representa uma função não linear, se a planta é inversível, deve existir uma função  $g(\cdot)$  tal que,  $U(t) = g(Y(t+1), \dots, Y(t-n+1); U(t-1), \dots, U(t-n+1))$ . Essa função  $g(\cdot)$  é conhecida como a função da dinâmica inversa da planta.

Uma planta linear pode ser representada, na forma discreta, pela equação 4.1, sendo  $U(t)$  a variável de controle e  $Y(t)$  a saída da planta. O valor de saída da planta no  $(t+n)$ -ésimo instante é mostrado na equação 4.2 e a matriz de controlabilidade  $W$  é descrita pela equação 4.3. O sinal de controle é calculado pela equação 4.4 que descreve a inversa da dinâmica da planta. A RNMC será treinada com a dinâmica inversa da planta representada pela equação 4.4.

$$Y(t+1) = f(U(t), Y(t)) = AY(t) + BU(t) \quad (4.1)$$

$$Y(t+n) = A^n Y(t) + WU(t) \quad (4.2)$$

$$W = [B \ AB \ \dots \ A^{n-1}B] \quad (4.3)$$

$$U(t+n) = W^{-1}[Y(t+n) - A^n Y(t)] \quad (4.4)$$

Desprezando os efeitos não lineares mencionados na Seção 1.2, o motor CC pode ser modelado como um sistema linear e a sua dinâmica pode ser descrita pela equação 1.4b que é equivalente à equação 4.1. A seguir serão mostrados os resultados experimentais obtidos no treinamento da RNMC com a inversa da dinâmica do motor CC.

## 4.3 - Dados experimentais do motor CC

No Capítulo 3 deste Trabalho mostrou-se que se deve usar sinais de entrada e saída simétricos para treinamento da RNMC com a dinâmica da planta. Alguns pesquisadores desenvolveram controladores neurais, baseados na dinâmica inversa da planta, em que foi necessário o treinamento "off line" da RNMC usando dados experimentais. Isso foi feito por Khalid & Omatu [1992] para um sistema de controle neural de temperatura, e por Cavalcanti et alii [1992b] para um sistema de controle neural da velocidade de um motor CC.

Neste Capítulo, o conjunto dos sinais simétricos experimentais de entrada e saída do motor CC, obtidos com o sistema de controle mostrado na Figura 1.1, são usados para treinar "off line" a RNMC com a inversa da dinâmica do motor CC.

Na figura 4.1a são mostradas as curvas de um período das formas de onda do motor CC obtidas com 200 sinais para o vetor de controle do inversor ( $U(t)$  na Figura 4.1). A cada intervalo de 1.5 ms ( $T_s = 1.5ms$ ) foi enviado um sinal PWM ao circuito chaveador da armadura e adquirido o valor da corrente de armadura ( $Ia(t)$ ) e da posição angular do eixo do motor CC. A cada intervalo de 12 ms ( $T_c = 8T_s$ ) foi gerado um novo sinal PWM e calculada a velocidade angular  $\Omega(t)$ . Somente os 200 valores dos sinais PWM da tensão da armadura ( $U(t)$ ) e da velocidade angular ( $\Omega(t)$ ) foram utilizados para treinamento da rede neural. Na Figura 4.1b é mostrado o "zoom" da Figura 4.1a obtido no intervalo de 40ms.

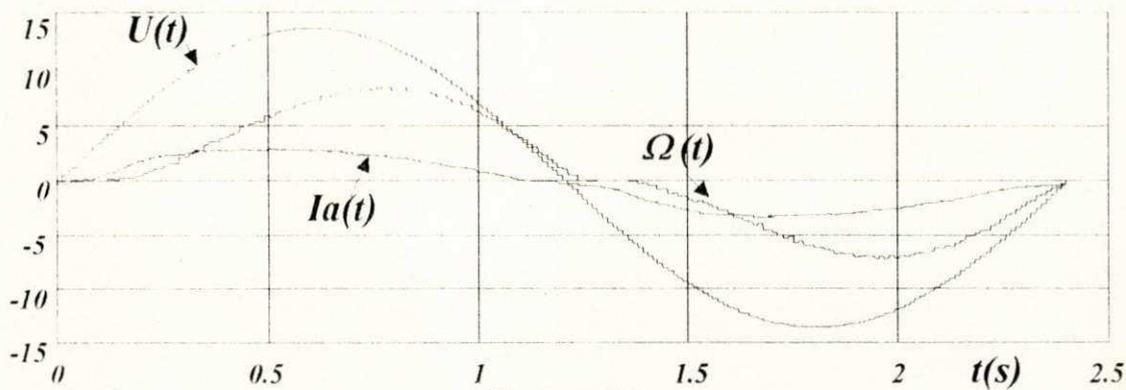


Figura 4.1a

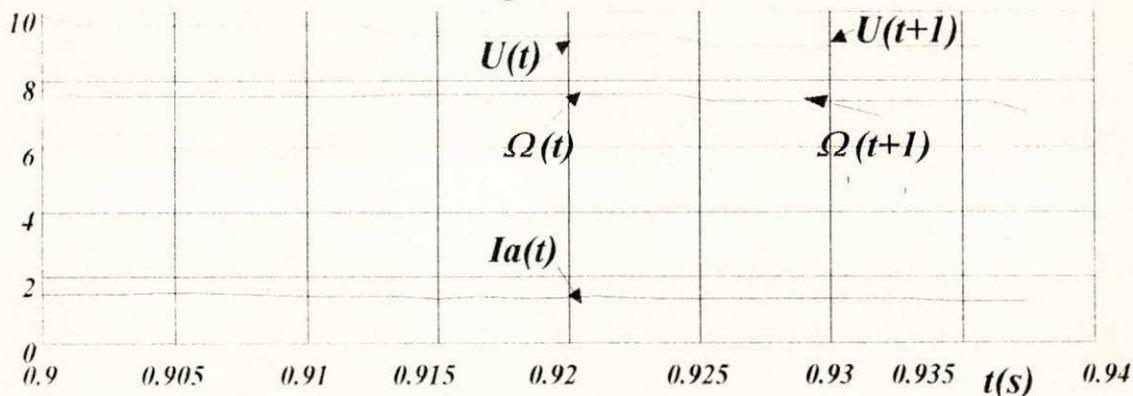


Figura 4.1b

Figura 4.1 - Resultados experimentais do motor CC.

As curvas experimentais  $Ia(t)$  e  $\Omega(t)$  do motor CC, mostradas na figura 4.1, além de estarem com ruído não são simétricas. A não simetria dessas formas de onda ocorre devido a saturação, fricção estática no eixo do motor CC, e ao efeito do tempo morto no inversor PWM [Kuo, 1985]. O ruído observado nos sinais da Figura 4.1 é gerado pelos conversores A/D e detetor de posição do rotor do motor CC. Durante os testes experimentais para controle da

velocidade do motor CC, observou-se que o ruído [Saud, 1989] e a não simetria existente nas curvas experimentais do motor CC aumenta o número de iterações necessárias ao treinamento satisfatório da RNMC controladora.

#### 4.4 - Controlador neural direto inverso

Na Figura 4.2 é mostrado o esquema geral do controlador neural direto inverso como descrito por Tanomaru & Omatu [1991] e Khalid & Omatu [1992]. Ele se caracteriza por ser um sistema aberto e o seu treinamento deve ser feito "off line". O controlador neural direto inverso passa por duas fases: a fase de treinamento "off line" e a fase de operação ou controle, "on line".

O controlador neural direto inverso, mostrado na Figura 4.2 durante a fase de controle e com a PLANTA representando um motor CC, tem como entradas [ $U(t-1)$ ,  $\Omega(t-1)$ ,  $\Omega(t)$  e  $D(t+1)$ ], com  $D(t+1)$  representando a velocidade desejada. Ele gera a saída  $U(t)$  que é usada para controlar a velocidade do motor CC. Antes da fase de controle a RNMC deve ser treinada "off line" com a dinâmica inversa do motor CC.

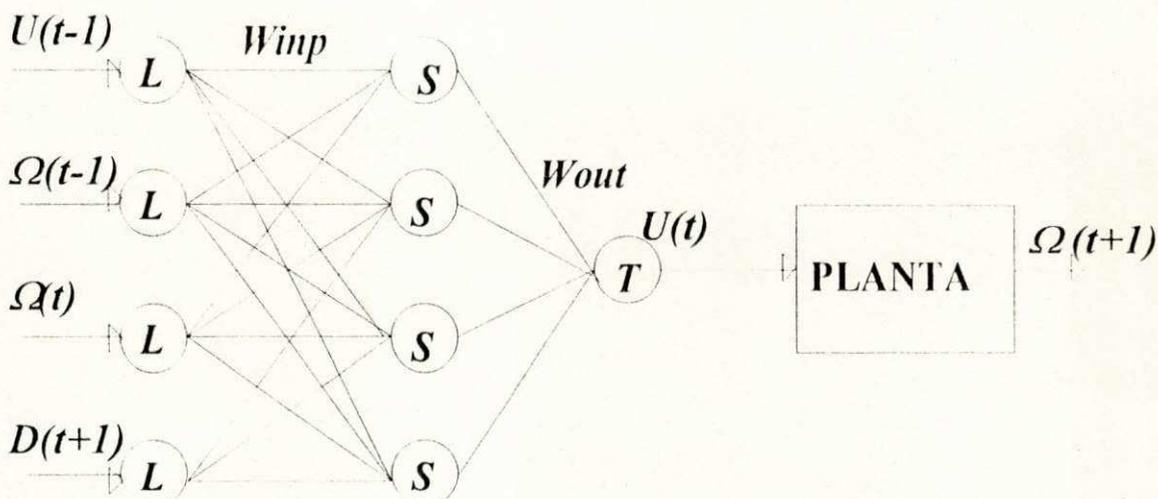


Figura 4.2 - Fase de controle do controlador neural direto inverso.

A RNMC do controlador neural direto inverso, mostrado na Figura 4.3 durante a fase de treinamento, foi treinada "off line" com a apresentação de 1000 períodos dos sinais experimentais de entrada e saída do motor CC,  $U(t)$  e  $\Omega(t)$ , mostrados na Figura 4.1 [Cavalcanti et alii, 1992b]. Na Figura 4.3,  $V(t)$  representa o valor da saída da RNMC. Usou-se a equação 4.5 como índice de desempenho para treinamento dos pesos e parâmetros da RNMC. Usou-se o fator de convergência  $\eta = 0.1$  no algoritmo de propagação retroativa. No Anexo A é mostrado a implementação, na linguagem C, do algoritmo de propagação retroativa.

$$J(t+1) = \frac{1}{2}e(t+1)^2 = \frac{1}{2}[U(t)-V(t)]^2 \quad (4.5)$$

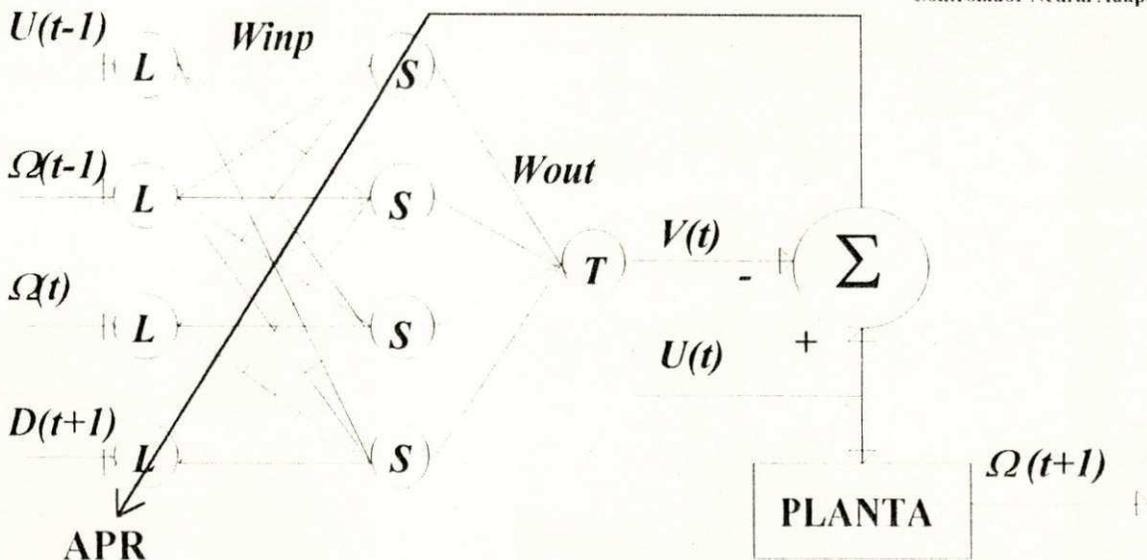


Figura 4.3 - Fase de treinamento "off-line" do controlador neural direto inverso.

Após o treinamento "off line" da RNMC, o desempenho do controlador neural direto inverso foi verificado experimentalmente "on line" usando o sistema de controle do motor CC, apresentado na Figura 1.1. Na figura 4.4 são mostradas as curvas experimentais obtidas com o controlador neural direto inverso. Nessa Figura a corrente  $I_a(t)$  está representada em Ampères, a velocidade referência  $D(t+1)$  e a velocidade do eixo do motor  $\Omega(t)$  estão representados em rd/s, e a tensão de armadura  $U(t)$  está representado em Volts.

Observe, na Figura 4.4, que a velocidade  $\Omega(t)$  do motor CC segue a velocidade de referência  $D(t)$ , e que a rede neural "aprendeu" corretamente a dinâmica inversa do motor CC (diz-se que a RNMC está sintonizada com a dinâmica do motor CC). Havendo uma mudança da dinâmica do sistema motor CC, aumentará o erro existente entre  $D(t)$  e  $\Omega(t)$ , diminuirá o desempenho do controlador neural direto inverso, e a RNMC sairá de sintonia. Na Figura 4.6, no intervalo de tempo entre N1 e N3, são mostrados os resultados experimentais do motor CC quando a RNMC do controlador está fora de sintonia.

Na Seção a seguir será mostrada uma configuração e um algoritmo, baseados no controlador neural direto inverso, capaz de treinar "on line" a RNMC para compensar pequenas variações na dinâmica do motor CC.

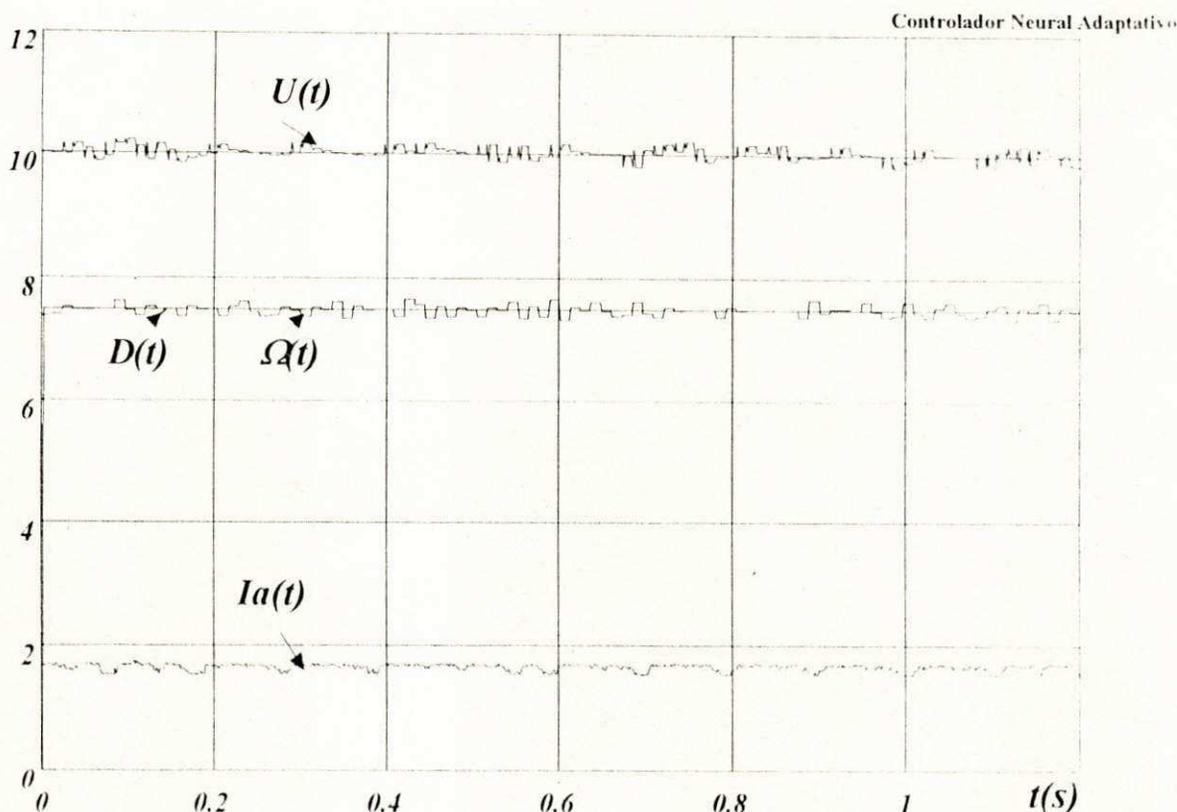


Figura 4.4 - Resultados experimentais obtidos com o controlador neural direto inverso.

#### 4.5 - Treinamento "on line" da RNMC do controlador neural direto inverso

Vários esquemas de controladores adaptativos digitais tem sido propostos nas últimas décadas [Åström & Hägglund, 1989]. O projeto e configuração desses controladores dependem primariamente de um conhecimento antecipado da dinâmica da planta sob controle e, para sintonização do controlador em tempo real, requer-se a determinação "on line" da dinâmica de operação da planta. Uma alternativa para a sintonização automática de um controlador tipo PID foi proposta por Åström & Wittenmark [1989] usando o chaveamento alternado entre o controlador PID e um controlador relé na malha de realimentação. Um ciclo limite ocorre quando o controlador relé está conectado e os parâmetros dessa oscilação são usados para adaptar os parâmetros do controlador PID. Outros pesquisadores, Hang & Sin [1991] propuseram uma versão de um algoritmo "on line" para auto-sintonização de controladores PID baseado em técnicas de correlação cruzadas e que não perturbam a operação normal do processo.

Cavalcanti et alii [1992b] propuseram para o controlador neural direto inverso uma configuração semelhante à proposta por Åström & Wittenmark [1989] para sintonização do controlador PID. Cavalcanti et alii [1992b] usaram a configuração mostrada na Figura 4.5 que funciona em conjunto com o Algoritmo 4.1.

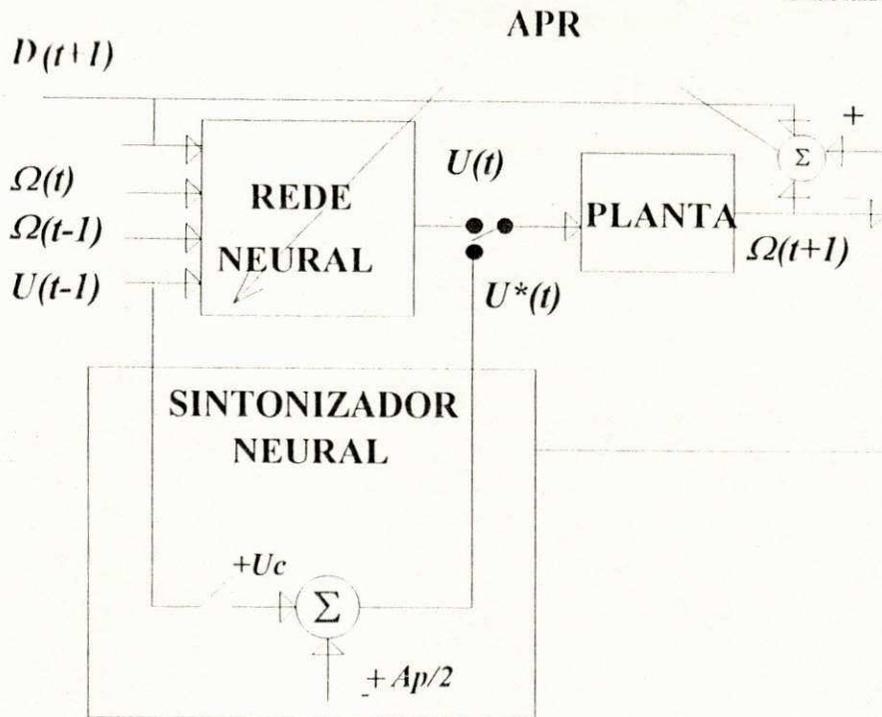


Figura 4.5 - Configuração do controlador neural direto inverso.

No Algoritmo 4.1, mostrado abaixo, é assumido que houve uma variação nos parâmetros do motor CC ou na sua carga, o desempenho do controlador se deteriorou e a RNMC saiu de sintonia (ver Figura 4.6). Inicialmente, o controlador neural está sintonizado com a dinâmica do motor CC como pode ser visto nas curvas experimentais do motor CC mostradas na Figura 4.4. Havendo a não sintonização da RNMC, que pode ser devida a uma variação na carga ou na dinâmica do motor CC, ocorrerá um erro entre a velocidade atual e a velocidade referência ( $ERRO = D(t) - \Omega(t)$ ). O  $ERRO$  pode ser visto na Figura 4.6, nos intervalos entre ( $N1$  e  $N2$ ), ( $N3$  e  $N4$ ), e ( $N5$  e  $N6$ ).

No Algoritmo 4.1, a mudança para o modo de treinamento da RNMC ocorrerá quando o somatório do erro ao quadrado for maior do que um valor pré-definido ( $\Sigma ERRO^2 > E_{max}$ , 100 valores do  $ERRO$  e  $E_{max} = 1$ ). No Algoritmo 4.1 os valores de  $Np$  ( $Np = 15$ , número de pulsos),  $Ap$  ( $Ap = IV$ , amplitude do pulso) foram escolhidos heurísticamente.

## Algoritmo 4.1

## Treinamento em tempo real da RNMC com a nova dinâmica da planta

- 1) Em  $t=N1$ , com  $\Sigma ERRO^2 > Emax$ , a saída da rede neural é congelada,  $U_c=U(t)$ .
- 2) Em  $N1 < t < N2$   $N_p$  pulsos com amplitude  $Ap/2$  são superpostos a  $U_c$  para gerar  $U^*(t) = U_c + Ap/2$  e  $U^*(t) = U_c - Ap/2$  e são armazenados na memória.
- 3) Em  $N2 < t < N3$  Envia-se os sinais  $U^*(t)$  ao circuito inversor da armadura do motor e os valores da corrente de armadura  $I_a(t)$  e velocidade do rotor  $\Omega(t)$  são adquiridos e armazenados na memória.
- 4) Em  $N2 < t < N3$  a rede neural é treinada usando o algoritmo de propagação retroativa com os valores adquiridos na etapa 3.
- 5) Em  $N3 < t < N4$  o controle retorna ao controlador neural direto.
- 6) Repetem-se as etapas acima enquanto  $\Sigma ERRO^2 > Emax$ .

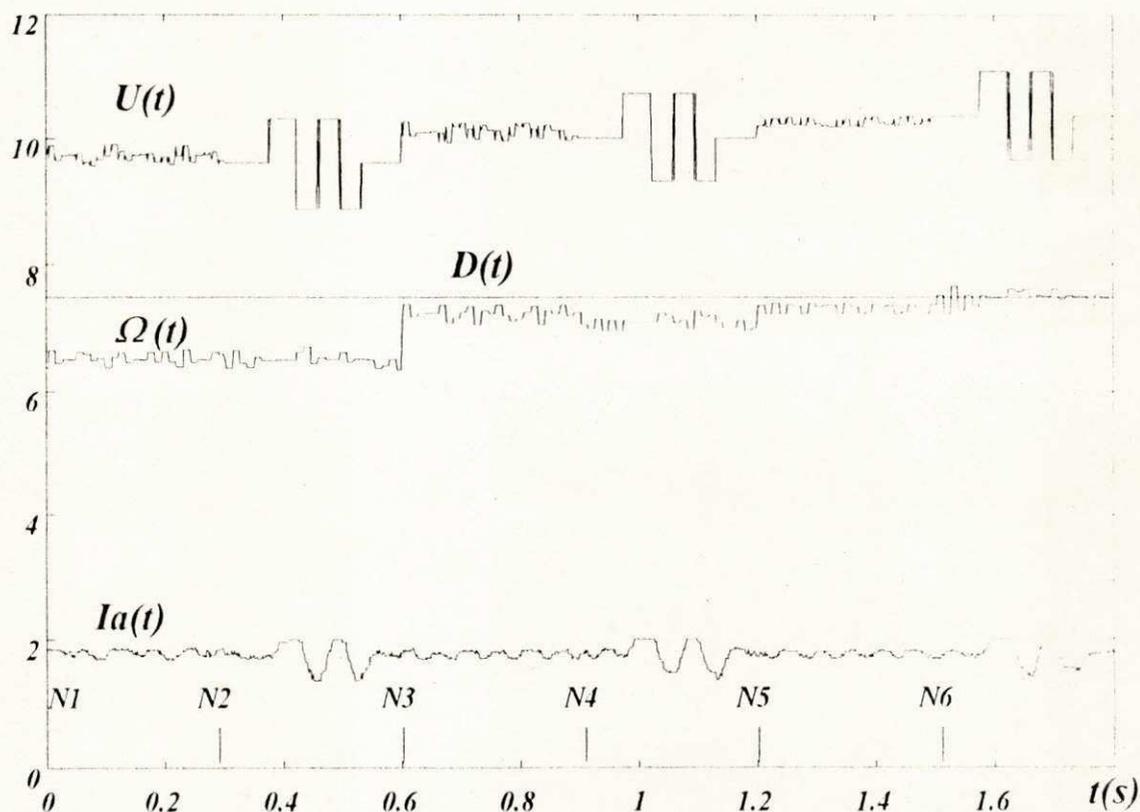


Figura 4.6 Aprendizagem "on line" da dinâmica do motor CC.

O Algoritmo 4.1 só é válido para pequenas variações na carga ou na dinâmica do motor CC. A sintonização do controlador neural direto inverso é dificultada por ele ser um sistema em

malha aberta isto é, sem realimentação [Hunt et alii, 1992]. Na próxima Seção será analisada a possibilidade de tornar o controlador neural direto inverso num controlador neural, trabalhando em malha fechada, e que possa se adaptar em tempo real.

#### 4.6 - Controlador neural adaptativo direto

Na Figura 4.7 é mostrado o esquema geral do controlador neural adaptativo direto, como descrito por Tanomaru & Omatu [1991] e Khalid & Omatu [1992]. Ele se caracteriza por ser um sistema de malha fechada em que a RNMC pode ser treinada "off line", como no controlador neural direto inverso, e sob algumas condições, ela pode ser treinada "on line".

No controlador neural adaptativo direto, o sinal na saída da planta ( $\Omega(t+1)$ ) deve seguir um sinal desejado ( $D(t+1)$ ). Isso é conseguido pela modificação de  $U(t)$  e pelo treinamento da RNMC, de forma a minimizar o índice de desempenho  $J_D(t+1)$  definido na equação 4.6 (ver Tanomaru & Omatu [1991]).

$$J_D(t+1) = \frac{1}{2}e(t+1)^2 = \frac{1}{2}[D(t+1) - \Omega(t+1)]^2 \quad (4.6)$$

No esquema do controlador neural direto (ver a Figura 4.7) define-se  $Ua(t)$  como o valor de saída atual da RNMC controladora e  $U^*(t)$  como o valor de entrada da planta capaz de reduzir o índice de desempenho  $J_D(t+1)$ .  $U^*(t)$  pode ser calculado usando a regra delta generalizada como mostrado na equação 4.7. Nessa equação, define-se  $\eta$  como o fator de adaptação do controlador neural direto ( $\eta > 0$ ) e o jacobiano da planta é definido como  $\frac{\partial \Omega(t+1)}{\partial U(t)}$ . Geralmente o jacobiano da planta é desconhecido e deve ser calculado indiretamente. O incremento da saída da RNMC controladora é definido como  $\Delta U(t) = U^*(t) - Ua(t)$ .

$$U^*(t) = Ua(t) - \eta \frac{\partial J_D(t+1)}{\partial U(t)} = Ua(t) + \eta e(t+1) \frac{\partial \Omega(t+1)}{\partial U(t)} \quad (4.7)$$

Nas Seções seguintes serão analisadas arquiteturas de controladores neurais baseadas em diferentes técnicas para o cálculo do valor de  $\Delta U(t)$ .

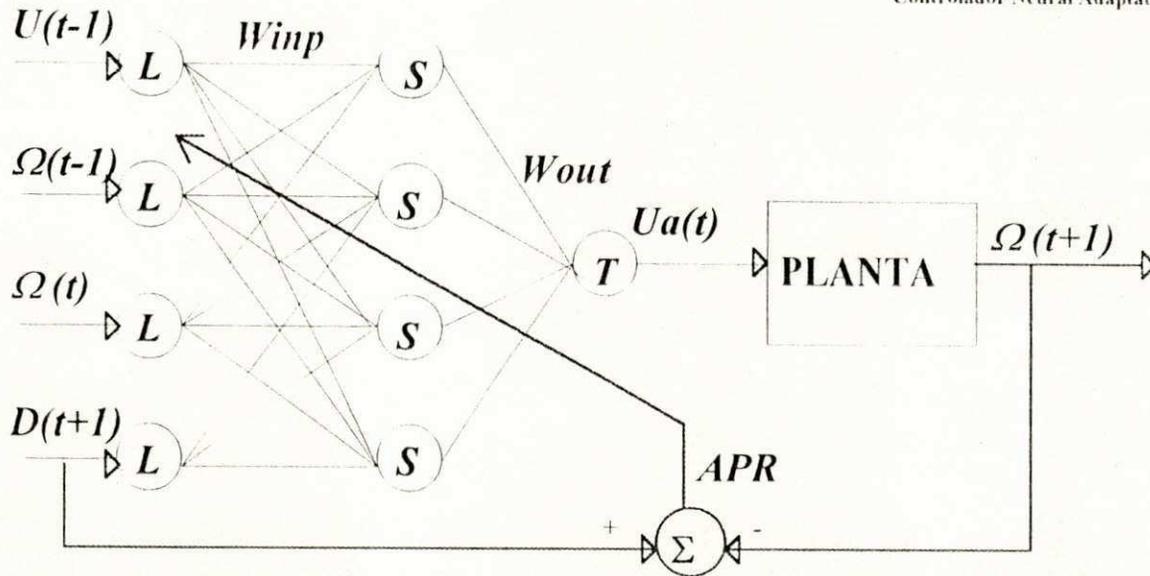


Figura 4.7 - Controlador neural adaptativo direto.

#### 4.7 - Controlador neural adaptativo indireto

O controlador neural adaptativo indireto, na forma descrita por Tanomaru & Omatu [1992], usa duas RNMC, uma RNMC CONTROLADOR e uma RNMC EMULADOR. A RNMC EMULADOR, depois de treinada, é usada para calcular indiretamente, usando o algoritmo de propagação retroativa, o jacobiano da planta. Conhecendo-se o valor estimado do jacobiano da planta, pode-se calcular o incremento  $\Delta U(t)$  da saída da RNMC CONTROLADOR usando a equação 4.7.

Na figura 4.9 é mostrado o esquema geral do controlador neural adaptativo indireto implementado para o controle adaptativo da velocidade do motor CC. Inicialmente, as duas RNMC são treinadas "off line" como mostrado na Figura 4.8, usando o fator de convergência  $\mu=0.1$  no algoritmo de propagação retroativa. As duas RNMC foram treinadas "off line" com a apresentação de 1000 períodos dos sinais experimentais de entrada e saída do motor CC,  $U(t)$  e  $\Omega(t)$ , mostrados na Figura 4.1. Usou-se a equação 4.8 como índice de desempenho para treinamento da RNMC CONTROLADOR e a equação 4.9 como índice de desempenho para treinamento da RNMC EMULADOR.

$$J_i(t+1) = \frac{1}{2}[U(t) - U_i(t)]^2 \quad (4.8)$$

$$J_e(t+1) = \frac{1}{2}[\Omega(t) - \Omega_e(t)]^2 \quad (4.9)$$

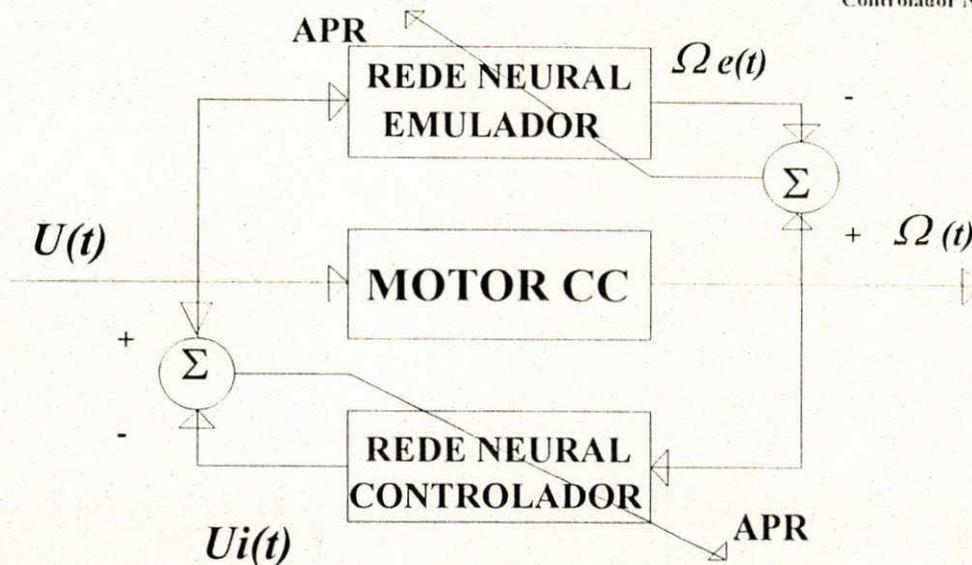


Figura 4.8 - Esquema de treinamento das RNMC.

Após o treinamento "off line" das duas RNMCs já se pode controlar a velocidade do motor CC. Durante o controle "on line", havendo variação na dinâmica do motor CC (ou da carga), a RNMC CONTROLADOR tem que se adaptar a essa variação. Observe-se que na Figura 4.8 usou-se  $\Omega_e(t+1)$  como o valor estimado da velocidade angular  $\Omega(t+1)$  do motor CC. A variação "on line" na dinâmica do motor CC (ou da carga) fará com que o valor da saída  $\Omega_e(t+1)$  da RNMC EMULADOR seja diferente do valor  $\Omega(t+1)$  da saída da planta. O índice de desempenho mostrado na equação 4.10 é usado para forçar que a saída  $\Omega(t+1)$  do motor CC siga a saída desejada  $D(t+1)$  durante o controle "on line" do motor CC.

$$J_D(t+1) = \frac{1}{2} [D(t+1) - \Omega(t+1)]^2 \quad (4.10)$$

O índice de desempenho  $J_e(t+1)$ , juntamente com o algoritmo de propagação retroativa, também é usado para calcular o valor do incremento  $\Delta U(t) = U^*(t) - U_a(t)$  da saída da RNMC controladora. A idéia é similar à do algoritmo de propagação retroativa para ajuste dos pesos e parâmetros da RNMC (ver a implementação do algoritmo de propagação retroativa no Anexo A), o sinal do erro é propagado retroativamente à entrada do neurônio da camada de entrada para modificar o valor de ativação desse neurônio no intuito de diminuir o erro na saída da planta [Hoskins et alii, 1992]. Nas equações A.5e e A.8e do Anexo A, mostra-se que, no algoritmo de propagação retroativa, o incremento do valor do sinal das entradas dos neurônios pode ser calculado de uma forma semelhante aos incrementos dos pesos e parâmetros da RNMC.

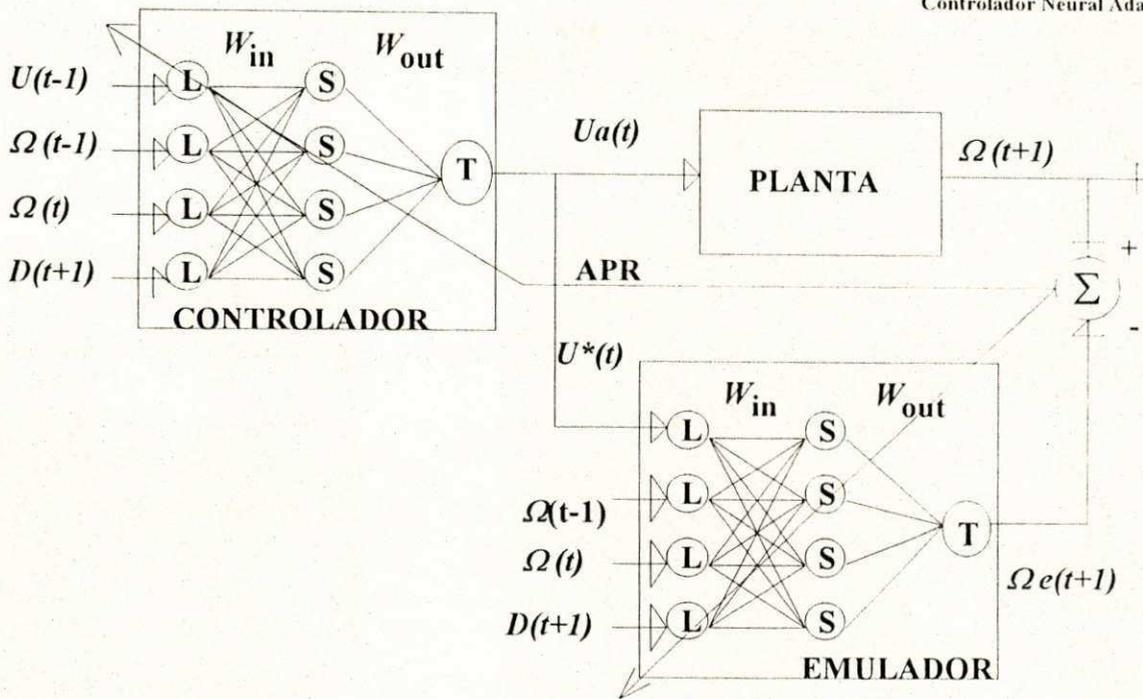


Figura 4.9 - Controlador neural indireto.

Cavalcanti et alii [1992d] usando a configuração mostrada na Figura 4.9, propôs e implementou o Algoritmo 4.2 para o controle da velocidade do motor CC e o treinamento em tempo real das RNMCs do controlador neural adaptativo indireto em função da variação da dinâmica da planta. Antes do controle em tempo real, as duas RNMCs devem ser treinadas "off line". No Algoritmo 4.2 definiu-se  $ERRO = D(t) - \Omega(t)$  e com  $\sum ERRO^2 > E_{max}$  (100 valores do  $ERRO$  e  $E_{max} = 1$  p.u.), usando  $\eta = 0.1$  no fator de adaptação do controlador neural direto, e  $\mu = 0.1$  no fator de convergência para treinamento da RNMC com o algoritmo de propagação retroativa.

Os resultados experimentais obtidos com o controlador neural indireto no controle "on line" da velocidade do motor CC são mostrados na Figura 4.10. Nas curvas da Figura 4.10,  $U(t)$  está representado em p.u. com valor de base de 12Volts,  $D(t)$  e  $\Omega(t)$  estão representados em p.u. com valor de base de 2rd/s. No instante  $t=t_1$  existe um erro entre os valores  $D(t)$  e  $\Omega(t)$ . Usando a RNMC EMULADOR e com o algoritmo de propagação retroativa, a partir do instante  $t=t_1$ , é calculado o incremento  $\Delta U(t)$  que é adicionado a  $Ua(t)$  para gerar o novo valor de controle. A partir do instante  $t_1$  e até  $t_2$ , a saída da planta  $\Omega(t)$  tende para o valor de saída desejado  $D(t)$ . Considera-se que  $\sum ERRO^2 \leq E_{max}$  a partir do instante  $t_2$ .

## Algoritmo 4.2

## Treinamento das RNMC do controlador neural adaptativo indireto

1) A partir do instante  $t=t1$  calcula-se o valor de controle  $Ua(t)$  usando a RNMC CONTROLADOR.

2) Enquanto  $\Sigma ERRO^2 > Emax$ , usando a RNMC EMULADOR, o índice de desempenho da equação 4.10, e o algoritmo de propagação retroativa, calcula-se  $\Delta U(t)$  e o adiciona a  $Ua(t)$ , e treina-se a RNMC CONTROLADOR.

3) A partir do instante  $t=t2$ , treina-se a RNMC EMULADOR usando o índice de desempenho da equação 4.10.

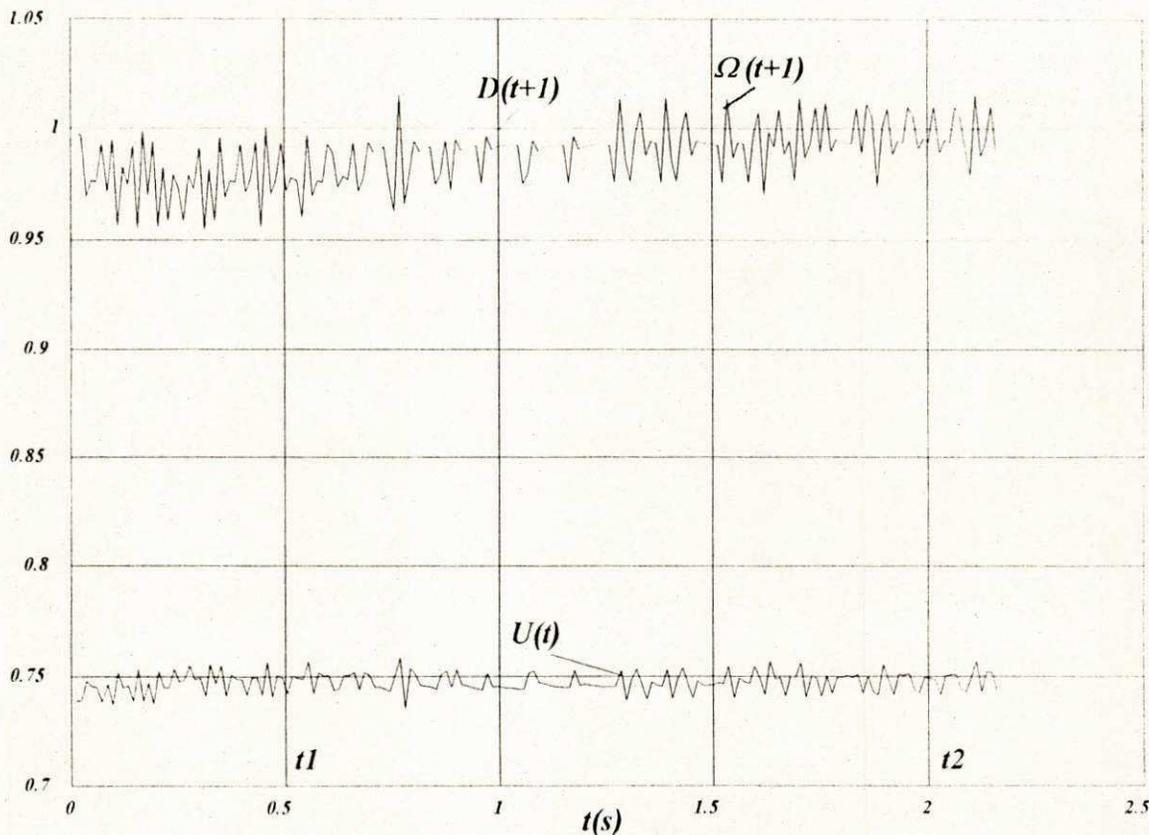


Figura 4.10 Resultados experimentais obtidos com o controlador neural indireto.

Na Seção a seguir é feita uma abordagem diferente para o controlador neural adaptativo. A planta será considerada como uma função não linear que pode ser emulada por duas RNMCs. O controlador neural adaptativo será baseado nessas duas RNMCs.

#### 4.8 - Controle neural adaptativo baseado em funções não lineares

Os controladores adaptativos propostos por Narendra & Parthasarathy [1990] baseiam-se no conhecimento a priori do modelo da planta. Eles apresentaram resultados satisfatórios de simulação obtidos com a identificação seguida do controle da planta e identificação e controle simultâneo da planta. Nas Seções anteriores, baseado nos trabalhos de Tanomaru e Cavalcanti, apresentam-se resultados experimentais de um controlador neural adaptativo que necessita do conhecimento do jacobiano da planta. Existe uma outra abordagem, que é o controlador neural adaptativo proposto por Chen [1990], mostrado no esquema geral da Figura 4.11, desenvolvido para ser aplicado em sistemas seguidores auto-sintonizáveis, e que utiliza duas RNMC para emulação da planta que é considerada uma função não linear.

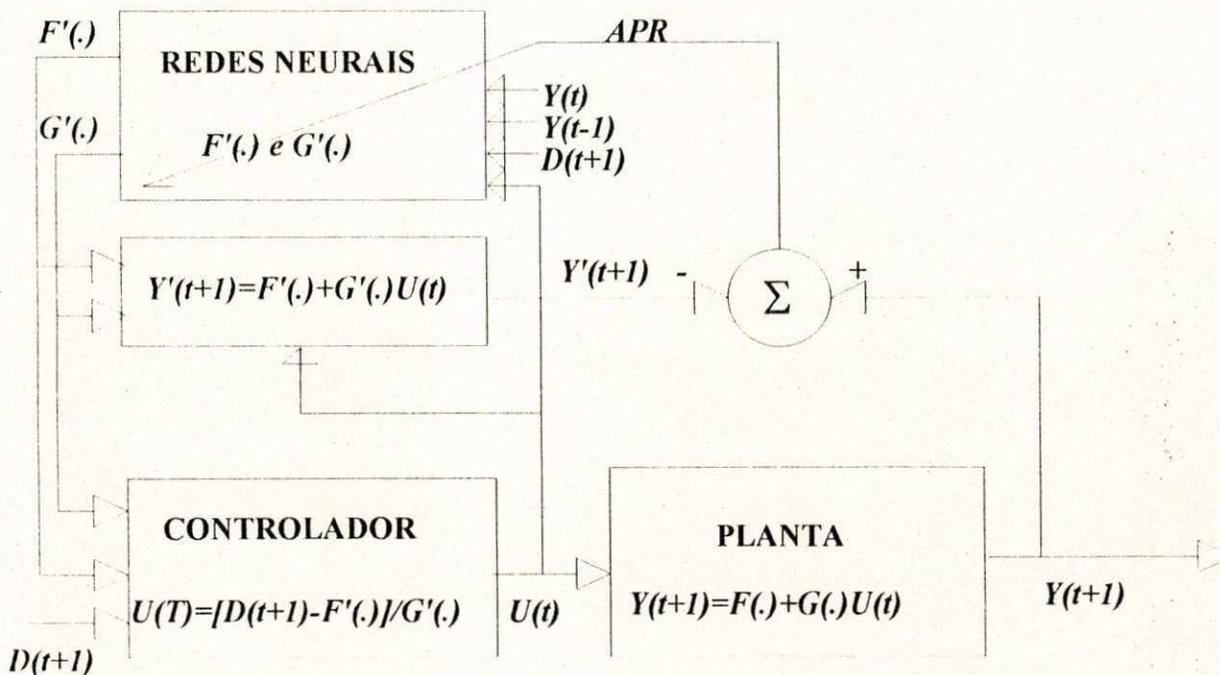


Figura 4.11 - Esquema do controlador neural adaptativo baseado em funções não lineares.

A seguir, apresenta-se o cálculo dos valores usados para adaptação "on line" das RNMC  $F'(.)$  e  $G'(.)$  do controlador neural adaptativo proposto por Chen [1990]. Considerando-se  $U(t)$  como entrada e  $Y(t)$  como saída da planta,  $Y(t)$  pode ser representada pelas funções  $F(.)$  e  $G(.)$  mostradas nas equações 4.11a e 4.11b.

$$Y_{t+1} = F(Y_t, Y_{t-1}, \dots, Y_{t-p}, U_t, U_{t-1}, \dots, U_{t-p}) + G(Y_t, Y_{t-1}, \dots, Y_{t-p}, U_{t-1}, \dots, U_{t-p})U_t \quad (4.11a)$$

$$Y(t+1) = F(.) + G(.)U(t) \quad (4.11b)$$

Supondo-se que as funções  $F(.)$  e  $G(.)$  da planta são perfeitamente conhecidas e que a função  $G(.)$  é diferente de zero, pode-se calcular o controle  $U(t)$  da planta usando-se a equação 4.12.

$$U(t) = \frac{-F(.) + Y(t+1)}{G(.)} \quad (4.12)$$

As funções  $F(.)$  e  $G(.)$  que representam a planta geralmente são desconhecidas mas Chen [1990] mostrou que se pode usar redes neurais para estimá-las. Denominando-se essas redes neurais de  $F'(Y,U,W)$  e  $G'(Y,U,V)$  com  $W$  e  $V$  representando o conjunto de parâmetros das redes neurais  $F'()$  e  $G'()$ , respectivamente.

Considere a equação 4.13a para o modelo estimado da planta com  $F'()$  e  $G'()$  representando os valores na saída das RNMC. Assumindo-se que o objetivo do controle é que o sinal de saída da planta siga um sinal  $D(t)$ , o controlador neural adaptativo pode ser representado pela equação 4.13b.

$$Y'(t+1) = F'(.) + G'(.)U(t) \quad (4.13a)$$

$$U(t) = \frac{-F'(.) + D(t+1)}{G'(.)} \quad (4.13b)$$

Combinando as equações 4.11b e 4.13b, obtém-se a equação 4.14.

$$Y(t+1) = F(.) + G(.) \left[ \frac{-F'(.) + D(t+1)}{G'(.)} \right] \quad (4.14)$$

Definindo-se como índice de desempenho as equações 4.15.

$$E = \frac{1}{2}(e_{t+1})^2 = \frac{1}{2}[D(t+1) - Y(t+1)]^2 \quad (4.15)$$

Os parâmetros das RNMC  $F'()$  e  $G'()$  devem ser ajustados para reduzir a zero o índice de desempenho da equação 4.15. A adaptação dos parâmetros  $W$  e  $V$  dessas duas redes neurais pode ser feita com a regra delta generalizada. Chen [1990] usou a regra delta generalizada para mostrar a adaptação dos parâmetros das RNMC  $F'()$  e  $G'()$ . Chen [1990] obteve as equações 4.16a e 4.16b a partir das equações 4.14 e 4.15.

$$\frac{\partial E}{\partial W} = \frac{\partial F'(Y_t; W)}{\partial W} \frac{G(.)}{G'(Y_t; V)} e_{t+1} \quad (4.16a)$$

$$\frac{\partial E}{\partial V} = \frac{\partial G'(Y_t; V)}{\partial V} \frac{G(.)}{G'(Y_t; V)} U_t e_{t+1} \quad (4.16b)$$

Cavalcanti et alii [1993b] mostraram que os valores na saída das RNMC  $F'()$  e  $G'()$ , usando a regra delta generalizada, devem ser adaptados de acordo com as equações 4.17a e 4.17b.

$$F'(\cdot)_{t+1} = F'(\cdot)_t - \mu \frac{\partial \mathcal{E}}{\partial F'(\cdot)} \quad (4.17a)$$

$$G'(\cdot)_{t+1} = G'(\cdot)_t - \eta \frac{\partial \mathcal{E}}{\partial G'(\cdot)} \quad (4.17b)$$

Cavalcanti et alii [1993b], usando a regra da cadeia nas equações 4.16, obtiveram os valores de  $\frac{\partial \mathcal{E}}{\partial F'(\cdot)}$  e  $\frac{\partial \mathcal{E}}{\partial G'(\cdot)}$  que estão mostrados nas equações 4.18a e 4.18b.

$$\frac{\partial \mathcal{E}}{\partial F'(\cdot)} = \frac{\partial \mathcal{E}}{\partial W} \frac{\partial W}{\partial F'(\cdot)} \quad \text{ou} \quad \frac{\partial \mathcal{E}}{\partial W} = \frac{\partial \mathcal{E}}{\partial F'(\cdot)} \frac{\partial F'(\cdot)}{\partial W} \quad (4.18a)$$

$$\frac{\partial \mathcal{E}}{\partial G'(\cdot)} = \frac{\partial \mathcal{E}}{\partial V} \frac{\partial V}{\partial G'(\cdot)} \quad \text{ou} \quad \frac{\partial \mathcal{E}}{\partial V} = \frac{\partial \mathcal{E}}{\partial G'(\cdot)} \frac{\partial G'(\cdot)}{\partial V} \quad (4.18b)$$

Observando-se as equações 4.16 vê-se que elas dependem diretamente da função  $G(\cdot)$  que é desconhecida. Estimando-se o sinal de  $G(\cdot)$ , as equações 4.16 podem ser avaliadas da forma mostrada nas equações 4.19a e 4.19b com  $SGN[G(\cdot)]$  indicando o sinal da função  $G(\cdot)$ .

$$\frac{\partial \mathcal{E}}{\partial W} = \frac{\partial F'(Y_t; W)}{\partial W} \frac{SGN[G(\cdot)]}{G'(Y_t; V)} e_{t+1} \quad (4.19a)$$

$$\frac{\partial \mathcal{E}}{\partial V} = \frac{\partial G'(Y_t; V)}{\partial V} \frac{SGN[G(\cdot)]}{G'(Y_t; V)} U_t e_{t+1} \quad (4.19b)$$

A partir das equações 4.18a, 4.18b, 4.19a e 4.19b pode-se calcular a variação do índice de desempenho em função de  $F'(\cdot)$  e  $G'(\cdot)$ . Nas equações 4.20a e 4.20b são mostrados os valores obtidos para adaptação das saídas das RNMC  $F'(\cdot)$  e  $G'(\cdot)$ .

$$\frac{\partial \mathcal{E}}{\partial F'(Y_t; W)} = \frac{SGN[G(\cdot)]}{G'(Y_t; V)} e_{t+1} \quad (4.20a)$$

$$\frac{\partial \mathcal{E}}{\partial G'(Y_t; W)} = \frac{SGN[G(\cdot)]}{G'(Y_t; V)} U_t e_{t+1} \quad (4.20b)$$

Estimando-se o sinal de  $G(\cdot)$  e conhecendo-se as saídas das RNMC  $F'(\cdot)$  e  $G'(\cdot)$ , pode-se treinar essas RNMCs para adaptá-las à variação na dinâmica da planta usando as equações 4.21a e 4.21b.

$$F'(\cdot)_{t+1} = F'(\cdot)_t - \mu \frac{SGN[G(\cdot)]}{G'(Y_t; V)} e_{t+1} \quad (4.21a)$$

$$G'(.)_{t+1} = G'(.)_t - \eta \frac{SGN[G'(.)]}{G'(Y_t; V)} U_t e_{t+1} \quad (4.21b)$$

A seguir serão mostrados os resultados experimentais obtidos, no controle da velocidade do motor CC, usando controlador neural proposto por Chen.

Inicialmente, foi feito o treinamento "off line" das duas redes neurais  $F'(.)$  e  $G'(.)$  usando os dados do motor CC apresentados na figura 4.1. Usou-se a mesma arquitetura para as duas RNMCs como mostrada na Figura 4.12.

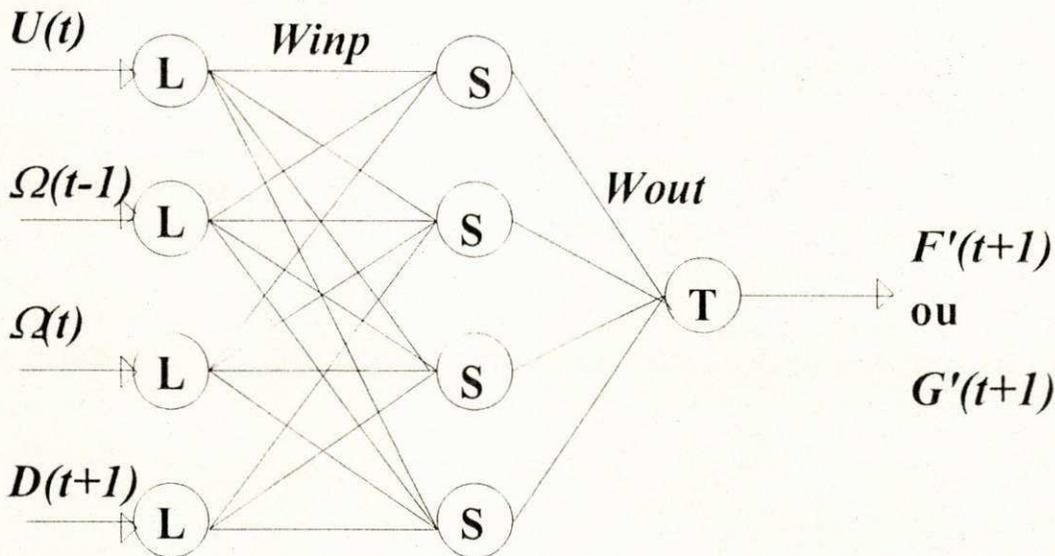


Figura 4.12 - Arquitetura da RNMC usada no controlador de Chen.

O treinamento das duas RNMC  $F'(.)$  e  $G'(.)$  foi feito com os valores calculados a partir das equações 4.21, usando os fatores de adaptação  $\mu = 0.1$  e  $\eta = 0.01$ , e estimou-se que  $SGN[G'(.)] = 1$ . A seguir, treinaram-se "off line" as duas RNMC  $F'(.)$  e  $G'(.)$ . O método de treinamento "off line" consistiu na apresentação, 1000 vezes, dos padrões das formas de ondas do motor CC, mostrados na figura 4.1. Observou-se que a saída da rede neural  $G'(.)$  é aproximadamente constante durante um ciclo de aprendizagem e que  $G'(.)$  poderia ser identificada com um único peso. Os resultados obtidos usando o controlador de Chen treinado "off line" foram semelhantes aos mostrados na Figura 4.4.

A seguir, verificou-se, experimentalmente em tempo real, o desempenho do controlador neural adaptativo de Chen após a variação na dinâmica do motor CC. Na figura 4.13 são mostrados os resultados experimentais do motor CC obtidas por Cavalcanti et alii [1993b] com o controlador neural adaptativo de Chen. Na Figura 4.13, a abscissa representa o tempo em segundos, a velocidade angular  $\Omega(t)$  e a referência  $D(t)$  são representados em rps, a corrente da armadura  $I_a(t)$  é representada em ampères.

Na Figura 4.13, em  $t=0$  a velocidade angular  $\Omega(t)$  do motor CC é menor que a velocidade referência  $D(t)$ . No instante  $t=0$  é inicializada a adaptação em tempo real do controlador neural de Chen. Na mesma figura pode-se observar que a curva do sinal  $\Omega(t)$  tende a curva do sinal  $D(t)$ , o sinal de controle  $U(t)$  aumenta e o valor de saída da RNMC  $G'(\cdot)$  é aproximadamente constante.

No algoritmo 4.3, implementado por Cavalcanti et alii [1993b], descreve-se o treinamento em tempo real das RNMCs  $F'(\cdot)$  e  $G'(\cdot)$  do controlador de Chen. Para o Algoritmo 4.3, definiu-se  $ERRO = D(t) - \Omega(t)$  e com  $\Sigma ERRO^2 > Emax$  (100 valores do  $ERRO$  e  $Emax = 1$  p.u.).

#### Algoritmo 4.3

##### Adaptação da RNMC do controlador usando funções não lineares

1) Em  $t=0$ , com  $\Sigma ERRO^2 > Emax$ , calcula-se  $U(t)$  usando a equação 4.13b.

2) Calcula-se, usando a equação 4.21, o incremento para adaptação do peso  $G'(\cdot)$  e o incremento para treinamento da RNMC  $F'(\cdot)$ .

3) Repetem-se as etapas acima enquanto  $\Sigma ERRO^2 > Emax$ .

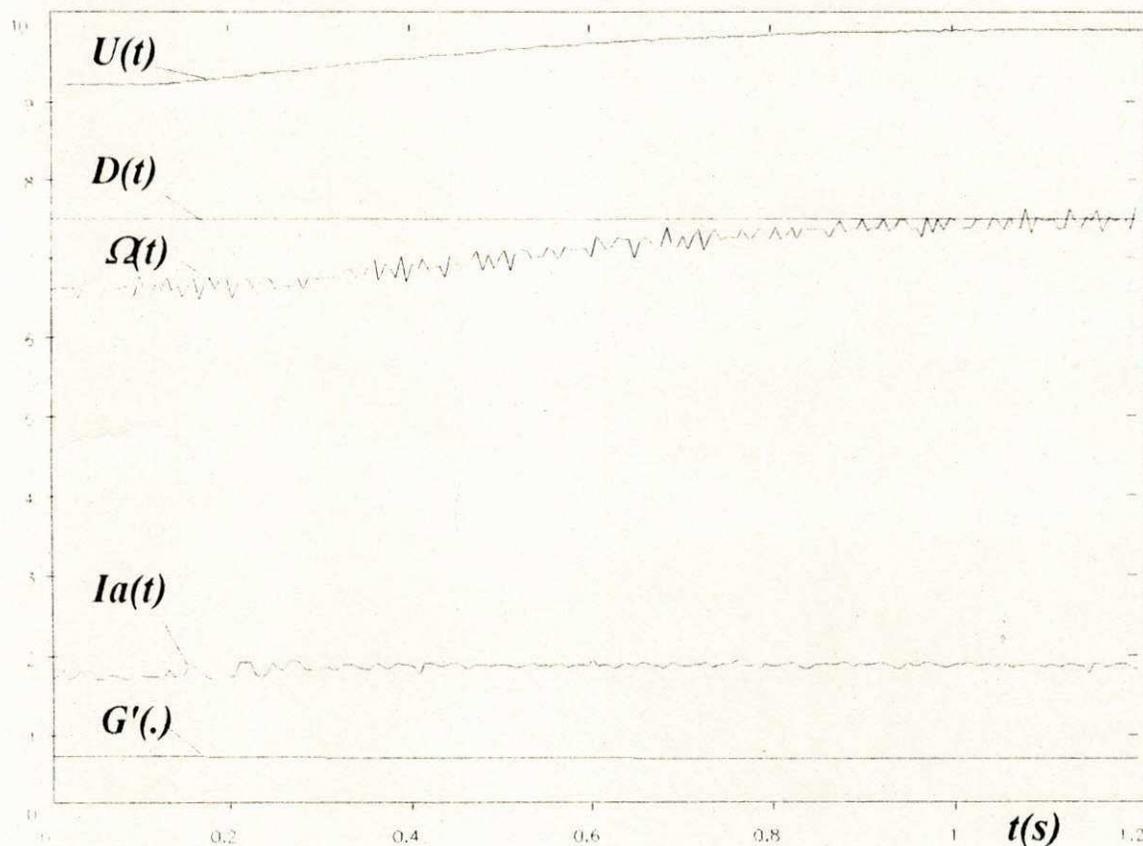


Figura 4.13 - Adaptação do controlador neural para sistemas não lineares.

A seguir, baseado no controlador neural adaptativo proposto por Chen [1990] e no controlador neural direto, mostra-se experimentalmente que, sob certas condições, o controlador neural direto pode ser adaptado em tempo real às variações na dinâmica da planta.

#### 4.9 - Controle neural adaptativo direto

Cavalcanti et alii [1993b] mostraram que os resultados obtidos na Seção anterior podem ser usados para formulação de um controlador neural adaptativo direto. Eles mostraram que como a saída da RNMC  $G'(\cdot)$  permanece constante durante o controle da velocidade do motor CC, a estimação do jacobiano da planta pode ser obtido usando algumas restrições no sinal de referência  $D(t)$  e no modelo da planta.

Reescrevendo-se a equação 4.7 na forma da equação 4.22, usando a equação 4.13b, e assumindo-se que o sinal de referência é constante,  $D(t+2)=D(t+1)$ , obtem-se a equação 4.23.

$$U(t+1) = U(t) + \Delta U(t) = U(t) + \eta e(t+1) \frac{\partial \Omega(t+1)}{\partial U(t)} \quad (4.22)$$

$$\Delta U(t) = U(t+1) - U(t) = \frac{D(t+2) - F'(\cdot)_{t+1}}{G'(\cdot)_{t+1}} - \frac{D(t+1) - F'(\cdot)_t}{G'(\cdot)_t} = \frac{F'(\cdot)_t - F'(\cdot)_{t+1}}{G'(\cdot)_t} \quad (4.23)$$

Substituindo a equação 4.21 na equação 4.23, obtem-se a equação 4.24 para adaptação da RNMC do controlador neural adaptativo direto.

$$\Delta U(t) = \eta \frac{SGN[G(\cdot)]}{G'(Y_t; V)^2} e_{t+1} \quad (4.24)$$

O controlador neural adaptativo direto foi testado experimentalmente no controle da velocidade do motor CC. Para adaptação em tempo real do controlador neural adaptativo, usou-se, na equação (4.24) do incremento da RNMC,  $SGN[G(\cdot)] = 1$  e  $\eta/G'(\cdot)^2 = 0.1$ .

Na Figura 4.14 são mostradas as formas de onda, representadas em p.u. e com a abscissa representada em segundos, obtidas usando o controlador neural adaptativo direto cujo esquema é mostrado na Figura 4.7. Na figura 4.12 são mostrados o sinal  $U(t)$  da tensão de armadura com valor de base ao sistema p.u. de 12Volts, as velocidades de referência  $D(t)$  e velocidade atual  $\Omega(t)$  do motor CC, representadas com valor de base ao sistema p.u. de 2rps. Na Figura 4.14, em  $t=220$  ms, é iniciado o controle adaptativo. Observa-se que o sinal da velocidade do motor CC  $\Omega(t)$ , tende para o valor do sinal de referência  $D(t)$ . No Capítulo 6 será mostrado que o controlador

neural direto pode ser usado para controle em tempo real da velocidade do motor CC, sem a necessidade do treinamento prévio e "off line" da RNMC.

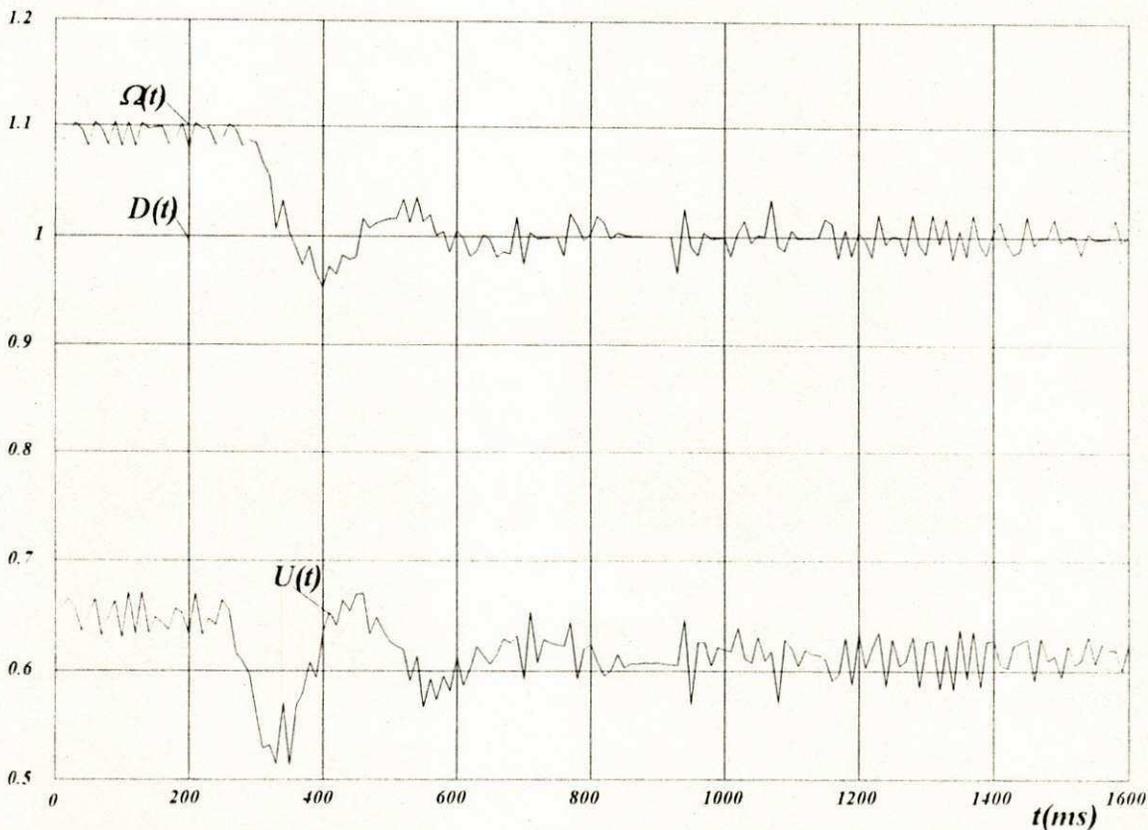
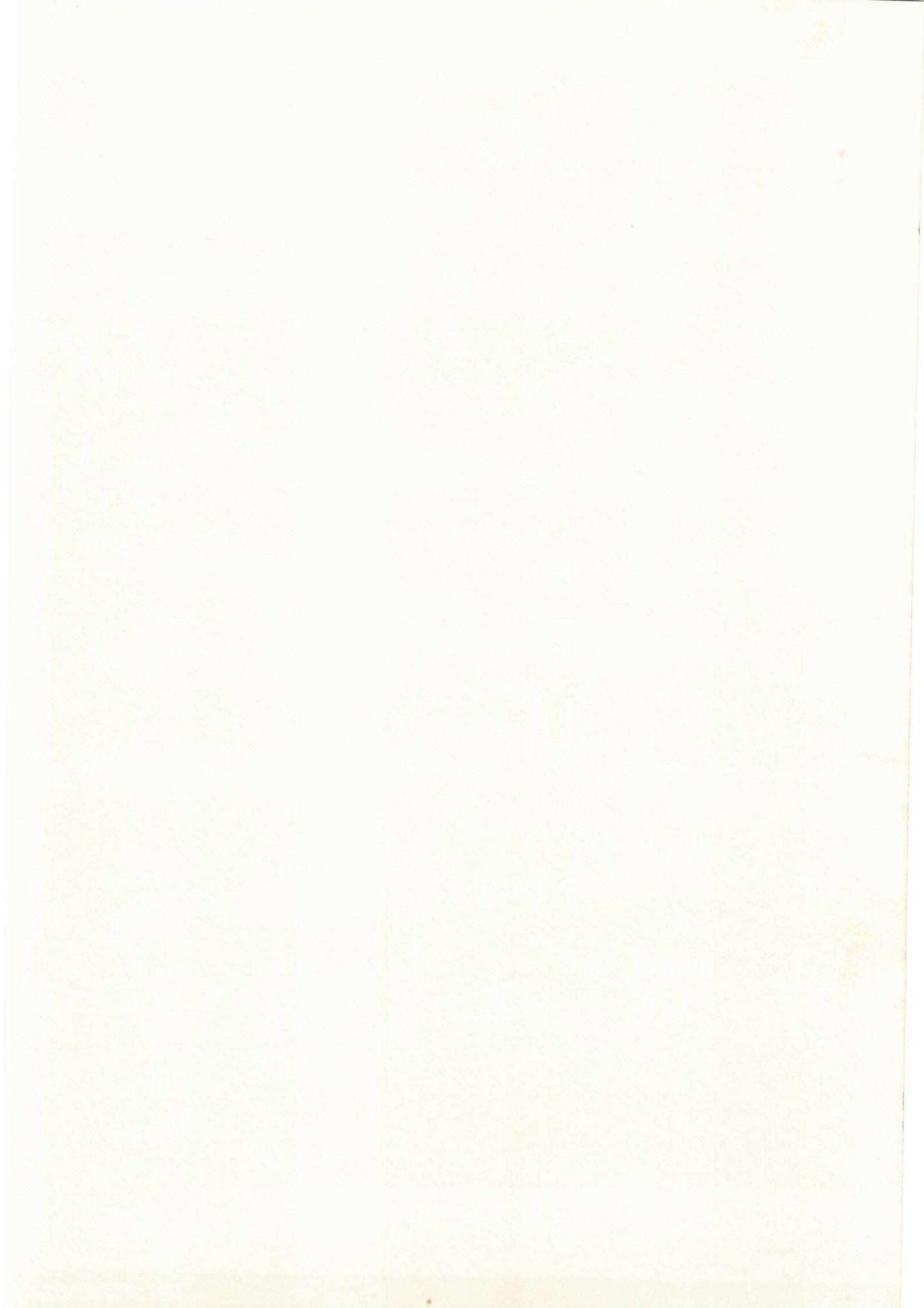


Figura 4.14 Adaptação em tempo real usando o controlador neural direto.

#### 4.10 - Conclusão

Implementou-se um controlador neural direto que foi treinado "off line" com a dinâmica inversa de uma planta. Durante a fase de controle "on line", quando houve variação na dinâmica da planta, apresentou-se um algoritmo heurístico para sintonização do controlador neural direto. Mostrou-se a necessidade do conhecimento do jacobiano da planta para se implementar um controlador neural adaptativo direto e que se pode usar uma RNMC para indiretamente estimar o jacobiano da planta. Usando-se duas RNMC, apresentou-se um controlador neural adaptativo indireto em que o jacobiano da planta é estimado por uma RNMC EMULADOR. A seguir, apresentou-se um outro tipo de controlador neural adaptativo, baseado na representação da planta como uma função não linear. A partir dos resultados obtidos com este tipo de controlador neural adaptativo, mostrou-se que, sob certas condições no sinal de referência e nas características da planta, é possível o desenvolvimento de um controlador neural adaptativo direto. Por último, apresentaram-se resultados experimentais obtidos no controle "on line" da velocidade do motor CC usando o controlador neural adaptativo direto.





## COMPARAÇÃO ENTRE OS CONTROLADORES NEURAIIS E CONTROLADORES CONVENCIONAIS

### 5.1 - Introdução

Neste Capítulo, inicialmente comparam-se os diferentes tipos de controladores convencionais com os controladores neurais. A seguir, comparam-se entre si os diferentes tipos de controladores neurais baseados em RNMC e usando o algoritmo de propagação retroativa para treinamento.

### 5.2 - Comparação entre controladores convencionais e neurais

Alguns pesquisadores compararam o desempenho dos controladores convencionais com o desempenho dos controladores não convencionais considerando: a) o ruído existente nos sinais de entrada e saída da planta; b) o distúrbio na carga do sistema; c) a superfície de controle.

Kraft & Campagna [1990] compararam o controlador neural com dois tipos de controladores adaptativos, um usando STR e o outro baseado no método dos gradientes usando funções de Lyapunov [Landau, 1979][Åstrom & Wittenmark, 1989]. Eles concluíram que o controlador neural é relativamente insensível ao ruído e é aparentemente mais adequado ao controle de sistemas não lineares. Cavalcanti et alii [1991b][1992a] compararam controladores "deadbeat" (ver também Gokhale et alii [1987]) com controladores neurais no controle de

sistemas UPS. Eles também concluíram que o controlador neural é mais robusto ao ruído do que o controlador "deadbeat".

Khaid & Omatu [1992] implementaram um controlador neural direto para um sistema de controle de temperatura e compararam o desempenho do controlador neural com um controlador convencional do tipo PI. Eles observaram que o controlador neural teve um melhor desempenho, comparado com o controlador PI, quando houve distúrbio na carga do sistema.

Usando outro tipo de abordagem, Chow et alii [1992] compararam controladores neurais, treinados a partir de controladores baseados em regras fuzzy, com controladores convencionais do tipo PI e com controladores fuzzy. Usando como critério de comparação as superfícies de controle geradas pelos controladores, e se sabendo que a superfície de controle gerada por um controlador PI é um plano, eles concluíram que a superfície de controle gerada pelo controlador neural, com a superfície de controle mostrada na Figura 5.2 (com  $U(t)$  representando o valor de controle,  $E(t)$  o erro e  $CE(t)$  a variação do erro), e pelo controlador fuzzy, com a superfície de controle mostrada na Figura 5.1, tem mais informações sobre a dinâmica da planta do que a superfície gerada pelo controlador PID. Eles também concluíram que a superfície de controle gerada pelo controlador neural, treinado com os sinais de entrada e saída gerados pelo controlador fuzzy, é contínua e apresenta uma maior suavidade. Cavalcanti et alii [1993a] também mostraram resultados experimentais obtidos com o controlador fuzzy. No Anexo B são apresentados mais detalhes sobre o projeto e implementação dos controladores fuzzy.

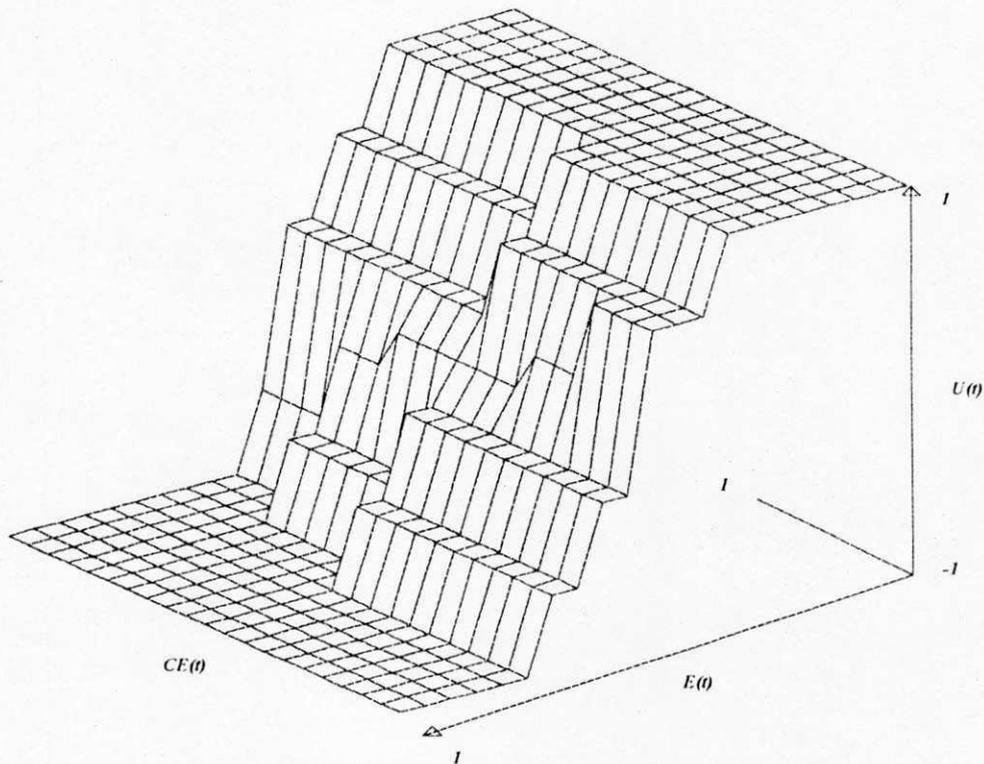


Figura 5.1 - Superfície de controle gerada pelo controlador fuzzy.

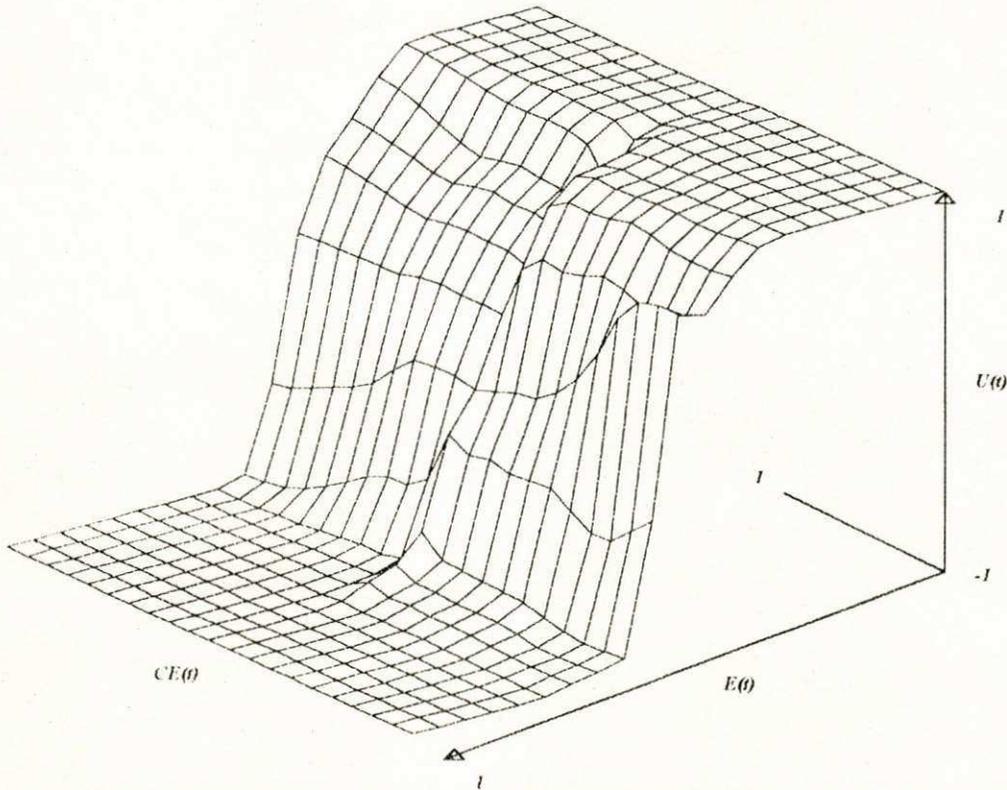


Figura 5.2 - Superfície de controle gerada pelo controlador neural.

De um modo geral, os pesquisadores observaram que a rede neural é lenta durante a fase inicial ("off line") de aprendizagem da dinâmica da planta e que os controladores neurais podem ser implementados sem o perfeito conhecimento da dinâmica da planta. Alguns pesquisadores observaram que durante o treinamento, a RNMC pode convergir para um mínimo diferente do mínimo global.

A escolha de sinais repetitivos tipo senóides e trem de pulsos, como sinais de referência, geralmente é feita para facilitar a escolha dos índices de desempenho usado nas comparações entre os diferentes tipos de controladores. Por exemplo, Cavalcanti et alii [1991] usaram a somatória do erro ao quadrado e a distorção harmônica total como índices de desempenho para comparação entre os controladores convencionais e controladores neurais.

Na Figura 5.3 são mostrados os sinais experimentais representados em p.u., com a abscissa representada em ms, obtidos do sistema motor CC usando controlador do tipo PI. Nessa figura apresentam-se o sinal de controle  $U(t)$  (valor de base de 12 Volts), o sinal da velocidade de referência  $D(t)$  e o sinal da saída  $\Omega(t)$  (valor de base de 120 rpm). Na Figura 5.3a são mostradas as formas de ondas com referência senoidal, com índice de desempenho  $\Sigma E^2 = 16.8$  (equação 3.8), e na Figura 5.3b, são mostradas as formas de ondas com a referência constante.

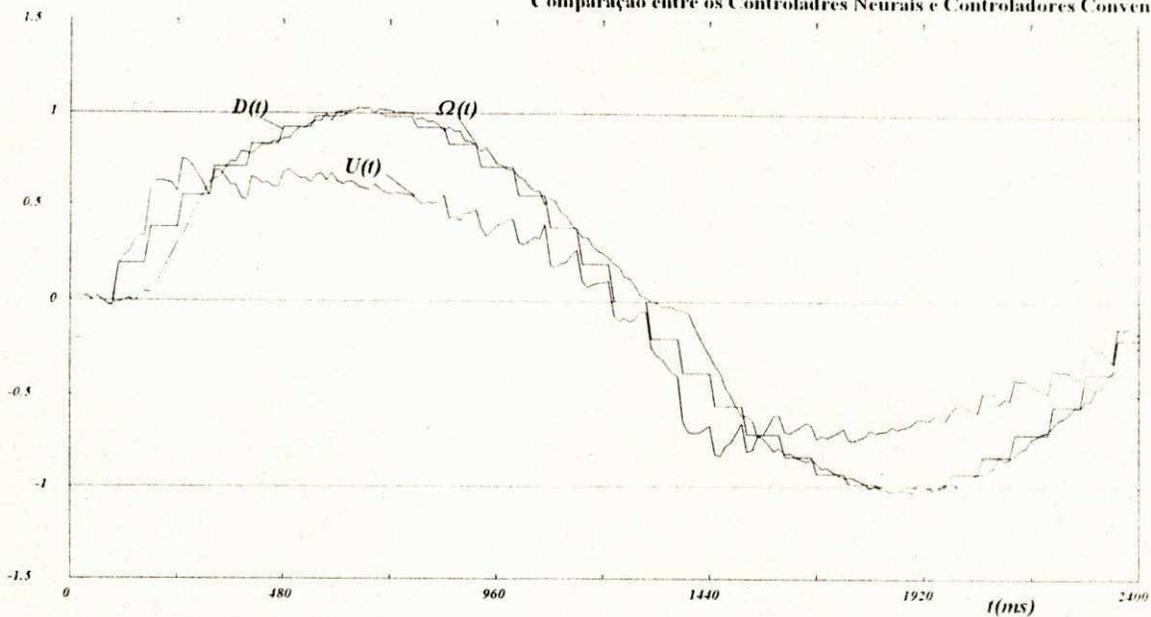


Figura 5.3a Resultados experimentais usando controlador PI - excitação senoidal.

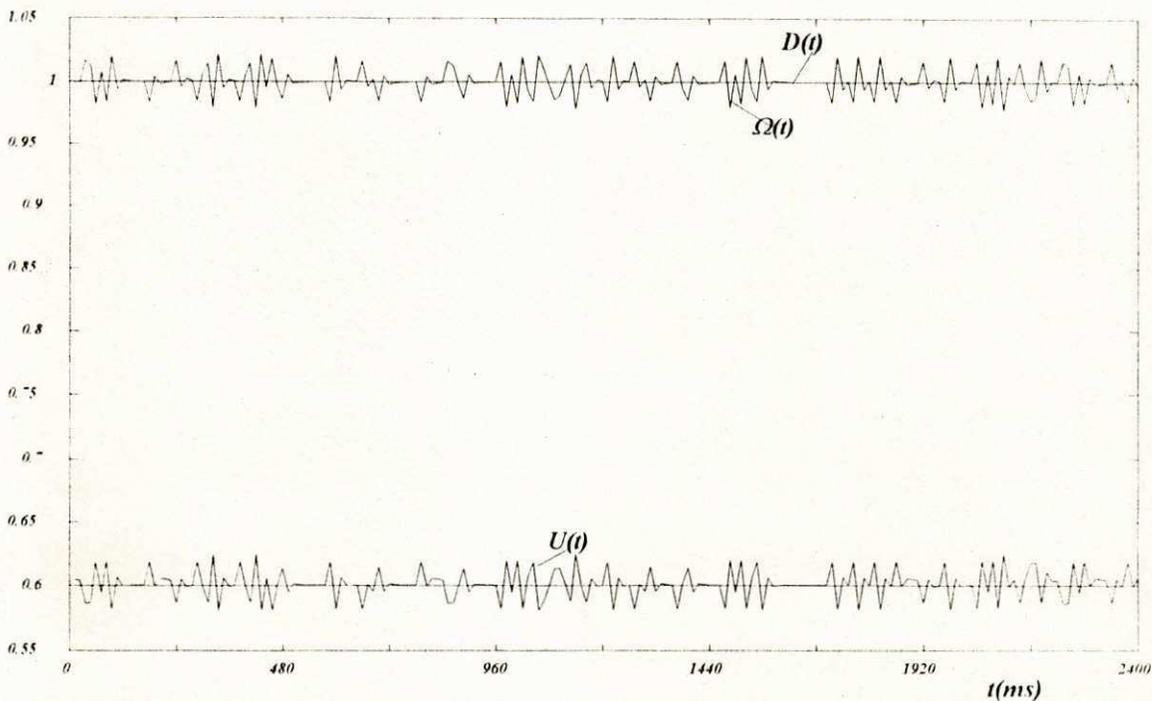


Figura 5.3b Resultados experimentais usando controlador PI - referência constante.

A seguir, os mesmos experimentos descritos anteriormente foram feitos usando os controladores adaptativos MRAC. Observou-se que os índices de desempenho obtidos com os controladores adaptativos, usando a regra MIT e usando as funções de Lyapunov, apresentam um índice de desempenho com aproximadamente a mesma ordem de grandeza.

Verificou-se que o controlador MRAC (regra MIT), usando referência senoidal, apresentou um índice de desempenho com valor  $\Sigma E^2 = 21.4$ . Este valor alto de  $\Sigma E^2$ , em relação ao valor

obtido com o controlador PI ( $\Sigma E^2 = 16.8$ ), pode ser explicado pela defasagem existente entre as formas de onda da velocidade de referência  $D(t)$  e a forma de onda do modelo de referência  $\Omega_n(t)$  quando se usa o controlador MRAC.

O índice de desempenho ( $\Sigma E^2 = 16.85$ ), obtido com o controlador neural direto inverso, apresentou aproximadamente a mesma ordem de grandeza que o índice de desempenho obtido com o controlador PI e foi menor que o índice de desempenho obtido com o controlador MIT.

Na figura 5.4 são mostradas as curvas do motor CC obtidas com o controlador MIT e usando velocidade referência constante. Os parâmetros  $t_0$  e  $s_0$  do controlador MIT foram adaptados com as formas de onda senoidais semelhantes às mostradas na Figura 5.3a. Na adaptação em tempo real, em  $t=0$  na abscissa da curva da Figura 5.4, a saída  $\Omega(t)$  da planta está deslocada em relação à referência constante  $D(t)$ . A adaptação dos parâmetros do controlador MIT é iniciada em  $t=0$ . O sinal de saída da planta (os valores de  $\Omega(t)$  tendem para o valor referência  $D(t)$  na Figura 5.4) tende ao valor do sinal de referência da velocidade com os parâmetros  $(s_0; t_0)$  variando de  $(1.27; 1.33)$  a  $(2.; 1.99)$ .

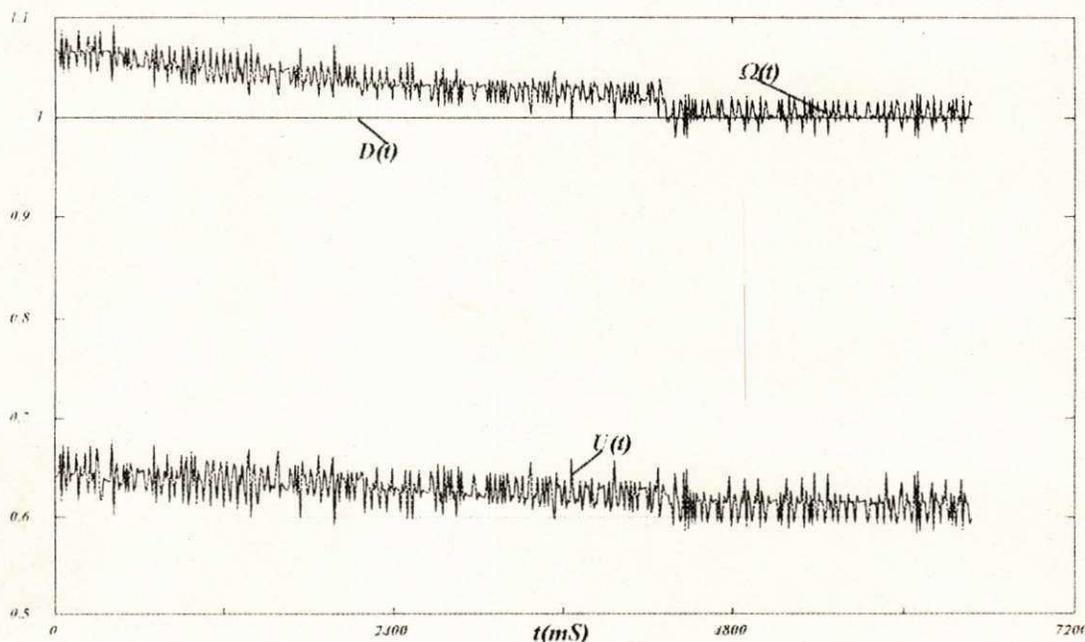


Figura 5.4 Adaptação do controlador MIT.

O controlador neural, logo após o treinamento "off line" da RNMC, geralmente necessita de uma sintonização fina ou adaptação em tempo real. No Capítulo 4 foram mostradas diversas arquiteturas de controladores neurais que podem ser adaptados em tempo real.

### 5.3 - Comparação entre os controladores neurais

O controlador neural adaptativo indireto apresenta resultados de adaptação em tempo real semelhante às observadas com o controlador neural adaptativo baseado em funções não lineares.

São necessárias duas RNMC para a implementação dos controladores neurais adaptativo indireto e usando funções não lineares.

No Capítulo 4 mostrou-se que, sob certas condições, o controlador neural direto que só necessita de uma única RNMC, pode ser adaptado em tempo real à variação na dinâmica do motor CC. Como o controlador neural direto só necessita de uma única RNMC, será vantajosa a sua utilização em controle adaptativo em tempo real.





**CONTROLE NEURAL ADAPTATIVO EM TEMPO REAL**

### 6.2 - Treinamento em tempo real do controlador neural

No desenvolvimento e implementação dos algoritmos de adaptação dos controladores neurais deve ser considerado o relativo grande número de iterações necessárias para o treinamento das RNMC. Chen [1990] apresentou um algoritmo de treinamento das RNMC do seu controlador neural adaptativo que não necessita do treinamento prévio "off line" das RNMC. No Capítulo 4 foi mostrado que o controlador neural adaptativo direto, com o treinamento prévio "off line" da RNMC, pode ser adaptado em tempo real. Baseado nas condições de adaptação em tempo real do controlador neural direto, propõe-se desenvolver um algoritmo que possibilite o treinamento em tempo real da RNMC controladora, sem o seu prévio treinamento "off line".

Na Figura 6.1 é mostrado o esquema geral do controlador neural adaptativo direto usado experimentalmente no controle em tempo real da velocidade do motor CC. O sinal na saída da planta  $\Omega(t+1)$  deve seguir o sinal desejado  $D(t+1)$ , modificando  $U(t)$  e treinando a RNMC, de forma a minimizar o índice de desempenho  $J_D(t+1)$  definido na equação 6.1.

$$J_D(t+1) = \frac{1}{2}e(t+1)^2 = \frac{1}{2}[D(t+1)-\Omega(t+1)]^2 \quad (6.1)$$

Na Figura 6.1, define-se  $Ua(t)$  como o valor de saída atual da RNMC controladora.

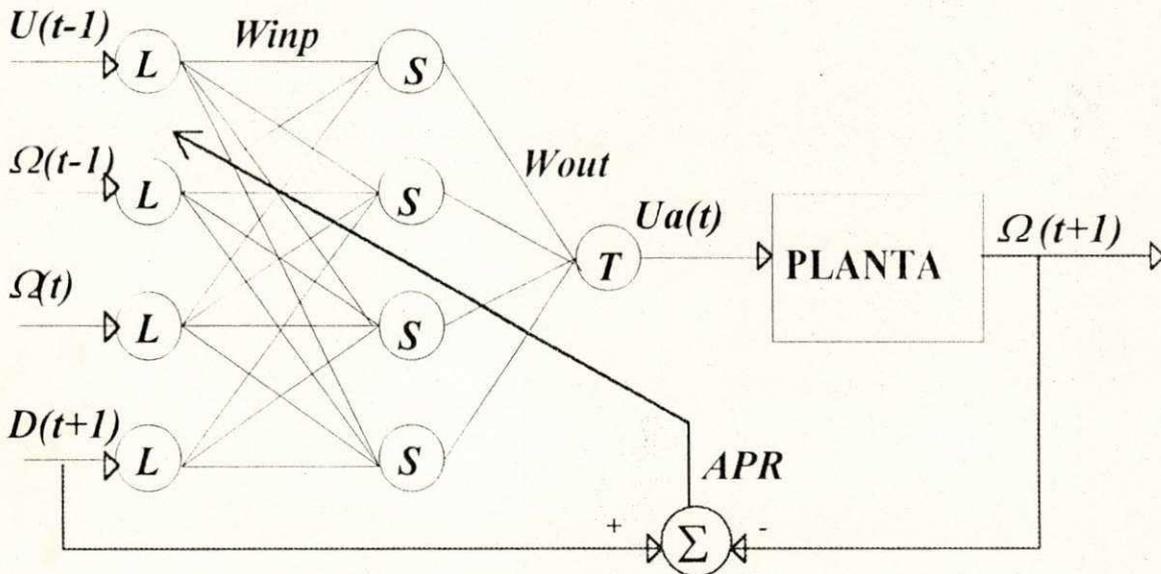


Figura 6.1 - Controlador neural adaptativo direto

No Capítulo 4, Seção 4.6, definiu-se  $U^*(t)$  como o valor capaz de reduzir o índice de desempenho  $J_D(t+1)$  que pode ser calculado usando a regra delta generalizada como mostrado na equação 6.2. No mesmo capítulo, definiu-se  $\eta$  como o fator de adaptação do controlador neural direto ( $\eta > 0$ ) e se mostrou que, para calcular o valor do sinal de controle  $U(t)$ , é necessário o conhecimento do jacobiano da planta, definido como  $\partial\Omega(t+1)/\partial U(t)$ .

$$U^*(t) = Ua(t) - \eta \hat{c} J_D(t+1) / \partial U(t) = Ua(t) + \eta e(t+1) \partial \Omega(t+1) / \partial U(t) \quad (6.2)$$

Na Seção 4.9 do Capítulo 4, usou-se, para o controle neural adaptativo direto da velocidade do motor CC, a equação 6.3 para cálculo do incremento da saída da RNMC controladora, e a equação 6.4 como valor para adaptação da RNMC. Na equação 6.4,  $G(.)$  é uma das funções do modelo não linear da planta,  $G'(.)$  representa a saída da RNMC que emula  $G(.)$ , e  $\chi = \eta \text{SGN}[G(.)] / G'(.)^2$ .

$$\Delta U(t) = U^*(t) - Ua(t) \quad (6.3)$$

$$\Delta U(t) = \eta \text{SGN}[G(.)] e(t+1) / G'(.)^2 = \chi e(t+1) \quad (6.4)$$

O controlador neural adaptativo direto foi testado experimentalmente no controle da velocidade do motor CC usando o Algoritmo 6.1, que foi desenvolvido para adaptação em tempo real do controlador neural adaptativo. Cavalcanti et alii [1993b] definiram  $\chi$  da equação 6.4 como o fator de adaptação do controlador neural adaptativo direto. A partir de testes experimentais, eles observaram que se deve usar um pequeno valor para a velocidade desejada ( $D(t) = 0.1$  p.u.) quando a velocidade do eixo do motor CC é zero ( $\Omega(t) = 0$ ).

### Algoritmo 6.1

#### Treinamento seguido de controle da RNMC

- 1) Geram-se aleatoriamente os pesos  $W_{in}$  e  $W_{out}$  entre os limites 0.2 e -0.2. Atribuem-se valores iniciais aos parâmetros dos neurônios ( $\Theta = \beta = 1$  e  $T = 0$ ),  $\mu = 0.1$  ao fator de treinamento do algoritmo de propagação retroativa, e  $\chi = 0.5$  ao fator de adaptação do controlador neural.
- 2) Treina a RNMC com o ponto zero representado por  $U(t-1) = \Omega(t) = \Omega(t-1) = D(t+1) = 0$  e faz  $D(t) = 0.1$  p.u.
- 3) Calcula:  $\Delta U(t) = \chi e(t+1)$  e  $U(t+1) = U(t) + \Delta U(t)$ . Treina a RNMC com esse novo valor de  $U(t)$ .
- 4) Repete a etapa 3 até que  $\Omega(t)$  seja igual a 0.1 p.u.

Na Figura 6.2 são mostradas as curvas experimentais, representadas em p.u. e com a abscissa representando o tempo, obtidas usando o Algoritmo 6.1 e o controlador neural da Figura 6.1 com 4 neurônios na camada oculta da RNMC. Nessa figura são mostrados o sinal  $U(t)$  da tensão de armadura com valor de base de 12Volts, as velocidades de referência  $D(t)$  e velocidade atual  $\Omega(t)$  do motor CC, representadas com valor de base de 2rps.

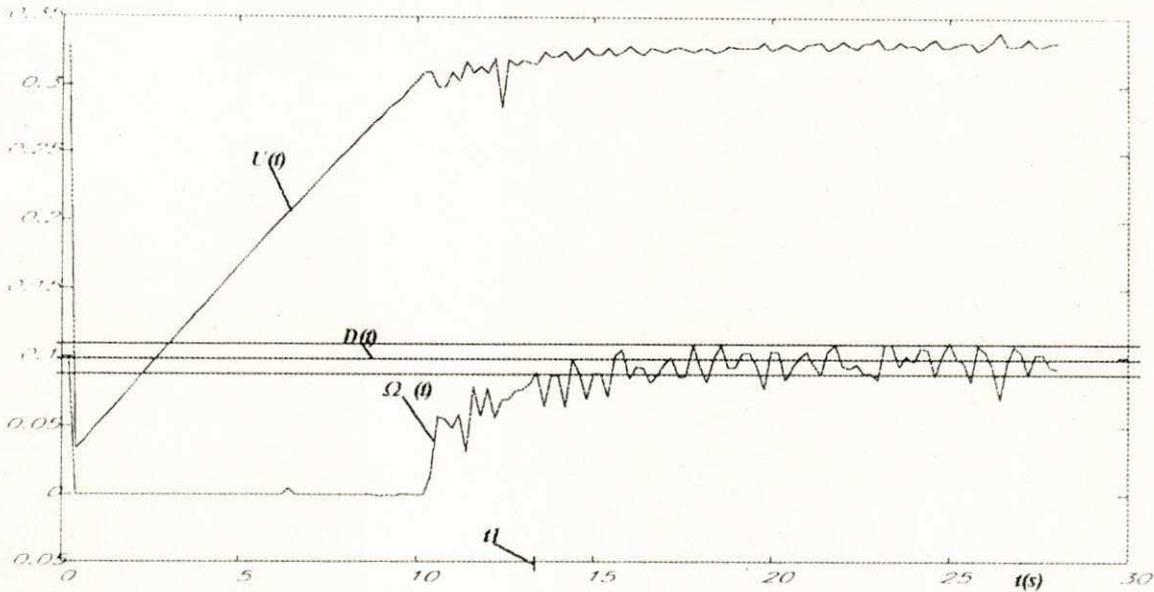


Figura 6.2 - Curvas experimentais obtidas na partida do motor CC

A partir da análise dos resultados experimentais do motor CC obtidos com o controlador neural adaptativo usando o Algoritmo 6.1, e mostrados na Figura 6.2, Lima et alii [1994] e Cavalcanti et alii [1994a] desenvolveram o Algoritmo 6.2 que considera todas as restrições na adaptação em tempo real da RNMC do controlador.

O Algoritmo 6.2 foi desenvolvido considerando o motor CC como o sistema a ser controlado, supõe-se que o conjunto sistema motor CC e controlador neural (sistema global) tem pelo menos um estado de equilíbrio [Levin & Narendra, 1993], e sem perda de generalidade, a origem tem sido considerada um desses estado de equilíbrio. Representando-se o sistema motor CC por  $x=f(x,u)$ , podem-se fazer as seguintes definições:

**Definição 6.1** - Um ponto  $x_e \in \mathcal{R}^n$  é um ponto de equilíbrio de  $x=f(x,u)$  se existir uma entrada  $u_e$  tal que:  $x_e = f(x_e, u_e)$  [Levin & Narendra, 1993].

O interesse principal está no conjunto planta e controlador neural. A RNMC da Figura 6.2 pode ser treinada, usando-se a tensão de excitação da armadura  $U$  e o estado  $X$ , para fornecer na sua saída o valor  $U$  do estado de equilíbrio. Baseado nessa estratégia, Cavalcanti et alii [1994a] e Lima et alii [1994] definiram o estado passivo do sistema global.

**Definição 6.2** - Um sistema dinâmico está no estado passivo quando estiver no ponto de equilíbrio com  $x_e = f(x_e, u_e)$  e o controlador neural garante a excitação de controle  $u_e$ .

Deve-se verificar a controlabilidade do sistema motor CC que pode ser especificada na definição 6.3.

**Definição 6.3** - Um sistema dinâmico é controlável, se, para os estados  $x_1$  e  $x_2$  em torno de um estado de equilíbrio, existe uma sequência de entrada de comprimento finito que transfere o sistema do estado  $x_1$  para o estado  $x_2$  sem sair da circunvizinhança do estado de equilíbrio [Levin & Narendra, 1993].

Quando a RNMC da Figura 6.1 estiver no estado passivo  $x=0$  isto é, quando as entradas do controlador neural forem zero ( $U(t-1)=\Omega(t)=\Omega(t-1)=D(t+1)=0.$ ), a RNMC fornecerá na sua saída  $U(t)=0$ .

No estado passivo da RNMC as seguintes condições devem ser satisfeitas:  $|U(t+1)-U(t)| < \xi$  para um  $\xi$  suficientemente pequeno, e se deve escolher os sinais de referência  $D(t)$  próximos de  $\Omega(t)$  para garantir uma diferença pequena entre eles. Essa condição pode ser especificada como  $|D(t)-\Omega(t)| < \varepsilon$ , para um  $\varepsilon$  suficientemente pequeno.

Lima et alii [1994] e Cavalcanti et alii [1994a] propuseram e implementaram experimentalmente o Algoritmo 6.2, que se diferencia do Algoritmo 6.1 por ser baseado no conceito de estado passivo, para treinamento seguido de controle em tempo real, da RNMC do controlador neural adaptativo direto da velocidade do motor CC. Usando-se o Algoritmo 6.2 não é necessário o treinamento prévio "off line" da RNMC com a dinâmica do motor CC.

#### Algoritmo 6.2

Controle neural "on line" usando o conceito de estado passivo

- 1) Geram-se aleatoriamente os pesos  $W_{inp}$  e  $W_{out}$  entre os limites 0.2 e -0.2. Atribuem-se valores iniciais aos parâmetros dos neurônios ( $\Theta = \beta = 1$  e  $T = 0$ ),  $\mu = 0.1$  ao fator de treinamento do algoritmo de propagação retroativa, e  $\chi=0.5$  ao fator de adaptação do controlador neural.
- 2) Treina a RNMC para o estado passivo zero e faz  $D(t)=0.1$  p.u.
- 3) Calcula:  $\Delta U(t) = \chi e(t+1)$  e  $U(t+1)=U(t)+ \Delta U(t)$ . Treina a RNMC com esse novo valor de  $U(t)$ .
- 4) Repete a etapa 3 até que  $\Omega(t)$  seja igual a 0.1 p.u.

As curvas apresentadas na Figura 6.2 podem ser obtidas tanto com o Algoritmo 6.1 quanto com o Algoritmo 6.2. Na figura 6.2 pode-se observar que o sinal de excitação  $U(t)$  cresce linearmente até o início do movimento do eixo do motor CC, e que, quando a velocidade do eixo do motor CC se aproxima da velocidade referência, o valor de  $U(t)$  fica aproximadamente

constante. O treinamento da RNMC com esse sinal de referência é feito até que a saída do motor CC tenha um valor próximo ao valor dentro do domínio do sinal de referência, representado pela região delimitadas pelas duas retas paralelas e equidistantes da reta que representa  $D(t) = 0.1$  p.u. Define-se tempo de transição ( $t_t$ ), representado no ponto  $t=13s$  na Figura 6.2, como o tempo necessário para o valor da saída da planta entrar no domínio do sinal de referência (representado pelas linhas paralelas à linha de  $D(t)=0.1$  da Figura 6.2). O crescimento linear de  $U(t)$  é explicado pelo não movimento do eixo do motor CC e o ajuste  $\Delta U(t) = \chi e^{(t+1)}$  em  $U(t)$  do controlador.

A seguir, mostram-se os resultados obtidos por Cavalcanti et alii [1994a] e Lima et alii [1994] usando o Algoritmo 6.2 para calcular alguns parâmetros da RNMC do controlador neural.

### 6.3 - Cálculo do valor ótimo do fator de adaptação do controlador neural

No Capítulo 3 foi mostrado o projeto da RNMC e foram discutidos como se calcular alguns parâmetros da RNMC. Nesta Seção mostra-se que o valor ótimo dos parâmetros da RNMC, usado no controlador neural adaptativo, pode ser determinado usando o Algoritmo 6.2. O fator de adaptação do controlador neural foi o primeiro parâmetro a ser determinado.

Cavalcanti et alii [1994c] apresentaram um método, usando o Algoritmo 6.2, para calcular o valor ótimo do fator de adaptação do controlador neural. Nesse método, inicialmente atribui-se à RNMC o estado passivo zero, estado em que as entradas do controlador neural são zero ( $U(t-1) = \Omega(t) = \Omega(t-1) = D(t+1) = 0.$ ) e com a RNMC fornecendo  $U(t) = 0$  na saída. A seguir, usando o Algoritmo 6.2 com a RNMC destreinada (valores aleatórios nos seus pesos e parâmetros), treinou-se a RNMC do controlador neural para atingir o estado passivo em que  $D(t) = 0.1$ . Usou-se a RNMC da Figura 6.1 com 6 neurônios na camada oculta,  $\mu = 0.3$  (fator de treinamento do algoritmo de propagação retroativa), e  $T_s = 20ms$  como tempo de amostragem do treinamento seguido de controle neural.

Para a determinação do valor ótimo do fator de adaptação do controlador neural foi escolhida uma tabela com diferentes valores de  $\chi$  [ 0.2; 0.4; 0.6; 0.8 e 1.0]. Para cada um desses valores de  $\chi$  foi executado o Algoritmo 6.2 obtendo diferentes curvas de  $U(t)$ , representadas como [  $U_2(t)$ ;  $U_4(t)$ ;  $U_6(t)$ ;  $U_8(t)$  e  $U_{10}(t)$  ], e diferentes curvas de  $\Omega(t)$  representadas como [  $\Omega_2(t)$ ;  $\Omega_4(t)$ ;  $\Omega_6(t)$ ;  $\Omega_8(t)$  e  $\Omega_{10}(t)$  ]. As curvas de  $U(t)$  e  $\Omega(t)$ , obtidas experimentalmente, são mostradas na Figura 6.3.

Observe-se na Figura 6.3 que, quanto maior for o fator de adaptação do controlador neural ( $\chi = 1.0$ ), mais rapidamente  $\Omega(t)$  se aproxima de  $D(t)$ , e menor é o tempo de transição. Observou-se que o aumento do fator de adaptação do controlador neural implica no aumento do ruído existente no sinal  $U(t)$  de controle (o que limita o valor de  $\chi$ ).

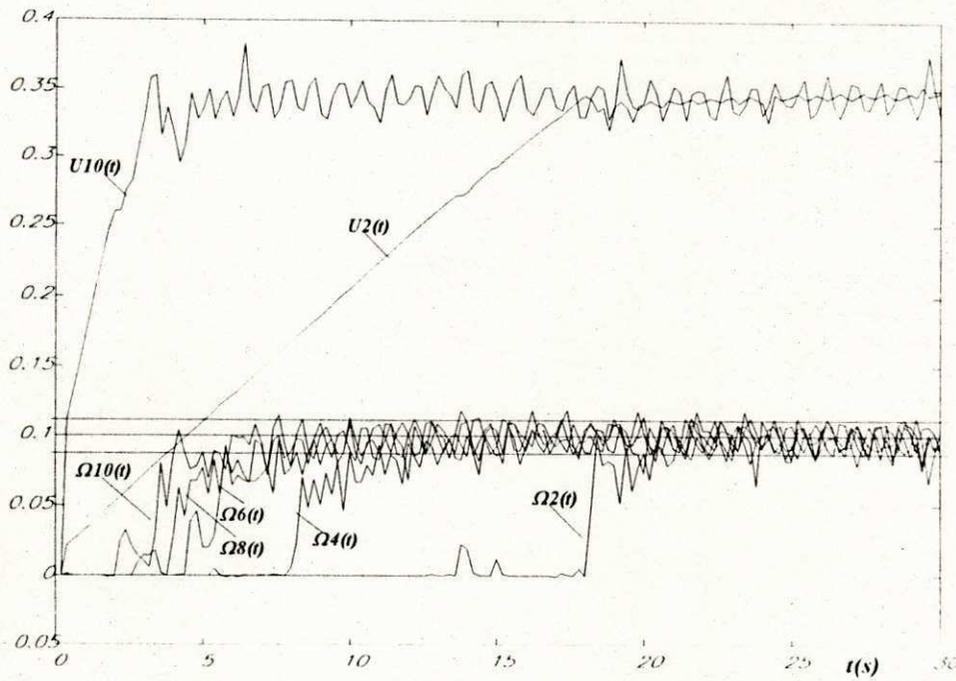


Figura 6.3 - Treinamento "on line" da RNMC com diferentes valores de  $\chi$ .

Na Figura 6.4 são mostrados os valores do tempo de transição (na ordenada) em função do fator de adaptação (na abscissa) obtidos a partir dos resultados experimentais mostrados na Figura 6.3.

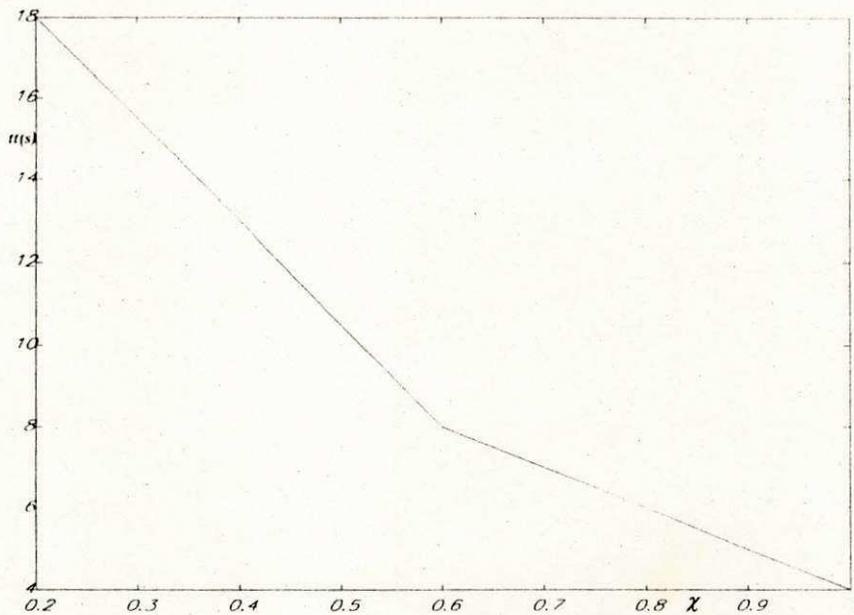


Figura 6.4 - Tempo para adaptação da RNMC versus fator de adaptação da RNMC

Observando-se a Figura 6.4, pode-se concluir que quanto maior for  $\chi$  ( $0 < \chi < 1$ ) mais rápido será a aprendizagem de um novo ponto passivo pela RNMC.

#### 6.4 - Cálculo do número ótimo de neurônios na camada oculta da RNMC

Os pesquisadores Villers & Barnard [1993] investigaram o desempenho de uma RNMC, usada em classificação de padrões, em função do número das camadas ocultas e concluíram que, se uma RNMC com única camada oculta satisfaz as especificações de uma aplicação em particular, a adição de uma outra camada oculta implica num mínimo ganho no desempenho do sistema neural. Considerando-se que uma única RNMC é suficiente para o controlador neural adaptativo direto, o problema é dimensioná-la isto é, determinar o número ótimo de neurônios nas suas camadas.

O problema de determinar o número ótimo de neurônios na camada oculta da RNMC foi investigado por diferentes grupos de pesquisas. Huang & Huang [1991] propuseram um método, baseado no número de elementos que definem o objetivo a ser conseguido com a RNMC, para calcular o número máximo de neurônios na camada oculta da RNMC. O algoritmo proposto por Azimi-Sadjadi et alii [1993], além da adaptação dos parâmetros da RNMC usando o algoritmo de propagação retroativa, emprega a declividade da curva do erro quadrático médio como o critério para a adição de um novo neurônio na camada oculta da RNMC.

Lima et alii [1994] propuseram a utilização do Algoritmo 6.2 para determinação do número ótimo de neurônios na camada oculta da RNMC. No experimento, usou-se a RNMC da Figura 6.1 com 6 neurônios na camada oculta,  $\mu = 0.3$  (fator de treinamento do algoritmo de propagação retroativa),  $\chi = 0.3$  (fator de treinamento do controlador neural) e  $T_s = 20ms$  como tempo de amostragem do treinamento seguido de controle neural.

Para a determinação experimental do valor ótimo do número de neurônios na camada oculta da RNMC foram escolhidos 3,4,5 e 6 neurônios. Para cada um desse número de neurônios foi executado o Algoritmo 6.2 obtendo duas curvas de  $U(t)$ , representadas como ( $U_3(t)$  e  $U_6(t)$ ), e duas curvas de  $\Omega(t)$  representadas como ( $\Omega_3(t)$  e  $\Omega_6(t)$ ). Essas curvas de  $U(t)$  e  $\Omega(t)$  são mostradas na Figura 6.5 com os tempos de transição  $t_3 = 20s$  e  $t_6 = 12s$ .

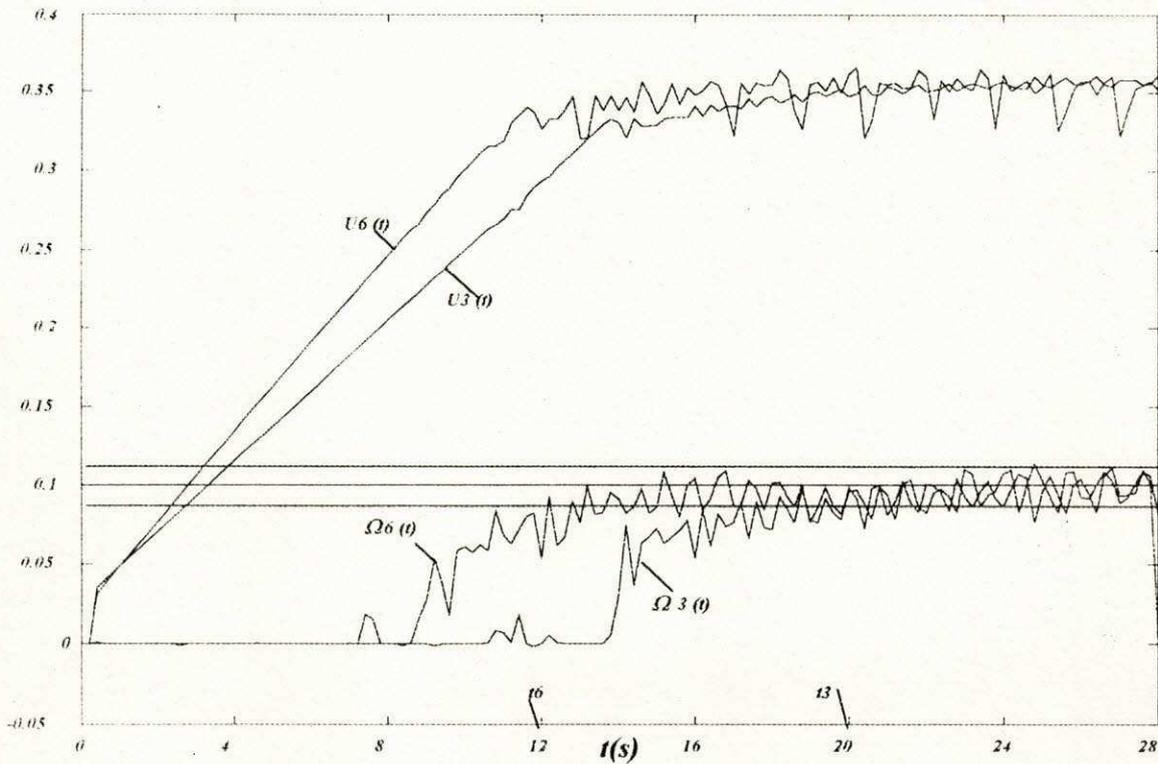


Figura 6.5 - Treinamento "on line" da RNMC com diferentes números de neurônios

Na Figura 6.6a é mostrada a curva experimental do tempo de transição ( $t_t(s)$ ) em função do número de neurônios ( $\#$ neurônios) na camada oculta da RNMC. Observe-se nessa figura que, quanto maior for o número de neurônios na camada oculta da RNMC mais rapidamente  $\Omega(t)$  se aproxima de  $D(t)$ , e menor é o tempo de transição.

A seguir, usando-se um simulador para o motor CC, os mesmos procedimentos foram executados para a determinação do tempo de transição com número de neurônios maiores que 6 na camada oculta da RNMC. Na Figura 6.6b apresenta-se a dependência do tempo de transição da RNMC em função do número de neurônios na camada oculta da RNMC. Esta curva foi obtida usando um simulador para o motor CC. Pode-se observar que o tempo de transição é inversamente proporcional ao número de neurônios na camada oculta da RNMC, e que quando o número de neurônios na camada oculta for maior do que 14 a redução no tempo de transição é mínima. Portanto, observando-se essas curvas, pode-se considerar como 14 o número ótimo de neurônios da RNMC.

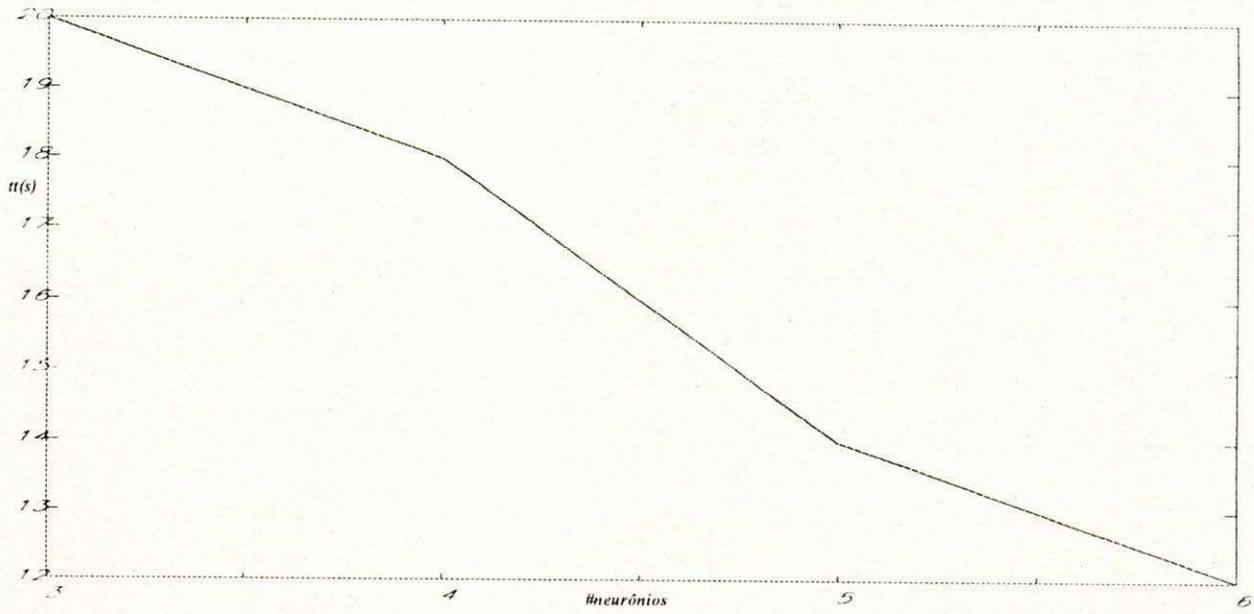


Figura 6.6a Tempo de transição da RNMC versus número de neurônios - experimental.

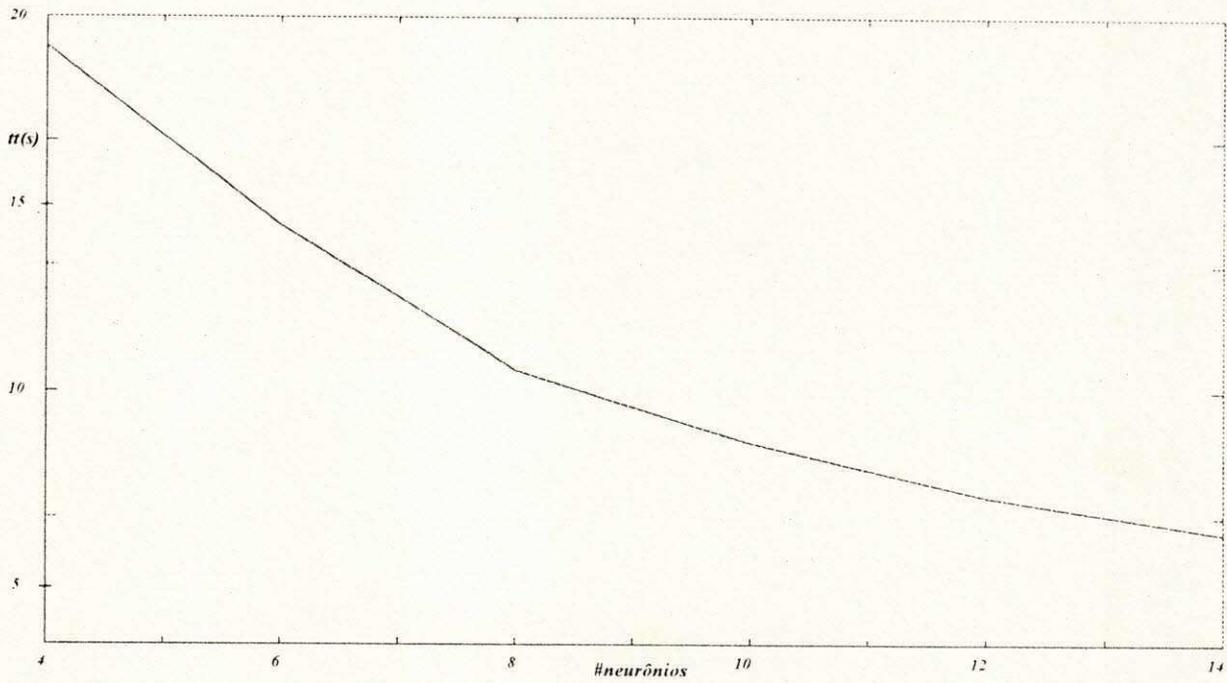


Figura 6.6b Tempo de transição da RNMC versus número de neurônios - simulado.

## 6.5 - Controle neural de um pêndulo invertido

Nesta Seção apresentam-se o projeto e resultados experimentais obtidos com o controlador neural adaptativo, desenvolvido a partir de regras fuzzy, no posicionamento de um pêndulo invertido.

### 6.5.1 - Controlador neural de posição

Os sistemas físicos do tipo pêndulo são utilizados frequentemente para demonstrar e verificar experimentalmente o desempenho de algoritmos de controle de sistemas não lineares. A escolha do pêndulo deve-se às suas características não lineares e pelo seu comportamento que apresenta modos instáveis e estáveis.

Alguns autores já estudaram como posicionar o pêndulo na posição vertical para cima (estabilizar o sistema pêndulo invertido).

Batur e Kasparian [1991] mostraram alguns resultados obtidos, a partir de simulação, de um sistema fuzzy usado para estabilizar um sistema carro e pêndulo invertido, sujeito a uma pequena variação no ângulo  $\theta$  do pêndulo em relação a normal. Ishida et alii [1991] mostraram uma RNMC usada para estabilizar um sistema carro e pêndulo invertido também sob pequenas variações no ângulo  $\theta$  entre o pêndulo e a normal.

Lin e Sheu [1992] mostraram resultados experimentais de um sistema híbrido, fuzzy e convencional, capaz de estabilizar um sistema carro e pêndulo invertido. Eles definiram os quadrantes de posicionamento do braço do pêndulo e mostraram como são desenvolvidas as regras do controlador fuzzy. Eles usaram um controlador convencional com realimentação para a sintonização fina do posicionamento do pêndulo quando o braço do pêndulo estiver na posição para cima.

Furuta e Yamakita [1991] apresentaram um sistema experimental de um pêndulo invertido e um controlador capaz de posicionar o braço do pêndulo na posição para cima. Eles usaram um modelo não linear para desenvolver o controlador para deslocar o braço do pêndulo até próximo da posição para cima, e usando um modelo linearizado do sistema pêndulo invertido, desenvolveram um controlador linear capaz de movimentar o pêndulo e posicionar o seu braço junto ao ponto de maior instabilidade.

Na Figura 6.7 é mostrado o esquema geral de um pêndulo simples. Na equação 6.5, equação referente ao movimento do pêndulo,  $P$  representa o peso,  $g$  aceleração da gravidade,  $L$  comprimento do braço do pêndulo, e  $\theta$  representa o ângulo entre o braço e a normal ao suporte do pêndulo. O torque resultante  $Tl(t)$  é definido na equação 6.5.

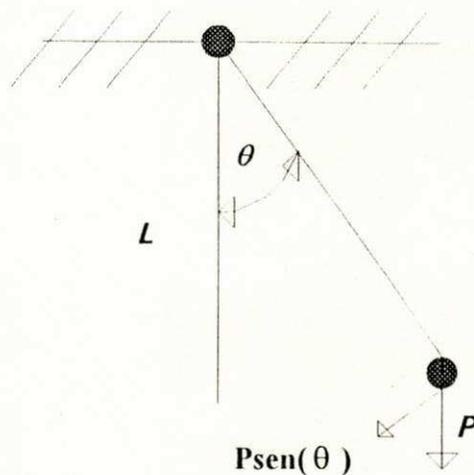


Figura 6.7 - Esquema geral do pêndulo simples.

$$Tl(t) = (P g) L^2 \frac{d^2 \theta(t)}{dt^2} + P L \text{sen}(\theta(t)) \quad (6.5)$$

Na Figura 6.8 é mostrado o esquema da montagem de um braço rígido, acoplado a um peso (círculo menor na Figura 6.8) e ao eixo do motor CC (círculo maior da Figura 6.8). Nessa montagem o eixo do motor é visto de perfil. No esquema da Figura 6.8 são mostrados 4 posições especiais do braço do pêndulo, representadas pelos ângulos  $\theta$  entre o braço do pêndulo e a normal. Um pêndulo é dito ser um pêndulo invertido se é permitido o seu posicionamento no ângulo  $\theta=0$  (ou  $3\pi/2 < \theta < \pi/2$ ), como mostrado na Figura 6.8a. Considera-se que os ângulos são medidos no sentido horário.

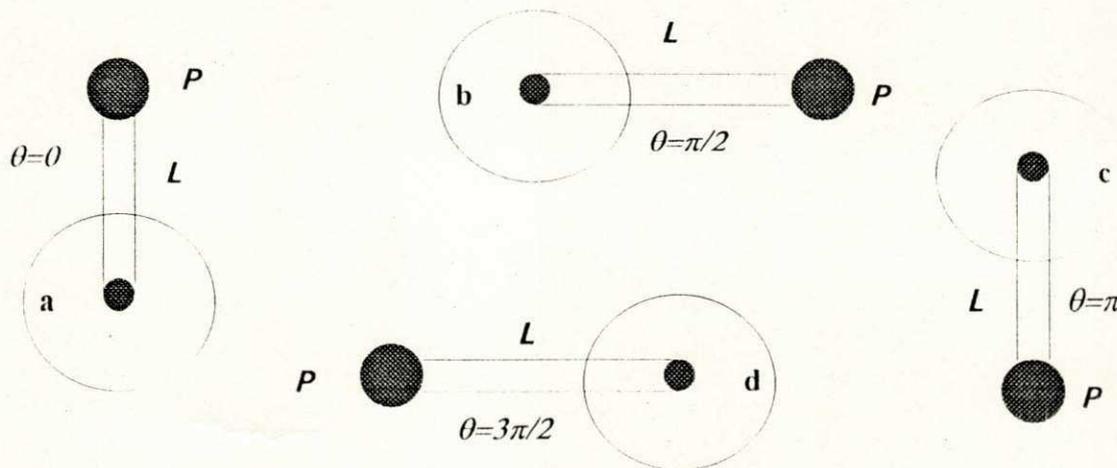


Figura 6.8 - Posicionamento do eixo do motor CC.

Usou-se o controlador neural adaptativo direto, funcionando como controlador de posição, para o posicionamento do braço do pêndulo nas posições escolhidas pelo operador. No projeto do controlador neural, assume-se que  $\theta_r(t+1)$  representa o valor de referência ou o valor

desejado de  $\theta(t+1)$  para a posição do eixo do motor CC.  $\theta_r(t+1)$  e  $\theta(t+1)$  são representados em p.u. com valor de base  $2\pi$ ,  $U(t)$  representado em p.u. com valor de base 12V, e  $\Omega(t)$  representado no sistema p.u. com valor de base 2rps. Usou-se a equação 6.6 para cálculo de  $PD(t+1)$ , definido como a ponderação da diferença entre a posição referência e a posição real do eixo do motor CC. Na Figura 6.9 é mostrado o controlador neural direto de posição do eixo do motor CC com  $X1=U(t)$ ,  $X2=\Omega(t)$ ,  $X3=PD(t+1)$ ,  $X4=seno[2\pi\theta_r(t+1)]$  e  $X5=seno[2\pi\theta(t+1)]$  como entradas da RNMC (todos os valores representados em p.u.).

$$PD(t+1) = P_d[\theta_r(t) - \theta(t)] \tag{6.6}$$

O índice de desempenho de adaptação da RNMC do sistema motor CC e pêndulo invertido é definido na equação 6.7.

$$E = \frac{1}{2}(e_{t+1})^2 = \frac{1}{2}[\theta_r(t+1) - \theta(t+1)]^2 \tag{6.7}$$

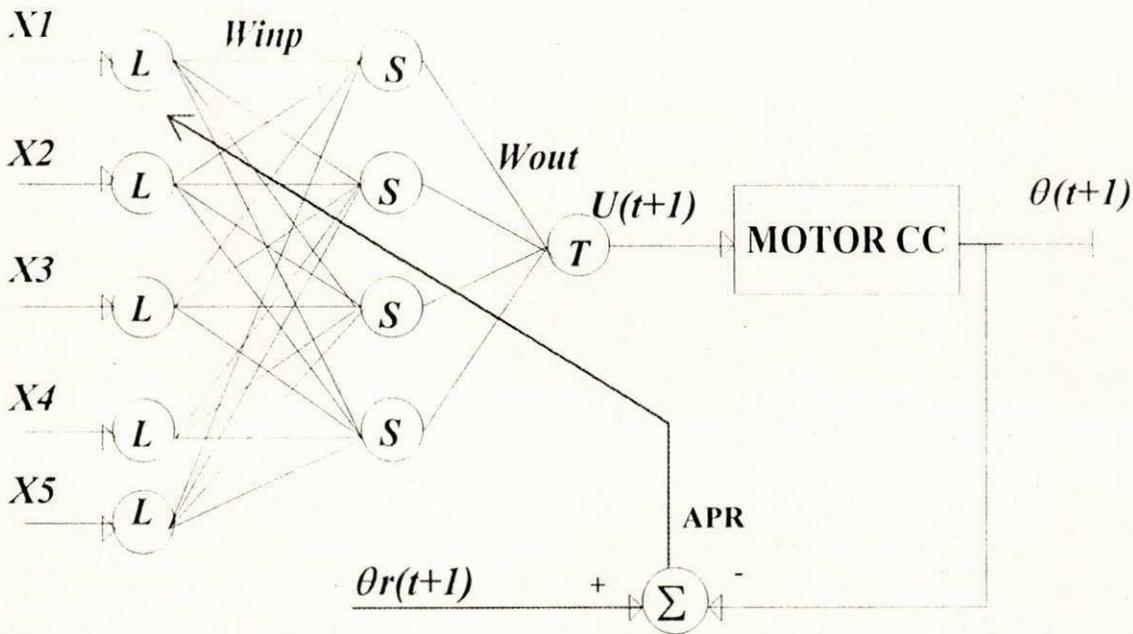


Figura 6.9 - Controlador neural de posição do eixo do motor CC.

### 6.5.2 - Controle neural de posição do pêndulo

Usando o Algoritmo 6.2, decidiu-se posicionar o braço do pêndulo próximo de  $\pi/2$  rd. Este teste foi realizado com  $L = 0.01$  m, e  $P = 0.1$  Kg. Com esses valores e para ângulo  $\theta(t) = \pi/2$  rd (Figura 6.8b), que é o ponto de maior torque, tem-se um torque  $Tl(t)$  menor que o atrito viscoso do eixo do motor CC. Isto é, o eixo do motor permanecerá em repouso para qualquer posição do pêndulo.

Na Figura 6.10 são mostrados os resultados experimentais obtidos com o controlador neural e o Algoritmo 6.3, usando  $P_d=2$  na equação 6.6. Os resultados apresentados nessa figura foram obtidos após o treinamento com a apresentação da curva  $\theta_r(t)$ , durante 20 intervalos de adaptação, da RNMC da Figura 6.9. Na Figura 6.10 são mostradas as curvas da posição referência ( $\theta_r(t) = \pm 0.25 \text{ p.u.}$ ), curva da posição real ( $\theta(t)$ ), e curva da tensão de armadura ( $U(t)$ ) do motor CC em função do tempo (representado em milisegundos). Observe-se que o controlador neural não foi capaz de posicionar  $\theta(t)$  exatamente em  $\theta_r(t)$ . Isso se explica pelas imperfeições mecânicas e outras limitações existentes na montagem.

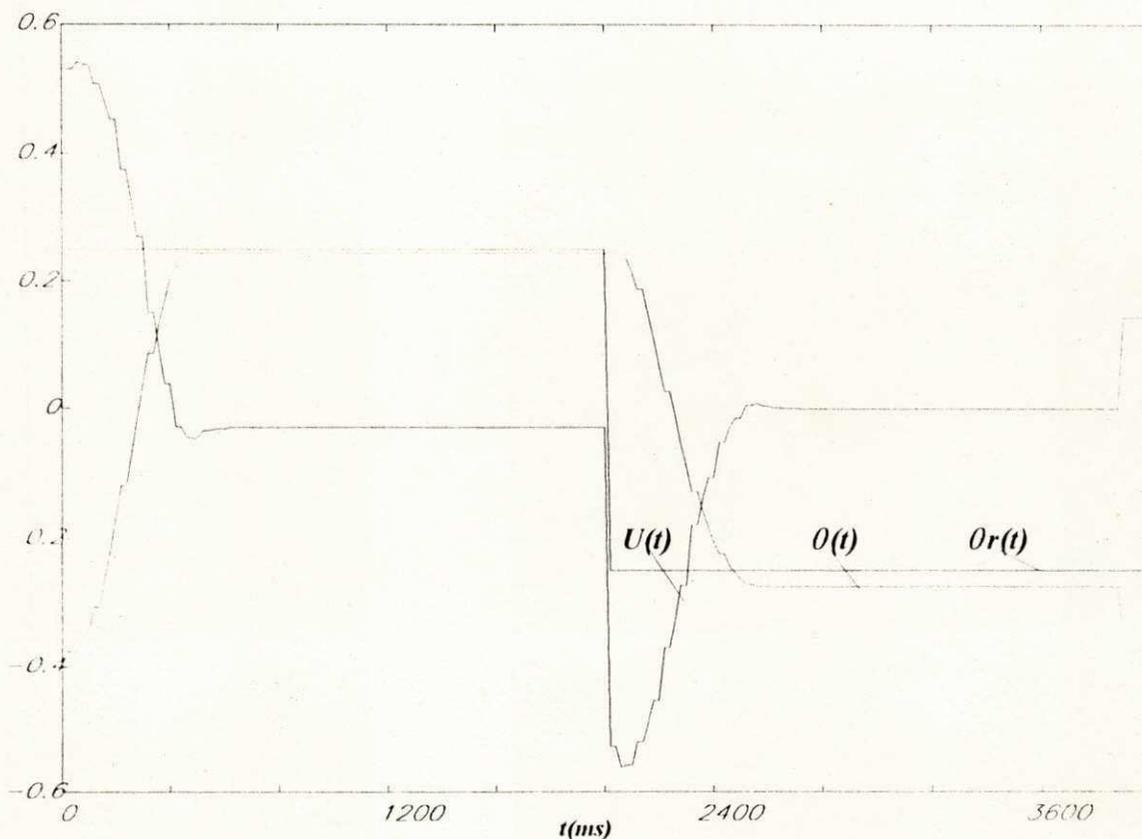


Figura 6.10 - Resultados experimentais do controle de posição do pêndulo

### 6.5.3 - - Posicionamento do pêndulo invertido.

Na segunda experiência aumentou-se o comprimento do braço do pêndulo para  $L = 0.1 \text{ m}$  e manteve-se constante o peso ( $P = 0.1 \text{ Kg}$ ). Esses valores de  $P$  e  $L$ , para determinados valores do ângulo  $\theta(t)$  resultam num torque  $Tl(t)$  maior que o atrito viscoso do eixo do motor CC.

Para o desenvolvimento do Algoritmo 6.3 de treinamento e controle da RNMC, e das regras fuzzy, do controlador neural do pêndulo, usou-se a equação 6.6 do torque da carga do pêndulo. Nessa equação, o torque da carga  $Tl(t)$  depende do deslocamento e também da aceleração do pêndulo.

A partir das Figura 6.8 e Figura 6.11a, e considerando o pêndulo em repouso (estado estável) podem ser observadas as seguintes características do sistema pêndulo invertido:

1) O pêndulo estará no estado instável em  $\theta(t) = 0$  rd (Figura 6.8a). Qualquer deslocamento do pêndulo fará  $Tl(t) \neq 0$ .

2) Com o pêndulo em repouso, o torque  $|Tl(t)|$  será máximo para  $\theta(t) = \pm\pi/2$  rd (ver Figura 6.8b e Figura 6.8d).

3) O pêndulo estará no estado estável com  $\theta(t) = \pm\pi$  rd (Figura 6.8c). Qualquer movimento no pêndulo em torno de  $\theta(t) = \pm\pi$  rd o fará retornar ao ponto  $\theta(t) = \pm\pi$  rd.

4)  $|Tm(t)|$  deverá aumentar para estabilizar (posicionar e imobilizar) o pêndulo quando  $\theta(t)$  crescer de  $0 \rightarrow \pi/2$  ou decrescer de  $0 \rightarrow -\pi/2$ .

5)  $|Tm(t)|$  deverá diminuir para estabilizar o pêndulo quando  $\theta(t)$  crescer de  $\pi/2 \rightarrow \pi$  ou decrescer de  $-\pi/2 \rightarrow -\pi$ .

Na Figura 6.11a é mostrado um círculo com quatro quadrantes representando o torque de carga  $Tl(t)$  gerado pelo pêndulo (ver [Lin & Sheu, 1992]) para diferentes ângulos  $\theta(t)$  do pêndulo. O torque  $Tm(t)$  gerado pelo motor CC (diretamente proporcional à tensão da armadura do motor CC) para estabilizar o pêndulo deve ter a mesma intensidade de  $Tl(t)$  mas com sinal contrário. Os quadrantes foram definidos a partir do ponto de instabilidade com  $\theta(t)=0$ . Q1 foi definido no intervalo  $0 < \theta(t) < \pi/2$ , Q2 foi definido para  $\pi/2 < \theta(t) < \pi$ , Q3 foi definido no intervalo  $\pi < \theta(t) < 3\pi/2$  e Q4 foi definido no intervalo  $3\pi/2 < \theta < 2\pi$ . Os quadrantes podem ser vistos na função de pertinência  $\theta_m$  de  $\theta(t)$  mostrada na Figura 6.11b.

Considerou-se o torque  $Tl(t)$  positivo nos quadrantes Q3 e Q4 e torque  $Tl(t)$  negativo, nos quadrantes Q1 e Q2. Isto é, o torque  $Tm(t)$ , a ser gerado pelo motor CC para estabilizar o pêndulo, deve ser positivo quando o braço do pêndulo estiver posicionado nos quadrantes Q1 e Q2 e deve ser negativo quando o braço do pêndulo estiver posicionado nos quadrantes Q3 e Q4.

O posicionamento  $\theta(t)$  do eixo do motor CC segue uma referência  $\theta_r(t)$ . O valor da tensão de armadura  $U(t)$  é modificado de acordo com as equações 6.3 e 6.4. No caso do pêndulo

invertido, para cada quadrante do círculo de torque da Figura 5a, existem diferentes valores de  $\Delta U(t)$  e de  $U(t)$ . Para o pêndulo invertido o valor exato de  $\Delta U(t)$  não é conhecido.

Baseado nas características do pêndulo invertido, no círculo de torque da Figura 6.11a, e na função de pertinência da Figura 6.11b, foram desenvolvidas as seguintes regras fuzzy para o treinamento seguido de controle neural ( $\theta_m$  representa a função de pertinência dos quadrantes de  $\theta(t)$ , “==” significa comparação):

1) Quando o braço do pêndulo estiver em Q1 ou Q2,  $U(t)$  deve ser positivo;

if  $\theta_m == Q1$  or  $\theta_m == Q2$  then  $U(t) > 0$ ;

2) Quando o braço do pêndulo estiver em Q3 ou Q4,  $U(t)$  deve ser negativo;

if  $\theta_m == Q3$  or  $\theta_m == Q4$  then  $U(t) < 0$ ;

3) Quando o braço do pêndulo estiver em Q2,  $U(t)$  será decrescente para posicionar o pêndulo de  $\theta = \pi/2$  rd a  $\theta = \pi$  rd;

if  $\theta_m == Q2$  then  $\Delta U(t) < 0$ ;

4) Quando o braço do pêndulo estiver em Q1,  $U(t)$  será crescente para posicionar o pêndulo de  $\theta = 0$  rd a  $\theta = \pi/2$  rd;

if  $\theta_m == Q1$  then  $\Delta U(t) > 0$ ;

5) Quando o braço do pêndulo estiver em Q3,  $U(t)$  será decrescente para posicionar o pêndulo de  $\theta = \pi$  rd a  $\theta = 3\pi/2$  rd;

if  $\theta_m == Q3$  then  $\Delta U(t) < 0$ ;

6) Quando o braço do pêndulo estiver em Q4,  $U(t)$  será decrescente para posicionar o pêndulo de  $\theta = 0$  rd a  $\theta = -\pi/2$  rd;

if  $\theta_m == Q4$  then  $\Delta U(t) < 0$ ;

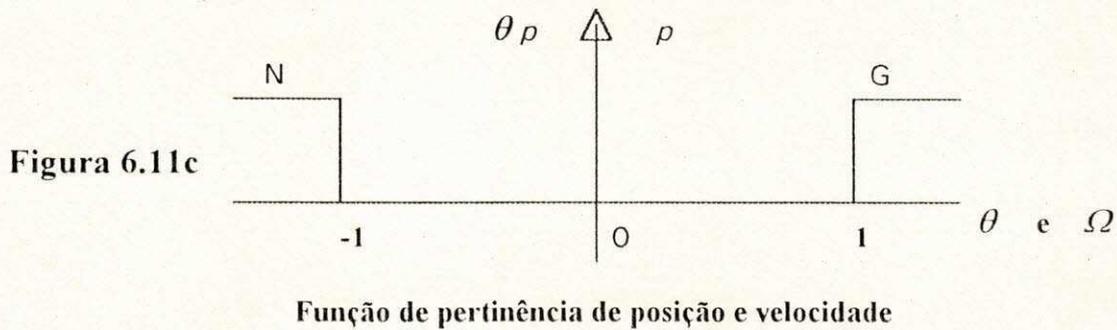
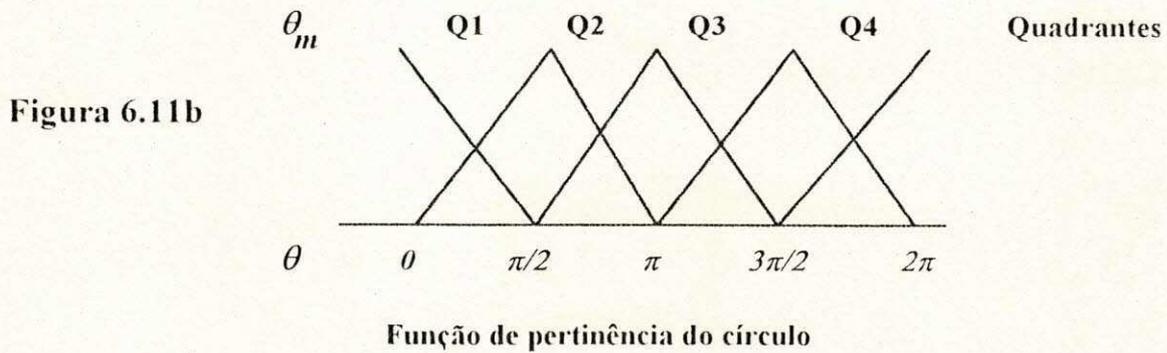
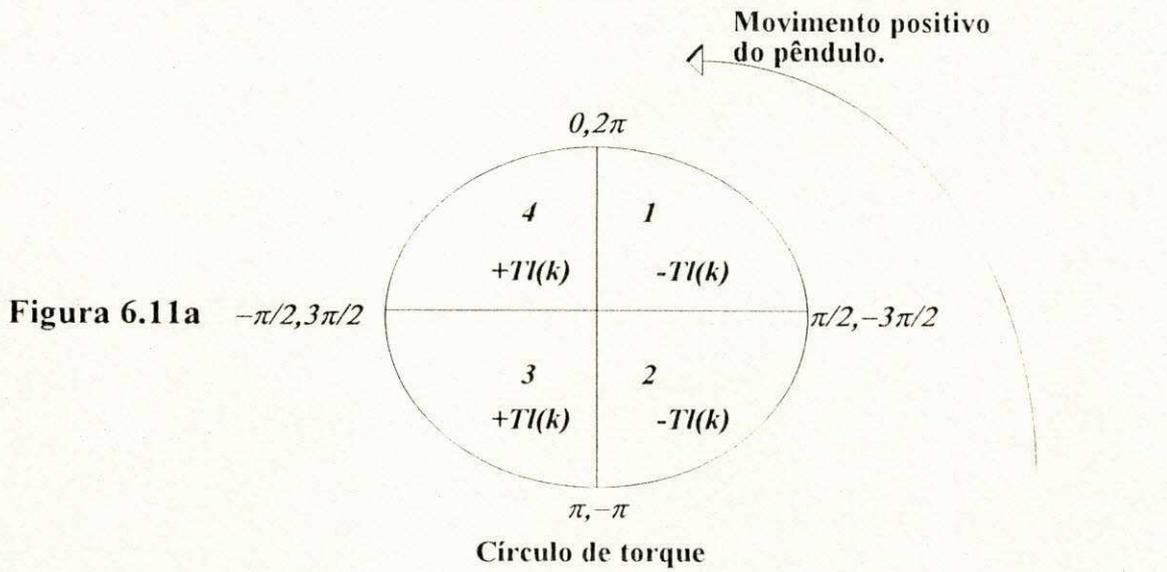


Figura 6.11 - Círculo de torque e função de pertinência de  $\theta(t)$  e  $\Omega(t)$ .

Durante o treinamento do controlador fuzzy/neural foram seguidas algumas regras adicionais, que não podem ser consideradas necessariamente regras fuzzy, obtidas do círculo de torque da Figura 6.11a.

1) Não foram usados como referência os valores de  $\theta_r(t) = \pm\pi/2$  rd que é o ponto de maior torque gerado pelo pêndulo. Nesses pontos, a corrente na armadura do motor CC se aproxima do seu limite máximo.

2) Treinou-se a RNMC do controlador neural com valores do ângulo de referência próximo do ponto de estabilidade do pêndulo, ou  $\theta_r(t) = \pm\pi$  rd (ver a Figura 6.12a)

3) Os valores de  $\theta_r(t)$  foram gerados para garantir os estados passivos durante o treinamento da RNMC. Isso garantiu que o pêndulo se movimentasse lentamente o que permitiu a simplificação na geração dos sinais usados no treinamento da RNMC.

O Algoritmo 6.3 foi desenvolvido a partir das características do sistema pêndulo invertido, para treinamento seguido de controle da RNMC. Na Figura 6.12a são mostrados os ângulos  $\theta_r(t)$  usados no treinamento, em torno de  $\theta(t) = \pi$ , da RNMC do controlador neural.

A partir das características do pêndulo invertido, o treinamento do controlador neural, com as posições mostradas na Figura 6.12a, pode ser estendido para as posições mostradas na Figura 6.12b. Isto é, o treinamento da RNMC no quadrante Q2 é equivalente ao treinamento no quadrante Q1, o treinamento da RNMC no quadrante Q3 é equivalente ao treinamento no quadrante Q4. Usando as características do pêndulo, é possível estender o treinamento da RNMC nos quadrantes Q2 e Q3 para todos os quadrantes. O uso das funções  $seno()$  nas entradas  $X4 = seno[2\pi\theta_r(t+1)]$  e  $X5 = seno[2\pi\theta(t+1)]$  da RNMC, permite a equivalência no treinamento entre as posições mostradas na Figura 6.12a e Figura 6.12b, sem a necessidade do desenvolvimento de novas regras fuzzy.

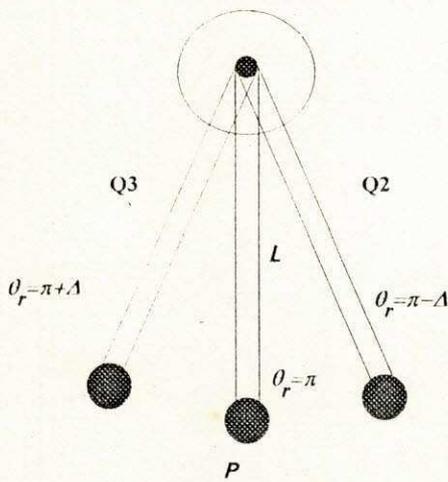


Figura 6.12a

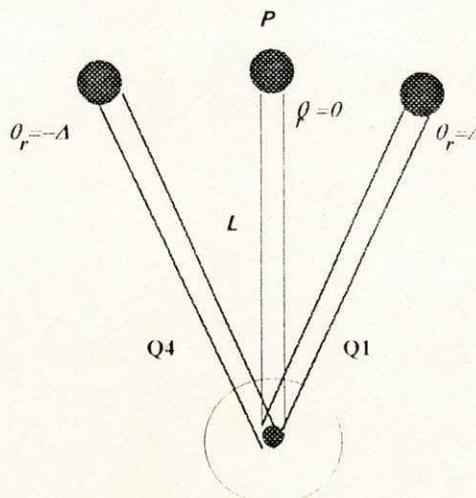


Figura 6.12b

Figura 6.12 - Posições de referência do braço do pêndulo para treinamento da RNMC.

Algoritmo 6.3

Algoritmo do treinamento da RNMC com a dinâmica do pêndulo

ETAPA 1 - Inicialização dos parâmetros da RNMC.

1) Geram-se aleatoriamente os pesos  $W_{in}$  e  $W_{out}$  entre os limites 0.2 e -0.2. Atribuem-se valores iniciais aos parâmetros dos neurônios ( $\Theta = \beta = 1$  e  $T = 0$ ),  $\mu = 0.1$  ao fator de treinamento do algoritmo de propagação retroativa, e  $\chi = 0.5$  ao fator de adaptação do controlador neural.

ETAPA 2 - Treinamento em torno do ponto de estabilidade representado com  $\theta = \pi$ .

2) Obtém um intervalo dos valores de referência  $\theta^*(t)$ .

3) Usa  $\theta^*(t)$  capaz de calcular o novo  $U(t)$  do ponto passivo da RNMC. Calcula:  $\Delta U(t) = \chi e^{(t+1)}$  e  $U(t+1) = U(t) + \Delta U(t)$ . Treina a RNMC com esse novo valor de  $U(t)$ .

4) Repetem-se as etapas 2 e 3 até que  $|\theta(t) - \theta^*(t)| < \varepsilon$ , quando  $\theta^*(t)$  for constante e  $\varepsilon$  suficientemente pequeno.

ETAPA 3 - Treinamento "off line" do estado passivo futuro

5) Treina a RNMC com as entradas:  $X1 = 1$ ,  $X2 = 0$ ,  $X3 = -0.5$ ,  $X4 = 0$  e  $X5 = 0$ .

6) Usa  $U(t+1) = 1$  como valor desejado da saída da RNMC.

A RNMC foi treinada durante 20 intervalos, usando o ETAPA 2 do Algoritmo 2, para valores de  $\theta(t)$  variando de  $\pi - \pi/10 \text{ rd} \leq \theta(t) \leq \pi + \pi/10 \text{ rd}$ . A curva experimental obtida para posicionamento do braço do pêndulo próximo de  $\theta = \pi$  é mostrada na Figura 6.13. Observe que a RNMC gera um valor residual de  $U(t)$ , representado como  $U_r(t)$ , capaz de estabilizar o braço do pêndulo próximo a  $\theta(t) = \pi$ .

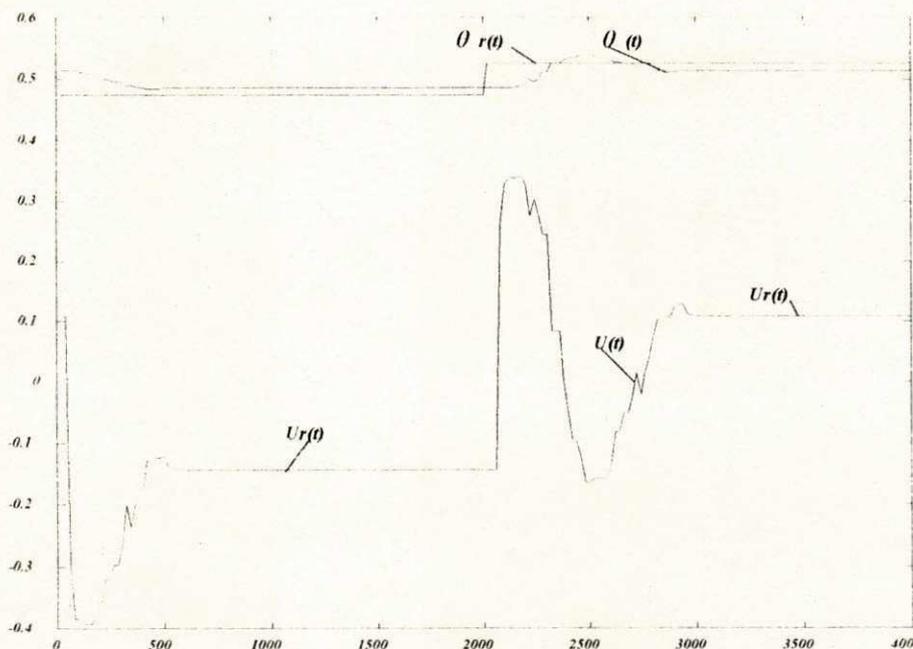


Figura 6.13 - Movimento do braço do pêndulo próximo de  $\theta = \pi \text{ rd}$ .

A utilização do conceito de estado passivo (ver o Algoritmo 6.3) permite o treinamento da RNMC com a dinâmica do pêndulo sem a consideração do termo da aceleração  $[(P/g)L^2 d^2\theta(t)/dt^2]$  da equação 6.5. Para segurança no movimento do pêndulo foram desenvolvidas regras fuzzy para o posicionamento e velocidade do eixo do motor CC. Para o posicionamento do eixo do motor, ver Figura 6.11c, definiu-se a variável fuzzy  $\theta_p$  com a função de pertinência  $\theta_p = G$  quando  $\theta(t) > 1$ , e  $\theta_p = N$  quando  $\theta(t) < -1$ . Para a velocidade, definiu-se a variável fuzzy  $\Omega_p$  com a função de pertinência  $\Omega_p = G$  quando  $\Omega(t) > 1$  rps, e  $\Omega_p = N$  quando  $\Omega(t) < -1$  rps.

7) Limite nos valores do ângulo  $\theta(t)$  em p.u.

**if  $\theta_p == G$  then  $\theta(t) = \theta(t) - 1$**

8) Limite nos valores do ângulo  $\theta(t)$  em p.u.

**if  $\theta_p == N$  then  $\theta(t) = \theta(t) + 1$**

9) Limite na velocidade do braço do pêndulo.

**if  $\Omega_p == G$  then  $U(t) = 0$**

10) Limite na velocidade do braço do pêndulo.

**if  $\Omega_p == N$  then  $U(t) = 0$**

#### 6.5.4 - Posicionamento do pêndulo invertido instável na posição para cima.

Devido a simetria no movimento do pêndulo só foi necessário o treinamento da RNMC numa determinada região de movimento do braço do pêndulo. Sabe-se que existem dois pontos de máxima carga no posicionamento do braço do pêndulo. A estratégia foi treinar a RNMC para posicionamento do braço do pêndulo até bem próximo ao ponto de máximo torque. Para o braço ser movido da posição "para baixo" até a posição "para cima", o braço do pêndulo deverá passar por um dos pontos de máxima carga. Decidiu-se acelerar o braço do pêndulo a partir da posição de repouso e na direção anti-horária para que, com a aceleração obtida, o braço consiga ultrapassar a barreira do máximo torque, posição em que  $\theta(t) = \pi/2$ .

Definiu-se o "estado passivo futuro" como o estado passivo a ser alcançado pelo sistema pêndulo e controlador. Escolheu-se o estado passivo em  $\theta^*(t) = \pi$  (ou 0.5p.u.) e estado passivo futuro em  $\theta^*(t) = 0$ . O valor de controle deve ser máximo ( $U(t) = 1$ .) para ser atingido o "estado passivo futuro". A RNMC foi treinada 20 vezes (ETAPA 3 do Algoritmo 2) com os seguintes valores:  $X1 = U(t) = 1$ ,  $X2 = \Omega(t) = 0$ ,  $X3 = D(t+1) = -0.5$ ,  $X4 = \text{seno}[2\pi\theta^*(t+1)] = 0$  e  $X5 = \text{seno}[2\pi\theta(t+1)] = 0$ .

O eixo do motor CC foi posicionado com o braço do pêndulo em  $\theta(t) = \pi$  (ou  $\theta(t)=0.5p.u.$ ) e foi usado  $\theta_r(t) = 0$  como posição referência. Na Figura 6.14 são mostradas as curvas obtidas no movimento do braço do pêndulo da posição com  $\theta(t) = \pi$  (ou  $\theta(t)=0.5p.u.$ ) até  $\theta(t)=0$ . Observe a variação no deslocamento do braço do pêndulo em torno de  $\theta(t) = 0$  e o pequeno valor residual de  $\theta(t)$  em relação a  $\theta_r(t)$ . Esse valor residual é zerado com a adaptação em tempo real do controlador neural (não mostrado na Figura 6.14). Na Figura 6.15 são mostradas as posições ocupadas pelo pêndulo durante o movimento para o seu equilíbrio. Essas posições são indicadas por  $p0, p1, p2, p3, p4$  e  $p5$ .

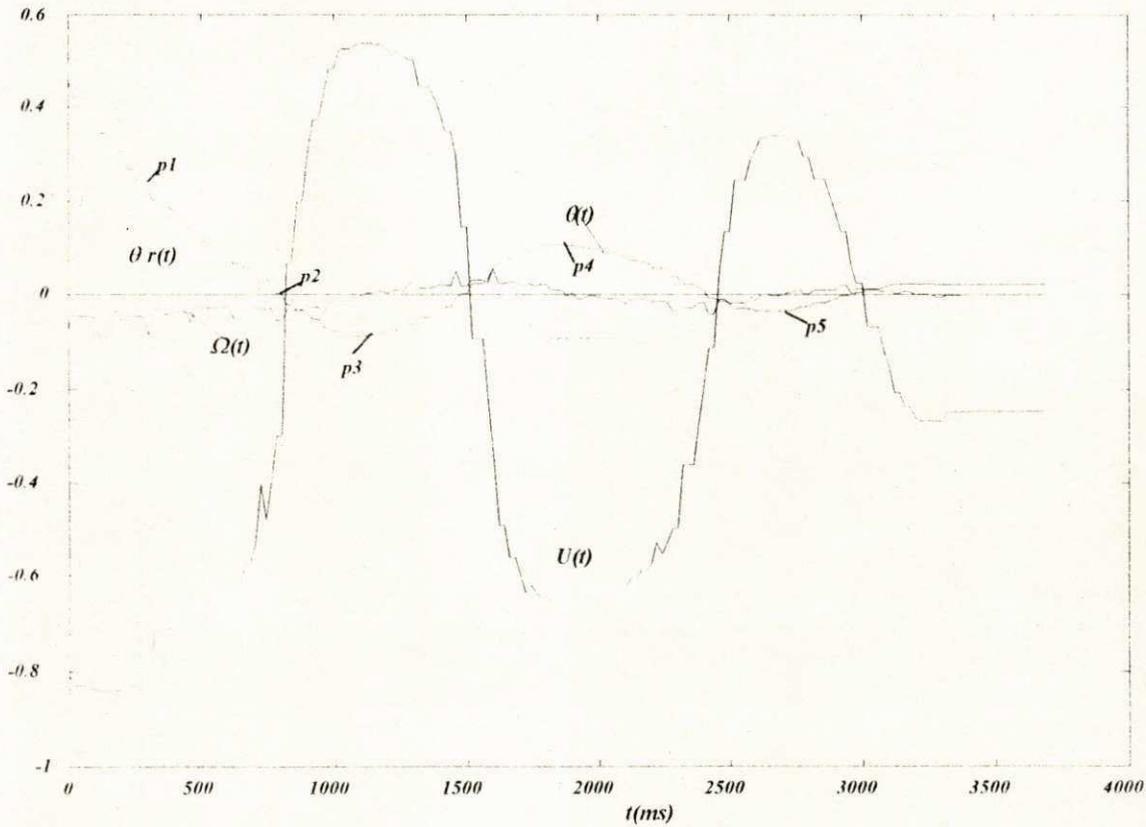


Figura 6.14 - Posicionamento em  $\theta(t)=0$  do pêndulo invertido.

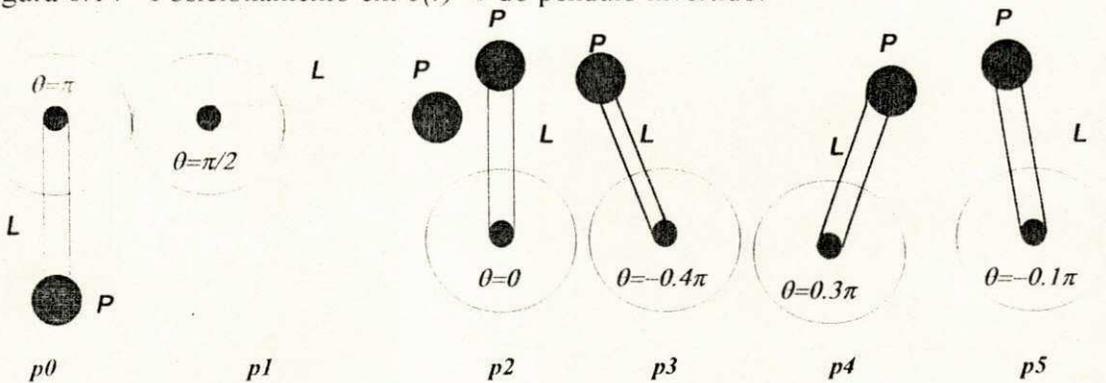


Figura 6.15 - Posições ocupadas pelo pêndulo para o seu posicionamento em  $\theta(t)=0$ .

## 6.6 - Conclusão

Apresentou-se um algoritmo para treinamento em tempo real da RNMC do controlador neural adaptativo direto sem a necessidade do seu prévio treinamento "off line". A seguir, usando o conceito de estado passivo, sugeriu-se uma forma de se encontrar o número ótimo de neurônios na camada oculta da RNMC e para o cálculo do fator de adaptação do controlador neural direto.

Por último, mostrou-se que o controlador neural direto é capaz de estabilizar um pêndulo invertido. O treinamento do controlador neural direto de posição foi feito somente com o conhecimento do jacobiano do sistema motor CC e das características fuzzy da posição e do torque do braço do pêndulo invertido. Só foi necessário o treinamento da RNMC para movimentar o braço do pêndulo entre os quadrantes  $Q2$  e  $Q3$ .

## CONCLUSÕES E PERSPECTIVAS FUTURAS

Nesta Tese mostrou-se que um controlador neural adaptativo direto, sob certas condições e com uma única RNMC, é capaz de aprender a dinâmica e controlar um sistema tão complexo quanto um sistema pêndulo invertido.

Inicialmente, mostrou-se o projeto e implementação de um sistema motor CC a ser usado nos testes experimentais dos controladores de velocidade e de posição do eixo do motor. Para o sistema motor CC, foram implementados controladores de velocidade do tipo PI, e controladores adaptativos baseados em modelos de referência. Para o mesmo sistema, também foram implementados controladores de posição do tipo PD.

A seguir, após a escolha da arquitetura da RNMC para ser usada como controlador neural, foram feitas simulações para se estudar a capacidade de aprendizagem da dinâmica da planta pela RNMC. A partir dos resultados dessas simulações, concluiu-se que a sequência dos sinais usados para treinamento da RNMC deve ser simétrica.

A seguir, foram feitos testes experimentais de controladores neurais direto com a RNMC treinada "off line" com a dinâmica inversa da planta. Mostrou-se experimentalmente que, quando o sinal de referência for constante e usando um sinal de controle fixo, é possível a RNMC INVERSA, usada como controlador, aprender a nova dinâmica da planta. Sabendo-se

que controladores neurais direto só podem ser treinados em tempo real se for conhecido o jacobiano da planta, implementou-se o controlador neural indireto, em que uma RNMC é treinada "off line" com a dinâmica da planta, e uma outra RNMC usada como EMULADOR, é usada para fornecer o jacobiano e permitir a adaptação em tempo real da RNMC controladora. Ainda usando duas RNMC, implementou-se um outro tipo de controlador neural adaptativo que considera a planta como sendo não linear. A partir de resultados verificados experimentalmente com esses tipos de controladores neurais, mostrou-se que, sob certas condições, o controlador neural direto pode ser adaptado "on line" e que não é necessário o conhecimento do valor exato do jacobiano da planta para a adaptação do controlador neural direto. Essa metodologia de adaptação "on line" do controlador neural pode ser considerada como uma contribuição original à área dos controladores neurais.

Baseado nos resultados experimentais obtidos com os controladores convencionais e controladores neurais no controle da velocidade e posição do motor CC, foram comparados quantitativamente e qualitativamente o desempenho desses controladores. Concluiu-se que se for possível a adaptação em tempo real do controlador neural direto, ele é o mais indicado dos controladores neurais, pois só utiliza uma única RNMC.

Após a escolha da arquitetura do controlador neural, definiu-se o conceito de estado passivo de uma planta e se desenvolveu um algoritmo, baseado em algumas condições no sinal de referência e um conhecimento mínimo da dinâmica da planta, que permite o treinamento em tempo real da RNMC do controlador neural sem o seu prévio treinamento "off line". O conceito de estado passivo pode ser considerado uma contribuição original à área dos controladores neurais. Usando o conceito de estado passivo foram desenvolvidos procedimentos e implementados experimentalmente algoritmos em tempo real para a estimação do fator de adaptação da RNMC e para determinação do número ótimo dos neurônios da camada oculta da RNMC. Para estudar a capacidade do controlador neural adaptativo direto usando o conceito de estado passivo, decidiu-se controlar o motor CC no posicionamento de um pêndulo. A partir dos resultados experimentais obtidos no posicionamento do eixo do motor CC, mostrou-se que, com algumas regras fuzzy desenvolvidas a partir do comportamento físico do pêndulo (regras baseadas na equação do pêndulo), juntamente com o conceito de estado passivo e de estado passivo futuro, e usando o controlador neural adaptativo, é possível a estabilização de sistemas não lineares representado por um braço do pêndulo invertido.

Atualmente, estuda-se a aplicação do conceito de estado passivo para, em tempo real, projetar toda a arquitetura da RNMC do controlador neural direto. Isto é, pretende-se encontrar, em tempo real, o número ótimo de neurônios nas camadas de entrada e interna da RNMC, o valor ótimo dos fator de treinamento da RNMC e do fator de adaptação do

controlador neural. Pretende-se fazer estudos sobre a estabilidade e as limitações dos algoritmos propostos neste trabalho, e estender os resultados obtidos com o conceito de estado passivo para controladores neurais adaptativos indireto.



## REFERÊNCIAS BIBLIOGRÁFICAS

---

[Äström & Hägglund, 1989] Äström, K.J. & Hägglund, T. **Automatic Tuning of PID Controllers**, Instrument Society of America, USA, 1988.

[Äström & Wittenmark, 1989] Äström, K.J. & Wittenmark, B. **Adaptive Control**, Addison-Wesley Publishing Company, USA, 1989.

[Äström, 1991] Äström, K.J. **Where is the Intelligence in Intelligent Control**, IEEE Control Systems Magazine, January 1991, pp.37-39.

[Azimi et alii, 1993] Azimi-Sadjadi & Sheedvash, S. & Trujillo, F.O. **Recursive Dynamic Node Creation in Multilayer Neural Networks**, IEEE Transactions On Neural Networks, Vol.4 No.2, march 1993, pp.242-256.

[Baglio et alii, 1991] Baglio, S. & Fortuna, L. & Graziani, S. & Muscato, G., **The Cell-to-Cell Mapping Design Approach Improves the Behaviors of Fuzzy Controller**, IECON'91, Proceedings, october 1991, Kobe - Japan, pp.1585-1589.

[Batur & Kasparian, 1991] Batur, C. & Kasparian, V. **Adaptive Expert Control**, International J. Control, Vol.54, No.4, april 1991, pp.867-881.

[Bavarian, 1988] Bavarian, B. **Introduction to Neural Networks for Intelligent Control**, IEEE Control Systems Magazine, april 1988, pp.3-7.

[Cavalcanti et alii, 1988] Cavalcanti, J.H.F. & Deep, G.S. & Lira, J.G.A., **Real Time Executive for Digital Signal Processor Employed in Process Control**, Proceeding IECON'88, october 1988, Singapura, pp.397-402.

- [1991a] \_\_\_\_\_. **Simulação do Controle Neural da Velocidade de um Motor CC**, publicação interna, Departamento de Engenharia Elétrica, CCT-UFPB, julho de 1991, Campina Grande, Brasil.
- [1991b] \_\_\_\_\_. **Controle Integrado Neural Adaptativo Direto com Aprendizagem em Tempo Real e Controle Deadbeat**, Anais do SBIA'91, Brasília DF, 18-21 novembro de 1991, Brasil, pp.162-172.
- [1992a] \_\_\_\_\_. **Comparação entre Controladores Deadbeat e Neural no controle de inversores PWM para Sistemas UPS**, Anais do INDUSCON'92, Aplicação Industrial de Eletricidade - Contribuições Técnicas em Sistema de Controle, São Paulo SP, 1-3 junho de 1992, Brasil, pp. 12-15.
- [1992b] \_\_\_\_\_. **A Neural Adaptive Controller For a D.C. Motor**, Third International Conference on Automation Robotics and Computer Vision, ICARCV'92, Singapura, 1992, pp. Co 2.8.1 - Co 2.8.5.
- [1992c] \_\_\_\_\_. **Executivo em Tempo Real para Controle de Motores Elétricos**, Anais do CBA'92, 14-18 setembro de 1992, UFES Vitória ES, Brasil, pp.1137-1139.
- [1992d] \_\_\_\_\_. **Controle Neural adaptativo Indireto da Velocidade de um Motor CC**, publicação interna, Departamento de Engenharia Elétrica, CCT-UFPB, outubro de 1992, Campina Grande, Brasil.
- [1993a] \_\_\_\_\_. **Controlador Fuzzy de Plantas Não Lineares**, Anais do I Simpósio Brasileiro de Automação Inteligente, UNESP, Rio Claro, São Paulo, Brasil, setembro de 1993, pp.534-542.
- [1993b] \_\_\_\_\_. **Design and Training of Multi-Layer Neural Networks for Adaptive Control**, Anais do 6'ISAI, International Symposium on Artificial Intelligence, Monterrey, Mexico, setembro de 1993.
- [1994a] \_\_\_\_\_. **Controlador Fuzzy/Neural de Plantas Não Lineares**, Anais do CBA'94, Espírito Santo, Setembro de 1994.
- [1994b] \_\_\_\_\_. **Fator de Adaptação para um Controlador Neural Direto**, publicação interna, Departamento de Engenharia Elétrica, CCT-UFPB, julho de 1994, Campina Grande, Brasil.
- [1994c] \_\_\_\_\_. **Posicionamento de um Pêndulo Invertido usando Redes Neurais**, Anais da XX Conferencia Latinoamericana de Informatica - 23' JAIIO, Buenos Aires, Argentina'94, setembro de 1994.
- [Chen, 1990] Chen, F.C. **Back-Propagation Neural Networks for Nonlinear Self-Tuning Adaptive Control**, IEEE Control System Magazine, april 1990, pp.44-48.
- [Chen & Chen, 1993] Chen, T. & Chen, H. **Approximation of Continuous Functionals by Neural Networks with Application to Dynamic Systems**, IEEE Transactions On Neural Networks, Vol.4, No.6, november 1993, pp. 910-918.

- [Chow et alii, 1992] Chow, M. & Menozzi, A. & Holcomb, F. **On the Comparison of the Performance of Emerging and Conventional Control Techniques for DC Motor Velocity and Position Control**, IECON'92, Proceedings, 1992, pp.1008-1013.
- [Cotter, 1990] Cotter, N.E., **The Stone-Weirstrass Theorem and Its Application to Neural Networks**, IEEE Transactions On Neural Networks, Vol.1 No.4, december 1990, pp.290-295.
- [D'azzo, 1988] D'azzo, J.J & Houpis, C.H., **Análise e Projeto de Sistemas de Controle Lineares**, Editora Guanabara, 1988.
- [Dorneles, 1993] Dorneles, C. R. **Aplicação de Redes Neurais em Controle Adaptativo**, Tese de Mestrado, IME, Rio de Janeiro, março de 1993.
- [Dote, 1988] Dote, Y., **Application of Modern Control Techniques to Motor Control**, Proceedings IEEE, Vol.76, N.4, april 1988, pp.438-454.
- [1990] \_\_\_\_\_, **Fuzzy and Neural Network Controller**, Proceeding of the IECON'90, Pacific Grove, California - USA, 1990, pp.1314-1343.
- [Dessaint et alii, 1990] Dessaint, L. A. & Hebert, B.J. & Huy, H.L. & Cavuoti, G. **A DSP-Based Adaptive Controller for a Smooth Positioning System**, IEEE Transactions on Industrial Electronics, Vol.37, N.5, october 1990, pp.372-377.
- [Fogel, 1991] Fogel, D.B. **An Information Criterion for Optimal Neural Network Selection**, IEEE Transaction On Neural Networks, Vol.2, N.5, september 1991, pp.490-497.
- [Fukuda & Shibata, 1992] Fukuda, T. & Shibata, T. **Theory and Applications of Neural Networks for Industrial Control Systems**, IEEE Transactions on Industrial Electronics, Vol.39, N.6, december 1992.
- [Furuhashi et alii, 1992] Furuhashi, T. & Horikawa, S.I. & Uchikawa, Y. **On stability of fuzzy control using a fuzzy modeling method**, IECON'92, Proceedings, 1992, pp.982-985.
- [Furuta & Yamakita, 1991] Furuta, K. & Yamakita, M. **Swing Up Control of Inverted Pendulum**, IECON'91, Proceedings, October 1991, Kobe - Japan, pp.2193-2198.
- [Gokhale et alii, 1987] Gokhale, K.P. & Kawamura, A. & Hoft, R.G. **Dead Beat Microprocessor Control of PWM Inverter for Sinusoidal Output Waveform Synthesis**, IEEE Trans. On Industry Applications, Vol.IA23, No.5, september/october 1987.
- [Hang & Sin, 1991] Hang, C.C. & Sin, K.K., **On-Line Auto Tuning of PID Controllers Based On Cross-Correlation Technique**, IEEE Trans. On Industrial Electronics, Vol.38, No.6, december, 1991, pp.428-437.
- [Handelman, 1990] Handelman, D.A. **Integrating Neural Networks and Knowledge-Based Systems for Intelligent Robotic Control**, IEEE Control Systems Magazine, april 1990, pp.77-87.

- [Hoskin et alii, 1992] Hoskin, D.A. & Hwang, J.N. & Vagners, J. **Iterative Inversion of Neural Network and its Application to Adaptive Control**, IEEE Transaction On Neural Networks, Vol.3, No.2, march 1992, pp.292-301.
- [Huang & Huang, 1991] Huang, S.C. & Huang, Y.F., **Bounds on the Number of Hidden Neurons in Multilayer Perceptrons**, IEEE Transaction On Neural Networks, Vol.2, No.1, jan. 1991, pp.47-55.
- [Hägglund & Åström, 1991] Hägglund, T. & Åström, K.J. **Industrial Adaptive Controllers Based on Frequency Response Techniques**, International Federation of Automatica Control, Automatica, Vol.27, No.3, 1991, pp.599-609.
- [Hunt et alii, 1992] Hunt, K.J. & Sbarbaro, D. & Zbikowski, R. & Gawthrop, P.J. **Neural Networks for Control Systems - A Survey** International Federation of Automatica Control, Automatica, Vol.28, No.6, 1992, pp.1083-1112.
- [Hornik, 1991] Hornik, K. **Approximation capabilities of multilayer feedforward networks**, Neural Networks, Vol.4, 1991, pp.251-257.
- [Ishida et alii, 1991] Ishida, T. & Shiokawa, N. & Nagado, T. & Shinichi, G. **Learning Control of a Inverted Pendulum Using a Neural Network**, IECON'91, Proceedings, october 1991, Kobe - Japan, pp.1401-1404.
- [Jacobina et alii, 1992] Jacobina, C.B. & Primo, A.R. & Lima, A.M.N., **Estimação dos Parâmetros da Máquina de Corrente Contínua Utilizando um Modelo Dinâmico Discreto**, Anais do CBA'92, 14-18 setembro de 1992, UFES Vitória ES, Brasil.
- [Kuo, 1985] Kuo, B.C., **Sistemas de Controle Automático**, Prentice/Hall do Brasil, 1985.
- [Kuperstein & Wang, 1990] Kuperstein, M. & Wang, J. **Neural Controller for Adaptive Movements with Unforeseen Payloads**, IEEE Transactions On Neural Networks, Vol.1 No.1, march 1990, pp.137-142.
- [Kawaji et alii, 1991] Kawaji, S. & Maeda, T. & Matsunaga, N., **Fuzzy Control Using Knowledge Acquired from PD Control**, IECON'91, Proceedings, October 1991, Kobe - Japan, pp.1549-1554.
- [Koivo, 1991] Koivo, H.N. **Adaptive Control and Neural Networks**, IECON'91, Tutorial Text Book, October 1991, Kobe - Japan, pp.70-82.
- [Khalid & Omatu, 1992] Khalid, M. & Omatu, S., **A Neural Network Controller for a Temperature Control System**, IEEE Control System, june 1992, pp.58-64.
- [Kraft, 1990] Kraft, L.G. & Campagna, D.P., **A Comparison Between CMAC Neural Network Control and Two Traditional Adaptive Control Systems**, IEEE Control System Magazine, april 1990, pp.36-43.
- [Landau, 1979] Landau, Y.D., **Adaptive Control**, Marcel Dekker, Inc., New York, USA, 1979.

- [Lin & Sheu, 1992] Lin, C.E. & Sheu, Y.R. **A Hybrid-Control Approach for Pendulum-Car Control**, IEEE Transactions on Industrial Electronics, Vol.39, No.3, June 1992, pp.208-214.
- [Levin & Narendra, 1993] Levin, A.U. & Narendra, K.S. **Control of Nonlinear Dynamical Systems Using Neural Networks: Controllability and Stabilization**, IEEE Transactions On Neural Networks, Vol.4 No.2, march 1993, pp.192-206.
- [Lima et alii, 1994] Lima, A.M.N., & Cavalcanti, J.H.F & Deep, G.S., **On-line Training of Adaptative Neural Network controllers**, Anais do IEEE Industrial Electronics Society IECON'94, Proceedings, , Bologna, Italy, setembro 1994.
- [Lima et alii, 1993] Lima, A.M.N. & Silva, E.R.C. & Jacobina, C.B. **Sistema de Acionamento Estático de Motor de CC com Controle Digital**, Relatório Técnico N.2, DEE/CCT/UFPB, Campina Grande PB, Brasil, maio de 1993.
- [Ljung & Söderström, 1983] Ljung, L. & Söderström, T. **Theory and Practice of Recursive Identification**, The MIT Press Series, Massachusetts USA, 1983.
- [Ljung, 1987] Ljung, L. **System Identification, Theory for the User**, Prentice-Hall, Inc., New Jersey, USA, 1983.
- [McClelland & Rumelhart, 1988] McClelland, J.L. & Rumelhart, D.E. **Explorations in Parallel Distributed Processing**, The MIT Press, Cambridge, Massachusettes, EUA, 1988.
- [Mamdani & Gaines, 1981] Mamdani, E.H. & Gaines, B.R., **Fuzzy Reasoning and its Applications**, Academic Press Inc., London, England, 1981.
- [Narendra, 1991] Narendra, K.S. **Intelligent Control**, Control Systems Magazine, january 1991, pp.39-40.
- [Narendra & Mukhopadhyay, 1991] Narendra, K.S. & Mukhopadhyay, S., **Intelligent Control Using Neural Networks**, IEEE Control Systems Magazine, april 1992, pp.11-18
- [Narendra & Parthasarathy, 1990] Narendra, K. S. & Parthasarathy, K., **Identification and Control of Dynamical Systems Using Neural Networks**, IEEE Transactions On Neural Networks, Vol.1 No.1, march 1990, pp.4-27.
- [1991] \_\_\_\_, **Gradient Methods for the Optimization of Dynamical Systems Containing Neural Networks**, IEEE Transactions On Neural Networks, Vol.2 No.2, march 1991, pp.252-262.
- [Ogata, 1982] Ogata, Katsuhiko **Engenharia de Controle Moderno**, Editora Prentice/Hall do Brasil Ltda, Rio de Janeiro, 1982.
- [Parks, 1981] Parks, P.C. **Stability and Convergence of Adaptive Controllers-Continuous Systems**, IEE Proc., Vol.128, Pt. D. No.5, september 1981, pp.195-200.
- [Porter, 1989] Porter, B. **Issues in the Design of Intelligent Control Systems**, IEEE Control Systems Magazine, january 1989, pp.97-99.

- [Polycarpou & Ioannou, 1992] Polycarpou M.M. & Ioannou, P.A., **Learning and Convergence Analysis of Neural-Type Structured Networks**, IEEE Trans. on Neural Networks, Vol.3, No.1, January 1992, pp.39-50.
- [Rumelhart et alii, 1986] Rumelhart, D.E. & Hinton, G.H. & Williams, R.J., **Learning Internal Representations by Error Propagation**, Parallel Distributed Processing Explorations in the Microstructures of Cognition, Vol.1, MIT Press, 1986, pp.318-362.
- [Raiskila & Werbos, 1991] Raiskila, P.E. & Werbos, P.J. **Properties of a Neural Network Controller**, IECON'91, Tutorial Text Book, October 1991, Kobe - Japan, pp.84-88.
- [Rangwala & Dornfeld, 1989] Rangwala, S.S. & Dornfeld, D.A., **Learning and Optimization of Machining Operations Using Computing Abilities of Neural Networks**, IEEE Trans. on Systems, Man, and Cybernetics, Vol.19, N.2, March/April 1989, pp.299-314.
- [Rosenfeld, 1988] Rosenfeld, E., **Neurocomputing Foundations of Researchs**, ed. J. Anderson e E. Rosenfeld, MIT Press, Cambridge, Massachusetts, USA, 1988.
- [Saud, 89] Saud, E. **Dimensionality-Reduction Using Connectionist Networks**, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol.11, No.3, March 1989, pp.304-314.
- [Strefezza & Dote, 1991] Strefezza, M. & Dote, Y., **Fuzzy and Neural Networks Controller**, IECON'91, Proceedings, October 1991, Kobe - Japan, pp.1437-1442.
- [Saito et alii, 1985] Saito, K. & Ohmae, T. & Matsuda, T. & Kamyama, K. & Matsuda, Y. **A Microprocessor-Controlled Speed Regulator with Instantaneous Speed Estimation for Motor Drives**, Proceeding IECON'85, pp.755-760.
- [Shoureshi, 1991a] Shoureshi, R. **The Mystique of Intelligent Control**, IEEE Control Systems Magazine, January 1991, pp.33-33.
- [1991b] \_\_\_\_\_, **Learning and Decision Making for Intelligent Control**, IEEE Control Systems Magazine, January 1991, pp.34-37.
- [Sontag, 1992] Sontag, E. D. **Feedback Stabilization Using Two-Hidden-Layer Nets**, IEEE Transactions on Neural Networks, Vol.3, N.6, November 1992.
- [Tanomaru & Omatu, 1991] Tanomaru, J. & Omatu, S. **Towards Effective Neuromorphic Controllers**, IECON'91, Proceedings, October 1991, Kobe - Japan, pp.1395-1400.
- [1992] \_\_\_\_\_, **Process Control by On-Line Trained Neural Controllers**, IEEE Trans. on Industrial Electronics, Vol.39, No.6, December 1992, pp.511-521.
- [Villiers & Barnard, 1993] Villiers, J. & Barnard, E. **Backpropagation Neural Nets with One and Two Hidden Layers**, IEEE Transactions on Neural Networks, Vol.4, No.1, January 1993, pp.136-141

[Watanabe & Shimizu, 1991] Watanabe, E. & Shimizu, H. **A Study on Generalization Ability of Neural Network for Manipulator Inverse Kinematics**, IECON'91, Proceedings, october 1991, Kobe - Japan, pp.957-962.

[Werbos, 1991] Werbos, P.J. **Artificial Neural Networks: General Capabilities and Control Applications**, IECON'91, Tutorial Text Book, October 1991, Kobe - Japan, pp.5-66.

[Widrow, 1988] Widrow, B. **The Original Adaptive Neural Network Broom Balance**, International Symposium on Circuits and Systems, 1988, pp.351-357.

[Yamada & Yabuta, 1990] Yamada, T. & Yabuta, T., **Nonlinear Neural Network Controller for Dynamic System**, Proceeding of the IECON'90, Pacific Grove, California - USA, 1990, pp.1244-1249.

[Yamashita et alii, 1991] Yamashita, T. & Katoh, R. & Singh, S. & Hori, T., **Stability Analysis of Fuzzy Control System Applying Conventional Methods**, IECON'91, Proceedings, october 1991, Kobe - Japan, pp.1579-1584.

[Zadeh, 1965] Zadeh, L.A. **Fuzzy Sets**, Information and Control, 1965, 8, pp.28-44.

[1973] \_\_\_\_\_, **Outline of a New Approach to the Analysis of Complex Systems and Decision Process**, IEEE Trans. 1973, SMC-3, pp.338-353.



## REDES NEURAIS MULTI-CAMADAS

## A.1 Introdução

Na Figura A.1 é mostrada uma rede neural multi-camadas (RNMC) [McClelland & Rumelhart, 1988][Rosenfeld, 1988][Rangwala & Dornfeld, 1989] usada para ser treinada com dinâmica inversa de uma planta. A RNMC da Figura A.1 usa neurônios do tipo linear (L) na camada de entrada, tipo sigmóide  $S(\eta, T_h, \beta_h, \Theta_h, Win)$  na camada interna ou escondida (representada pelo subscrito h), e tipo tangente hiperbólico  $T(\eta, T_o, \beta_o, \Theta_o, Wout)$  na última camada da RNMC (representada pelo subscrito o).

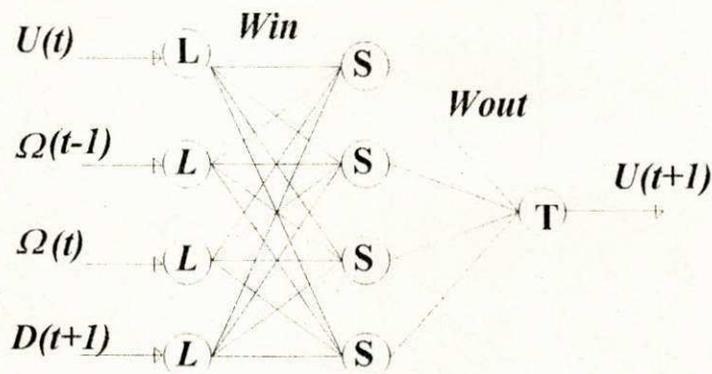


Figura A.1 Rede neural multi-camadas.

O índice de desempenho da RNMC é mostrado na equação A.1.

$$E = \frac{1}{2}e(t+1)^2 + \frac{1}{2}[U^*(t+1) - U(t+1)]^2 \quad (A.1)$$

Aos neurônios do tipo sigmóide da RNMC podem ser associados os seguintes parâmetros

$T_{ik}$  - Limiar do  $i$ -ésimo nó na  $k$ -ésima camada.

$W_{ijk}$  - Peso entre o  $j$ -ésimo nó na  $(k-1)$ -ésima camada e o nó na  $k$ -ésima camada.

$NET_{ik}$  - Entrada do  $i$ -ésimo nó na  $k$ -ésima camada.

$O_{ik}$  - Saída do  $i$ -ésimo nó na  $k$ -ésima camada.

Associa-se a função  $f(.)$  da equação A.2 aos parâmetros dos neurônios.

$$f(NE_{T_{ik}}) = \frac{\Theta_{ik}}{1 + e^{-\beta_{ik} * NE_{T_{ik}}}} \quad (A.2)$$

A equação A.3 representa a entrada e a equação A.4 representa a saída para um nó da RNMC.

$$NET_{ik} = \sum_j |W_{ijk} O_{jk-1}| + T_{ik} \quad (A.3)$$

$$O_{ik} = f(NE_{T_{ik}}) \quad (A.4)$$

O esquema de aprendizagem usa a regra delta generalizada desenvolvida por Rumelhart [1986]. O índice de desempenho é utilizado para treinamento da RNMC é dado pela equação A.1 e os parâmetros para treinamento da RNMC são os mostrados nas equações A.2, A.3 e A.4. O procedimento de treinamento da RNMC espera tornar o erro total, mostrado da equação A.1, igual ou próximo de zero pelo ajustamento necessário nos parâmetros da RNMC. Isto se constitui essencialmente num problema de minimização que pode ser resolvido pela regra delta generalizada. O cálculo do gradiente do erro em relação aos parâmetros de aprendizagem é feita pela propagação retroativa do erro pela rede.

O procedimento de aprendizagem dos parâmetros da RNMC é equivalente a modificação direta nas parâmetros da rede (equações A.5).

$$\Delta(W_{ijk}) = -\eta \frac{\partial E}{\partial W_{ijk}} \quad (A.5a)$$

$$\Delta(T_{ik}) = -\eta \frac{\partial E}{\partial T_{ik}} \quad (A.5b)$$

$$\Delta(\theta_{ik}) = -\eta \frac{\partial E}{\partial \theta_{ik}} \quad (A.5c)$$

$$\Delta(\beta_{ik}) = -\eta \frac{\partial \mathcal{E}}{\partial \beta_{ik}} \quad (\text{A.5d})$$

$$\Delta(O_{jk-1}) = -\eta \frac{\partial \mathcal{E}}{\partial O_{jk-1}} \quad (\text{A.5e})$$

O cálculo das derivadas parciais das equações A.5 é feito com o auxílio das equações A.3 e A.4, resultando as equações A.6.

$$\frac{\partial NET_{ik}}{\partial W_{ijk}} = O_{jk-1} \quad (\text{A.6a})$$

$$\frac{\partial NET_{ik}}{\partial T_{ik}} = 1 \quad (\text{A.6b})$$

$$\frac{\partial O_{ik}}{\partial \theta_{ik}} = O_{ik} \theta_{ik} \quad (\text{A.6c})$$

$$\frac{\partial O_{ik}}{\partial \beta_{ik}} = O_{ik}(1-O_{ik})\theta_{ik}NET_{ik} \quad (\text{A.6d})$$

$$\frac{\partial O_{ik}}{\partial NET_{ik}} = O_{ik}(1-O_{ik})\theta_{ik}\beta_{ik} \quad (\text{A.6e})$$

Para simplificar a representação das derivadas parciais dos parâmetros das equações A.6, definem-se  $\rho$  e  $\phi$  nas equações A.7.

$$\frac{\partial \mathcal{E}}{\partial NET_{ik}} = -\rho_{ik} \quad (\text{A.7a})$$

$$\frac{\partial \mathcal{E}}{\partial O_{ik}} = -\phi_{ik} \quad (\text{A.7b})$$

As derivadas parciais do índice de desempenho em relação aos parâmetros de aprendizagem podem ser definidas pelas equações A.8.

$$\frac{\partial \mathcal{E}}{\partial W_{iik}} = \frac{\partial \mathcal{E}}{\partial NET_{ik}} \frac{\partial NET_{ik}}{\partial W_{iik}} = -\rho_{ik}O_{jk-1} \quad (\text{A.8a})$$

$$\frac{\partial \mathcal{E}}{\partial T_{ik}} = \frac{\partial \mathcal{E}}{\partial NET_{ik}} \frac{\partial NET_{ik}}{\partial T_{ik}} = -\rho_{ik} \quad (\text{A.8b})$$

$$\frac{\partial \mathcal{E}}{\partial \theta_{ik}} = \frac{\partial \mathcal{E}}{\partial \omega_{ik}} \frac{\partial \omega_{ik}}{\partial \theta_{ik}} = -\phi_{iik}(O_{iik} \theta_{iik}) \quad (\text{A.8c})$$

$$\frac{\partial \mathcal{E}}{\partial \beta_{ik}} = \frac{\partial \mathcal{E}}{\partial \omega_{ik}} \frac{\partial \omega_{ik}}{\partial \beta_{ik}} = -\phi_{iik}[O_{iik}(1 - O_{iik} \theta_{iik})NET_{ik}] \quad (\text{A.8d})$$

$$\frac{\partial \mathcal{E}}{\partial \omega_{ik-1}} = \frac{\partial \mathcal{E}}{\partial NET_{ik}} \frac{\partial NET_{ik}}{\partial \omega_{ik}} = -\rho_{ik} W_{ijk} \quad (\text{A.8e})$$

Os valores de  $\rho$  e  $\phi$  para a camada de saída podem ser calculados pelas equações A.9.

$$-\rho_{in} = \frac{\partial \mathcal{E}}{\partial NET_{in}} = \frac{\partial \mathcal{E}}{\partial \omega_{in}} \frac{\partial \omega_{in}}{\partial NET_{in}} \quad (\text{A.9a})$$

$$\rho_{in} = (d_i - O_{in}) O_{in} (1 - O_{in} \theta_{in}) \beta_{in} \quad (\text{A.9b})$$

$$-\phi_{in} = \frac{\partial \mathcal{E}}{\partial \omega_{in}} \quad (\text{A.9c})$$

$$\phi_{in} = (d_i - O_{in}) \quad (\text{A.9d})$$

Os valores de  $\rho$  e  $\phi$  para as camadas anteriores podem ser calculados pelas equações A.10.

$$-\phi_{ik} = \frac{\partial \mathcal{E}}{\partial \omega_{ik}} = \sum_j \left[ \frac{\partial \mathcal{E}}{\partial NET_{jk+1}} \frac{\partial NET_{jk+1}}{\partial \omega_{ik}} \right] \quad (\text{A.10a})$$

$$\phi_{ik} = \sum_j [\rho_{jk+1} W_{jik+1}] \quad (\text{A.10b})$$

$$\rho_{ik} = \phi_{ik} O_{ik} (1 - O_{ik} \theta_{ik}) \beta_{ik} \quad (\text{A.10c})$$

A emulação da rede neural multi-camadas em computadores digitais foi feita com duas funções básicas mostradas no Algoritmo A.1 que é escrito na linguagem de programação C.

No Algoritmo 1 descrevem-se duas funções. Uma função para calcular o valor atual das saídas dos neurônios da RNMC, denominada de `entrasai()`, e uma outra função para implementação do algoritmo de propagação retroativa para treinamento da RNMC, denominada de `backp()`. Neste Algoritmo, usou-se `nin=4` (número de entradas da RNMC), `nhid=4` (número de neurônios da camada oculta da RNMC), e `nout=1` (número de neurônios da camada de saída da RNMC). No Algoritmo 1 não serão mostrados os termos do "momentum" para treinamento das RNMCs.

## Algoritmo A.1

Funções usadas para calcular a saída e treinar a RNMC

```

/*      *      *      *      *      *      *      *      *      *      *      *      */
/*Descrição das variáveis globais                                                    */
/*nin   número de neurônios na primeira camada                                     */
/*nhid  número de neurônios na camada oculta                                       */
/*nout  número de neurônios na última camada                                       */
/*in[4]  entradas normalizadas da RNMC                                             */
/*win[4][4] pesos de entrada da RNMC                                             */
/*tihid[4] T da camada oculta                                                       */
/*tetahid[4]  $\theta$  da camada oculta                                           */
/*betahid[4]  $\beta$  da camada oculta                                             */
/*hid[4] valor de saída dos neurônios da camada oculta                             */
/*tiout[1] T da última camada                                                       */
/*tetaout[1]  $\theta$  da última camada                                           */
/*betaout[1]  $\beta$  da última camada                                           */
      */
/*out[1] valor sigmoide do neurônio da última camada                               */
/*iin[1] valor sigmoide de saída usado no treinamento da RNMC                     */
/*valsai[1] valor tangente hiperbólico do neurônio da última camada, saída da RNMC */
/*dwin valor de ajuste da entrada  $U(t)$                                        */
/*      *      *      *      *      *      *      *      *      *      *      *      */
entrasai()
{
float somaa;
/*Gera entradas e saídas da camada oculta*/
for(j=1;j<=nhid;j++)
{
somaa = tihid[j];
for(i=1;i<=nin;i++)somaa = somaa + in[i]*win[i][j];
hid[j] = tetahid[j]/(1. + exp(-betahid[j]*somaa));
}
/*Gera entradas e saídas da última camada*/
for(j=1;j<=nout;j++)
{
somaa = tiout[j];

```

```

for(i=1;i<=nhid;i++)somaa = somaa + hid[i]*wout[i][j];
out[j] = tetaout[j]/(1. + exp(-betaout[j]*somaa));/*Sigmoide*/
valsai[j] = 2.*out[j]-1.; /*Tangente hiperbolica*/
}
}
backp()
{
float soma[10],d[10]dl[10],dll[10];
/*Ultima camada*/
for(i=1;i<=nout;i++)d[i]=out[i]*(1.-out[i]/tetaout[i])*(iin[i]-out[i]);
for(j=1;j<=nout;j++)
{
soma[j] = 1.;
for(i=1;i<=nhid;i++)soma[j] = soma[j] + hid[i]*wout[i][j];
}
for(i=1;i<=nout;i++)
for(j=1;j<=nhid;j++)
{
wout[j][i]=wout[j][i]+fatcon*d[i]*hid[j]*betaout[i];
betaout[i] = betaout[i] + fatcon*in[i]*d[j]*soma[i];
}
}
for(i=1;i<=nout;i++)
{
tiout[i] = tiout[i] + fatcon*d[i];
tetaout[i] = tetaout[i] + fatcon*(iin[i]-out[i])*out[i]/tetaout[i];
}
}
/*Primeira camada*/
for(i=1;i<=nhid;i++)
{
dll[i] = 0;
for(j=1;j<=nout;j++)dll[i]=dll[i]+d[j]*wout[i][j];
}
for(j=1;j<=nhid;j++)dl[j]=hid[j]*(1.-hid[j]/tetahid[j])*dll[j]*betahid[j];
for(j=1;j<=nhid;j++)
{
soma[j] = 1.;
for(i=1;i<=nin;i++)soma[j] = soma[j] + in[i]*win[i][j];
}
}

```

```

for(i=1;j<=nin;i++)
for(j=1;j<=nhid;j++)
{
win[i][j]=win[i][j]+fatcon*in[i]*dl[j];
betahid[j] = betahid[j] + fatcon*dl[j]*soma[j]/betahid[j];
}
for(j=1;j<=nhid;j++)
{
tihid[j]=tihid[j] + fatcon*dl[j];
tetahid[j]=tetahid[j] + fatcon*dll[j]*hid[j]/tetahid[j];
}
}

```

O algoritmo A.2 é usado na simulação do motor CC. Neste algoritmo, mostram-se a função implementada para o cálculo do algoritmo de integração de quarta ordem de Runge-Kutta, a função para o cálculo da corrente de armadura, e a função para o cálculo da velocidade angular do motor CC.

### Algoritmo A.2

Simulação do sistema motor CC

```

float funcaoia(float yy) /*Calcula a corrente de armadura*/
{
return(ui/La - yy * Ra/La - wm * Kb/La);
}
float funcaoΩ(float yy) /*Calcula a velocidade do motor CC*/
{
return(KKi*ia/Jm - yy*Bm/Jm - Tl/Jm );
}
rungekuta4() /*Ordem 4*/
{
/* ia corrente de armadura*/
/* Ω velocidade angular */
/* Ts tempo de amostragem */
k1 = Ts*funcaoia(ia);
k2 = Ts*funcaoia(ia+k1/2.);
k3 = Ts*funcaoia(ia+k2/2.);
k4 = Ts*funcaoia(ia+k3);
ia = ia + (k1 + 2. * k2 + 2. * k3 + k4)/6.;
k1 = Ts*funcaoΩ(wm);

```

$$k_2 = T_s * \text{funcao}\Omega(\omega_m + k_1/2.);$$

$$k_3 = T_s * \text{funcao}\Omega(\omega_m + k_2/2.);$$

$$k_4 = T_s * \text{funcao}\Omega(\omega_m + k_3);$$

$$\Omega = \Omega + (k_1 + 2. * k_2 + 2. * k_3 + k_4)/6.;$$

}

## CONTROLADORES FUZZY

### B.1 Introdução

A teoria de controle moderno tem tido sucesso nas áreas em que os sistemas determinísticos ou estocásticos são bem definidos. Controladores com operadores humanos são geralmente adequados em algumas aplicações. Isto se deve ao fato de que os operadores humanos são capazes de construir nas suas mentes um modelo do processo com a necessária precisão para a execução à contento da tarefa de controle, podendo inclusive aprender com a experiência obtida no controle da planta. Para suportar a transformação das declarações sobre a estratégia de controle geralmente vagas, não numéricas dos controladores humanos, é necessário uma forma de cálculo semi-quantitativo. Baseado nessas considerações é que Zadeh [1965][1973] introduziu e desenvolveu a teoria dos conjuntos fuzzy e do raciocínio aproximado. Conjuntos fuzzy são ferramentas que podem ser usadas na manipulação de conceitos vagos e particularmente, controladores fuzzy representam os esforços na direção da emulação da capacidade humana.

Diversos controladores fuzzy [Mamdani & Gaines, 1981] já foram implementados e exaustivamente testados. Alguns pesquisadores estão estudando formas híbridas de controladores fuzzy com redes neurais artificiais, modo deslizante, etc [Dote, 1990]. Além disso, estudos estão sendo iniciados sobre a estabilidade de controladores fuzzy. Por outro lado, alguns pesquisadores

estão estudando estratégias de projeto automático de controladores fuzzy. Usando estratégias de projeto heurística, Strefezza & Dote [1991] implementaram, para o controle de posição de um motor de corrente contínua, um controlador fuzzy híbrido usando redes neurais multi-camadas com algoritmo de aprendizagem com propagação retroativa. Seguindo esse mesmo padrão de projeto, Dote usou duas redes neurais. A primeira rede neural foi usada para geração da função de pertinência, e a segunda para geração das regras fuzzy do motor.

Recentemente diversos pesquisadores implementaram controladores fuzzy baseado em métodos de controle tradicionais ou usando uma nova abordagem. Por exemplo, Kawaji et alii [1991] propuseram um método para o projeto de sistemas de controle fuzzy baseado no esquema convencional de controladores PD. Baglio et alii [1991] desenvolveram uma estratégia de projeto automático de controladores fuzzy baseado na abordagem conhecida por "cell-to-cell", que consiste em mapear o espaço discreto da planta em células, e desenvolver um controlador otimizado para cada célula.

Atualmente, alguns pesquisadores têm estudado a estabilidade de controladores fuzzy. Por exemplo, Yamashita et alii [1991] estudaram a estabilidade de sistemas de controle fuzzy usando métodos de análise de estabilidade convencional. Furuhashi et alii [1992] apresentaram a análise da estabilidade de sistemas de controle fuzzy usando um método de modelamento baseado em conjuntos fuzzy.

## B.2 Controlador fuzzy tradicional

O controlador fuzzy é uma ferramenta capaz de processar uma forma fuzzy de informação. Ele é construído a partir de um conjunto de regras ou de implicações **if - then** fuzzy do tipo {if premissa<sub>1</sub> **and** premissa<sub>2</sub> ... **and** premissa<sub>n</sub> **then** consequência}. Define-se premissa<sub>i</sub> como uma condição no conjunto fuzzy definido para uma quantidade relativa à planta e consequência como a ação de controle expressa como um conjunto fuzzy e definida no intervalo de variação da entrada da planta. No projeto do controlador fuzzy é requerida a fixação das variáveis da premissa e a definição das funções de pertinência ao conjunto fuzzy usado no controlador.

Usualmente, uma planta requer um controle na forma de quantidade real e não na forma fuzzy. No controlador fuzzy usam-se dois dispositivos para transformar uma variável da forma real para a forma fuzzy e vice versa. Esses dispositivos são o fuzzyficador ("fuzzyfier") que precede o controlador fuzzy para associar a um valor linguístico a uma quantidade real, e o defuzzyficador ("defuzzyfier") que segue o controlador e associa um número real a um conjunto fuzzy.

Na figura B.1 é mostrada a arquitetura típica de um controlador baseado nas regras fuzzy [Mamdani & Gaines, 1991]. O controlador foi desenvolvido para o controle de posição de uma planta SISO (motor CC), com entrada  $U(t)$  e saída  $\theta(t)$  (posição), usando uma ação de controle que se assemelha aquela do algoritmo do controlador proporcional e derivativo (PD), como

estratégia de projeto. Para a planta da Figura B.1, define-se  $\theta_r(t)$  como o sinal de referência,  $E(t) = \theta_r(t) - \theta(t)$ , e  $CE(t) = E(t) - E(t-1)$ .

As entradas do controlador fuzzy são duas variáveis fuzzy. Uma delas definida como o erro entre o valor desejado e o atual da saída da planta ( $E_f$ ), e a outra como a diferença entre os dois últimos erros ( $CE_f$ ). Essas variáveis são obtidas pela transformação das variáveis discretas  $E(t)$  e  $CE(t)$  em variáveis fuzzy, que em seguida são escaladas para valores exigidos pelo bloco de regras de controle. A saída  $U_f$  do controlador fuzzy é novamente escalado para a forma determinística  $U(t)$  a ser usado no controle da planta.

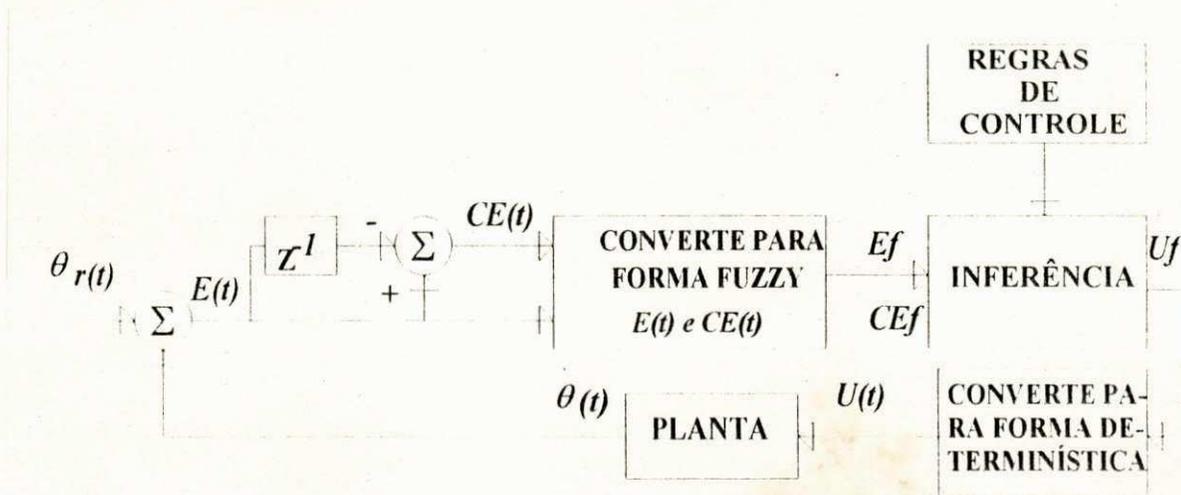


Figura B.1 - Controlador fuzzy.

O conjunto das regras usadas por um controlador lógico fuzzy geralmente é armazenado como uma tabela usando variáveis lógicas fuzzy do tipo *PG* (positivo grande), *PM* (positivo médio), *PP* (positivo pequeno), *ZE* (zero), *NG* (negativo grande), *NM* (negativo médio), *NP* (negativo pequeno).

O projeto de controladores fuzzy, baseado no esquema convencional de controladores PD, pode ser formulado em três etapas:

1) Definição do conjunto fuzzy para representar o domínio discreto do erro ( $E(t)$ ), variação dos erros ( $CE(t)$ ), e entrada da planta ( $U(t)$ ). Na figura B.2 são mostradas as curvas típicas, numa forma triangular, da função de pertinência  $E_f$  em toda a faixa de variação de  $E(t)$  ( $-1 \leq E(t) \leq 1$ ).

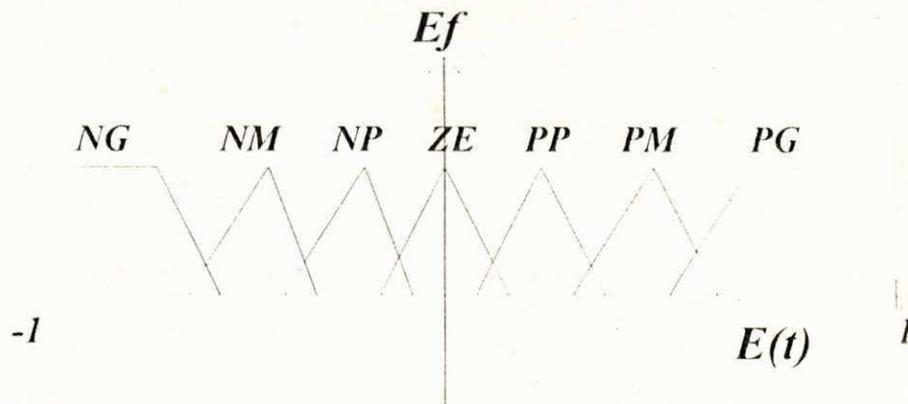


Figura B.2 - Função de pertinência de  $E_f$ .

2) Conversão do conjunto fuzzy escolhido e do algoritmo de controle linguístico numa tabela com as saídas discretas do controlador. A tabela B.1 representa uma tabela típica encontrada em controladores fuzzy. Ela foi desenvolvida a partir das regras heurísticas (mostradas nas regras B.1) observadas no controlador padrão PD. Na Tabela B.2 são mostrados os valores discretos atribuídos às variáveis lógicas fuzzy de  $U_f$ .

TABELA B.1

$U_f$	$CE_f$						
$E_f$	NG	NM	NP	ZE	PP	PM	PG
$E_f$	NG						
$E_f$	NG	NG	NG	NM	NM	NM	NM
$E_f$	NG	NM	NM	NP	NP	NP	NP
$E_f$	NP	NP	ZE	ZE	ZE	PP	PM
$E_f$	PP	PP	PP	PP	PM	PM	PG
$E_f$	PM	PM	PM	PM	PG	PG	PG
$E_f$	PG						

REGRAS B.1

- 1) if  $E_f$  is PG and  $CE_f$  is algum, then  $U_f$  is PG;
- 2) if  $E_f$  is PM and  $CE_f$  is negativo ou ZE, then  $U_f$  is PM;
- 3) if  $E_f$  is PP and  $CE_f$  is ZE, then  $U_f$  is PP;
- 4) if  $E_f$  is ZE and  $CE_f$  is ZE, then  $U_f$  is ZE;
- 5) if  $E_f$  is NP and  $CE_f$  is ZE, then  $U_f$  is NP;
- 6) if  $E_f$  is NM and  $CE_f$  is positivo ou ZE, then  $U_f$  is NM;
- 7) if  $E_f$  is NG and  $CE_f$  is algum, then  $U_f$  is NG;

TABELA B.2

$U_f$	$NG$	$NM$	$NP$	$ZE$	$PP$	$PM$	$PG$
$U(t)$	$-1.$	$-0.7$	$-0.3$	$0.$	$0.3$	$0.7$	$1.$

Na Figura B.3 apresenta-se o diagrama tri-dimensional da superfície de controle gerada pelo controlador fuzzy. Observe-se a não linearidade em  $U(t)$  devido às regras fuzzy.

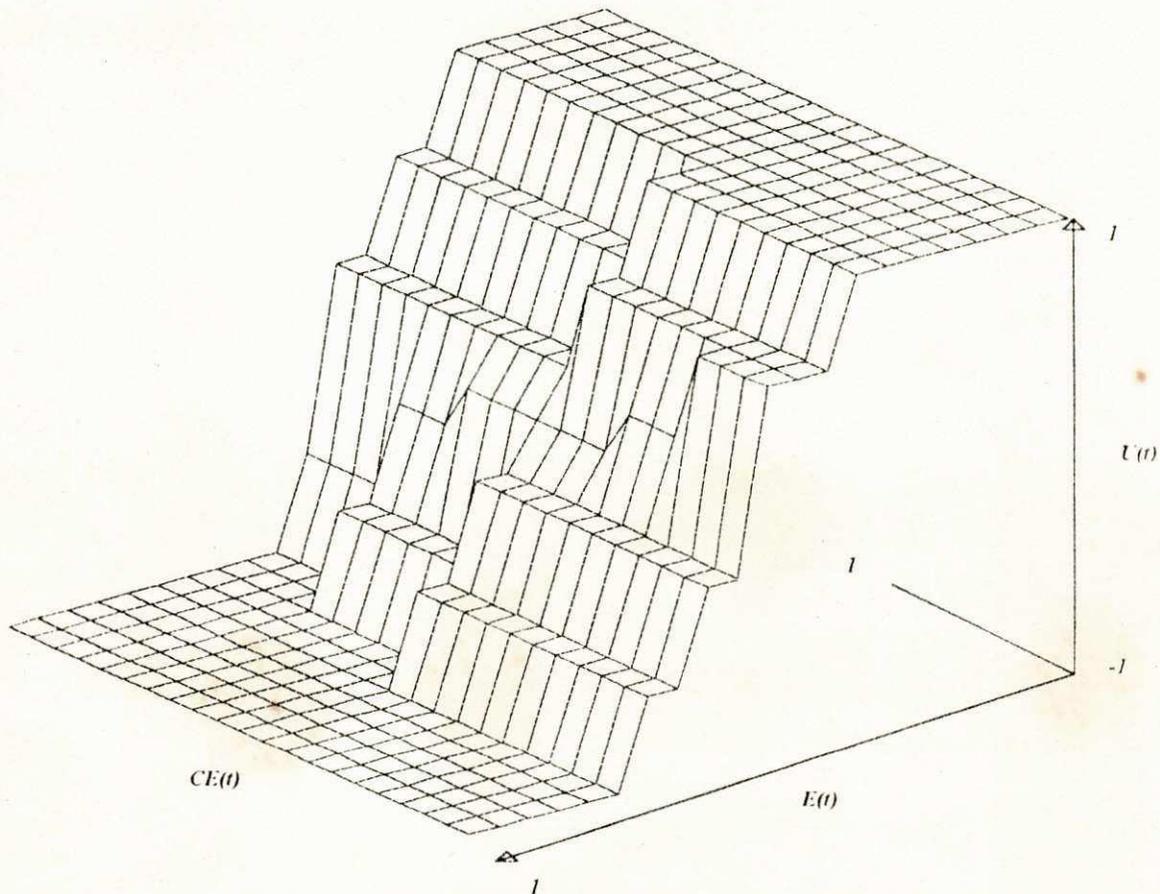


Figura B.3 - Superfície de controle gerada pelo controlador fuzzy

3) Os valores de saída discretos de controle são selecionados de acordo com um fator de escala, ou peso. Esse peso pode ser constante ou possuir valores múltiplos dentro da faixa de variação dos parâmetros dependentes.

Na Figura B.4 são mostrados os resultados experimentais obtidos com o controlador fuzzy de posição do eixo do motor CC.

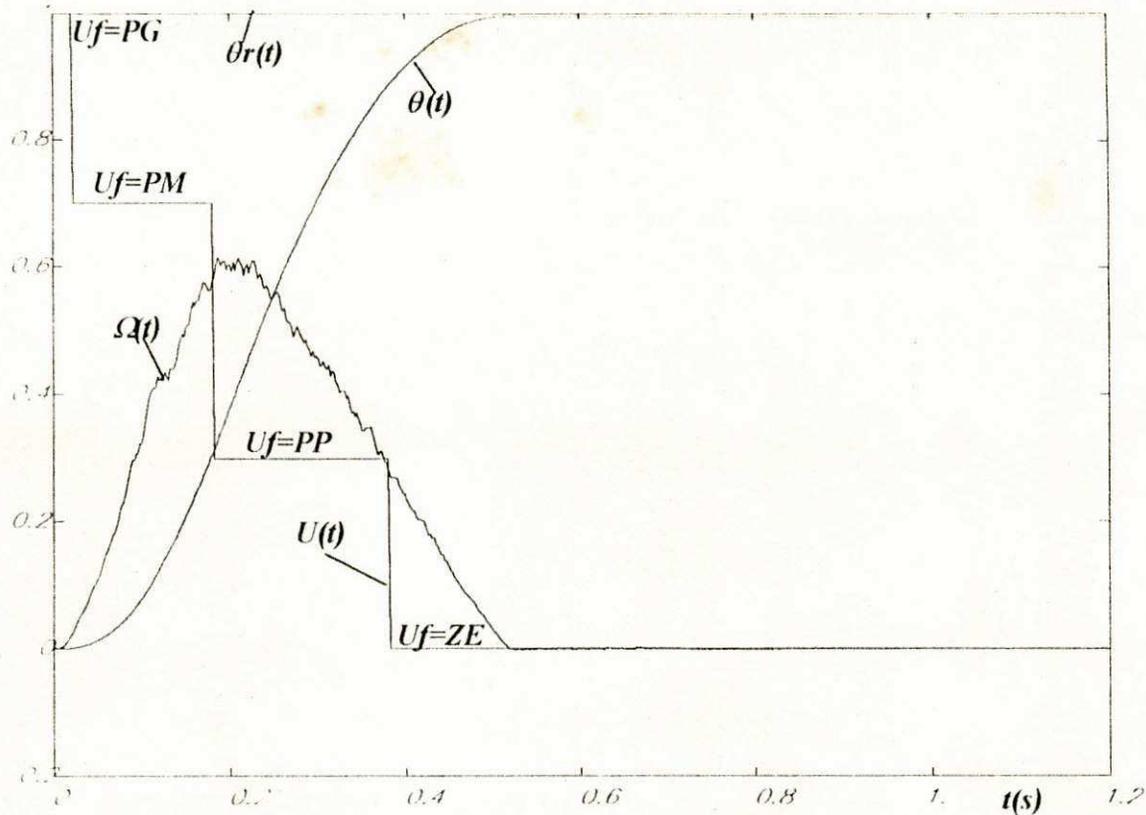


Figura B.4 - Resultados experimentais do controlador fuzzy

A seguir serão mostrados resultados experimentais usando controladores fuzzy baseados na inversa da dinâmica da planta.

### B.3 Controle fuzzy inverso

Cavalcanti et alii [1993a] propuseram um mecanismo para adaptação dos pesos das funções de pertinência dos controladores fuzzy baseado em técnicas fuzzy. Eles se basearam na metodologia de projeto denominada de centro de gravidade descrita por Sugeno & Yasukawa [1993], e no modelo fuzzy da inversa da dinâmica da planta proposto por Furuhashi et alii [1992]. Eles propuseram o conceito de PWM a níveis móveis que usa o erro  $E(t)$ , definido como a diferença entre os valores atuais e de referência da saída da planta ( $E(t) = \Omega_r(t) - \Omega(t)$ ) representada pelo motor CC, para criar níveis intermediários para  $U(t)$  gerados partir de valores de  $U_f$  com dois níveis distantes entre si. Esses níveis especiais são obtidos da variável fuzzy  $E_f$  ( $E_f = NEG$ ;  $E_f = ZE$ ;  $E_f = POS$ ), como mostrado na TABELA B.3.

TABELA B.3

$U_f$	$\Omega_{rf}=NB$	$\Omega_{rf}=NM$	$\Omega_{rf}=NS$	$\Omega_{rf}=ZE$	$\Omega_{rf}=PS$	$\Omega_{rf}=PM$	$\Omega_{rf}=PB$
$E_f=NEG$	NG	NG	NM	NP	ZE	PP	PM
$E_f=ZE$	NG	NM	NP	ZE	PP	PM	PG
$E_f=POS$	NM	NP	ZE	PP	PM	PG	PG

Na Figura B.5 são mostrados os resultados experimentais obtidos com o controlador fuzzy a nível móvel. Esse tipo de controlador fuzzy assemelha-se a um controlador PD de ganho elevado. A adaptação do peso de  $U(t)$  permite corrigir o erro estacionário e o controlador fuzzy adaptativo se assemelha ao controlador PID.

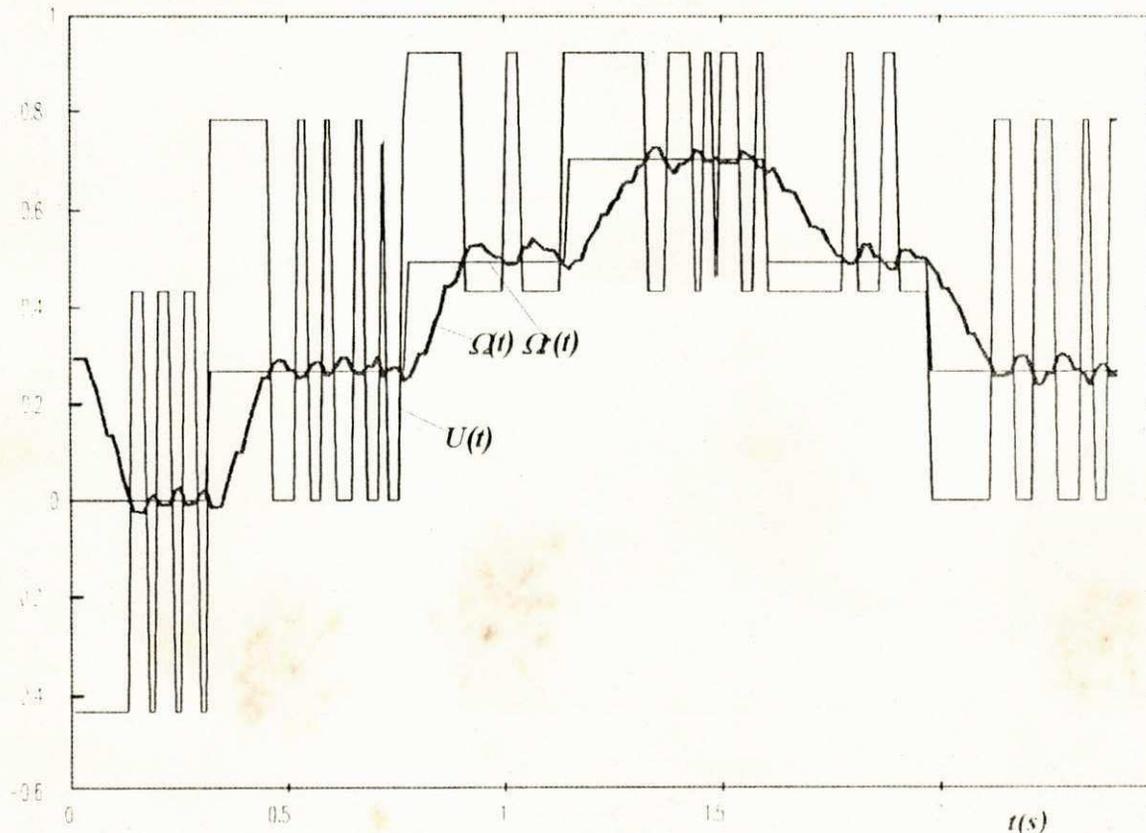


Figura B.5 - Resultados experimentais do controlador fuzzy a nível móvel.

A seguir alocaram-se 7 pesos, que podem ser adaptados, aos valores discretos de  $U(t)$  ( $P=PNG; PNM; PNP; PZE; PPP; PPM; PPG$ ). O novo valor da variável de controle será dado por  $U_p(t)=P*U(t)$ . Por exemplo, se a saída do controlador é dada por  $U_f = PM$  com o valor contínuo associado  $U(t) = 0.8$  p.u., e com  $PPM=0.9$ , então o valor atual de controle é dado por  $U_p(t) = 0.8*0.9 = 0.72$  p.u..

No controlado a nível móvel adaptativo proposto por Cavalcanti et alii [1993a], os pesos podem ser adaptados em tempo real sob algumas condições. 1)O sinal de referência deve ser constante; 2)só existem mudanças entre dois níveis de controle (PWM a nível móvel); 3)inicialmente, todos os pesos devem ter valor inicial igual a um. O algoritmo proposto por

Cavalcanti et alii [1993a] inclui a seguinte etapa: 4) aumentar o peso associado ao menor valor fuzzy da função de pertinência e diminuir o peso associado ao maior valor fuzzy da função de pertinência.

Nas REGRAS B.2, a serem executadas logo após o cálculo de  $U_f$  usando a TABELA B.3, é mostrado como se adaptar os pesos do controlador fuzzy. Os resultados experimentais obtidos com o controlador fuzzy adaptativo são mostrados na Figura B.6. Observe-se na Figura B.6 que a sequência de adaptação dos pesos do controlador fuzzy converge para a geração de um valor constante para  $U(t)$ .

#### REGRAS B.2

Se  $U_f$  atual é menor que  $U_f$  anterior, então aumente o peso associado a  $U_f$ .  
 Se  $U_f$  atual é maior que  $U_f$  anterior, então diminua o peso associado a  $U_f$ .

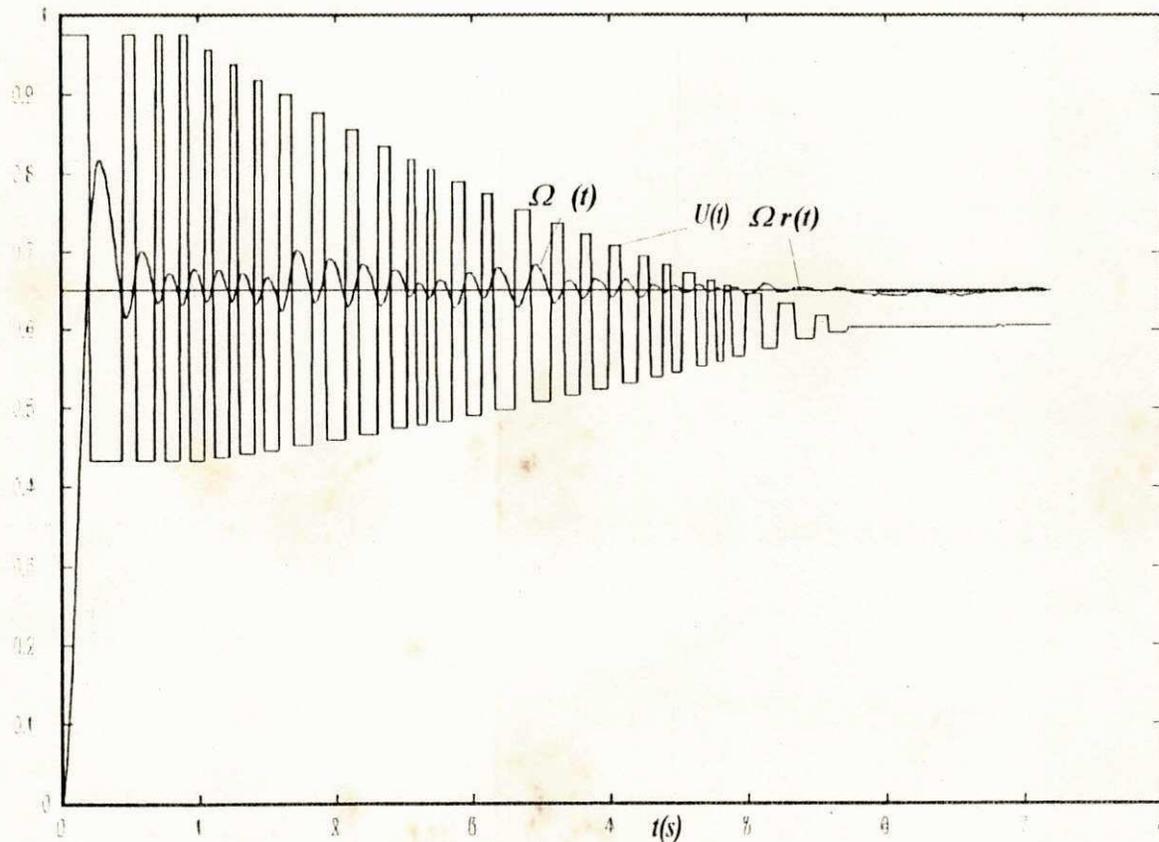


Figura B.6 - Resultados experimentais do controlador fuzzy adaptativo a nível móvel.