

UNIVERSIDADE FEDERAL DE UBERLÂNDIA
FACULDADE DE ENGENHARIA MECÂNICA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA MECÂNICA

**UMA METODOLOGIA DE PROJETO DE CONTROLADORES
HÍBRIDOS INTELIGENTES COM APLICAÇÕES NO CONTROLE ATIVO
DE VIBRAÇÕES MECÂNICAS**

Dissertação apresentada
à Universidade Federal de Uberlândia por:

RAFAEL LUÍS TEIXEIRA

como parte dos requisitos para obtenção do título de Mestre em
Engenharia Mecânica

Banca Examinadora:

Prof. Dr. José Francisco Ribeiro	(UFU) – Orientador
Prof. Dr. Agenor de Toledo Fleury	(IPT-USP)
Prof. Dr. Francisco Paulo Lépore Neto	(UFU)
Prof. Dr. Domingos Alves Rade	(UFU)
Prof. Dr. Marcus Antônio Viana Duarte	(UFU)

Uberlândia, 18 de maio de 2001.

FICHA CATALOGRÁFICA

Elaborado pelo Sistema de Bibliotecas da UFU / Setor de Catalogação e Classificação

T266m Teixeira, Rafael Luís, 1971-
 Uma metodologia de projeto de controladores híbridos inteligentes com aplicações no controle ativo de vibrações mecânicas / Rafael Luís Teixeira. - Uberlândia, 2001.
 116f. : il.
 Orientador: José Francisco Ribeiro.
 Dissertação (mestrado) - Universidade Federal de Uberlândia, Programa de Pós-Graduação em Engenharia Mecânica.
 Inclui bibliografia.
 1. Inteligência artificial - Teses. 2. Lógica difusa - Teses. 3. Redes neurais (Computação) - Teses. 4. Algoritmos genéticos - Teses. 5. Controle ativo de vibrações - Teses. 6. Engenharia mecânica - Teses. I. Ribeiro, José Francisco. II. Universidade Federal de Uberlândia. Programa de Pós-Graduação em Engenharia Mecânica. III. Título.

CDU: 681.3 : 007.52 (043.3)

A meus pais Pedro e Vânia,
a meus irmãos Renata e Pedro,
a meu tio Evandro Rafael,
e à minha mulher Flávia.

MEUS AGRADECIMENTOS

A Deus pela oportunidade de estudos.

Ao professor Dr. José Francisco Ribeiro pelo ensinamento e auxílio no decorrer da orientação e ao amigo Tito pelo companheirismo e paciência.

A todos os colegas estudantes e professores da FEMEC, de dentro e fora do Laboratório de Sistemas Mecânicos, que diretamente ou não, me apoiaram e incentivaram durante o desenvolvimento da pesquisa.

Aos colegas de laboratório, Patrick Magalhães Cardoso e Gustavo Luiz C. M. de Abreu, pela ajuda nos ensaios experimentais.

A todos os funcionários e técnicos da FEMEC que contribuíram para a execução deste trabalho.

Ao apoio financeiro oferecido pelo CAPES – Fundação Coordenação de Aperfeiçoamento de Pessoal de Nível Superior.

UMA METODOLOGIA DE PROJETO DE CONTROLADORES HÍBRIDOS INTELIGENTES COM APLICAÇÕES NO CONTROLE ATIVO DE VIBRAÇÕES MECÂNICAS

SUMÁRIO

1 – Introdução	1
2 – Metodologia de projeto de controladores híbridos inteligentes	4
3 - Fundamentos da teoria da lógica <i>fuzzy</i>	7
3.1 Teoria dos conjuntos <i>fuzzy</i>	7
3.1.1 Algumas definições em torno dos conjuntos <i>fuzzy</i>	10
3.1.2 Compensação de conjuntos <i>fuzzy</i> através de variáveis lingüísticas	12
3.1.3 Operações com os conjuntos <i>fuzzy</i> : união, intersecção e complemento	13
3.2 Teoria da lógica <i>fuzzy</i>	17
3.2.1 Proposições <i>fuzzy</i> e conectivos lógicos	18
3.2.2 Métodos de inferência da base de regras <i>fuzzy</i>	20
3.2.3 Implicações de regras <i>fuzzy</i>	21
3.2.4 Agregações de regras <i>fuzzy</i>	22
3.2.5 Propriedades das regras <i>fuzzy</i>	23
3.2.6 Algoritmo <i>fuzzy</i> de controle	25
3.2.7 Regras <i>fuzzy</i> para controle	34

4 - Fundamentos das redes neurais artificiais	39
4.1 O neurônio biológico	39
4.2 O neurônio artificial	40
4.3 Redes neurais artificiais	41
4.3.1 Redes neurais: <i>feedforward</i> x <i>feedback</i>	43
4.3.2 Redes neurais de associação linear	43
4.3.3 Treinamento de redes neurais artificiais	44
4.4 Histórico das redes neurais artificiais	45
4.4.1 As redes neurais <i>adeline</i> e <i>madeline</i>	46
4.4.2 A rede neural <i>perceptron</i>	48
4.5 O treinamento <i>backpropagation</i> e seu algoritmo	50
4.5.1 Funções de ativação para o <i>backpropagation</i>	51
4.5.2 Estágio de propagação das entradas para o treinamento <i>backpropagation</i>	52
4.5.3 Estágio de atualização dos pesos para o treinamento <i>backpropagation</i>	53
4.5.4 Metodologia de treinamento e validação de uma rede neural	57
4.6 Identificação de sistemas dinâmicos	59
5 - Fundamentos dos algoritmos genéticos	61
5.1 Codificação de um indivíduo	61

5.1.1 Codificação binária	62
5.1.2 Codificação por ponto flutuante	64
5.2 População, geração e extinção em algoritmos genéticos	64
5.3 Processo de seleção natural	67
5.3.1 Método da roleta	67
5.3.2 Método do ranking	68
5.4 Reprodução dos indivíduos	69
5.5 Cruzamento dos indivíduos	69
5.5.1 Cruzamento binário	70
5.5.2 Cruzamento aritmético	71
5.5.3 Cruzamento heurístico	71
5.6 Mutação dos indivíduos	71
5.6.1 Mutação binária	72
5.6.2 Mutação uniforme	72
5.6.3 Mutação de fronteira	73
5.6.4 Mutação não uniforme	73
5.7 AG como ferramenta de otimização de parâmetros de controladores <i>fuzzy</i>	74
5.7.1 Funções de avaliação para controladores <i>fuzzy</i>	74
5.7.2 Otimização dos pesos das regras de sistemas <i>fuzzy</i>	75

5.7.3 Otimização dos conseqüentes das regras <i>fuzzy</i>	76
5.7.4 Otimização das funções de pertinência das entradas e saídas	76
5.7.5 Otimização dos universos do discurso das entradas e saídas	77
5.7.6 Otimização simultânea de parâmetros de controladores <i>fuzzy</i>	78
6 - Identificação e controle de um sistema com um grau de liberdade	79
6.1 Identificação experimental do sistema de um grau de liberdade	81
6.1.1 Treinamento	81
6.1.2 Validação com o sistema em malha aberta	83
6.1.3 Validação com o sistema em malha fechada	83
6.2 Otimização numérica dos controladores <i>fuzzy</i> utilizando algoritmos genéticos	84
6.2.1 As estratégias de otimização	85
6.2.2 Controlador inicial	87
6.2.3 Resultados numéricos de otimização	87
6.3 Resultados dos testes experimentais no domínio do tempo	89
6.4 Caracterização do controle no domínio da frequência	91
7 - Identificação e controle de um sistema com infinitos graus de liberdade	94
7.1 Identificação experimental do sistema com infinitos graus de liberdade	97
7.1.1 Treinamento	98
7.1.2 Validação com o sistema em malha aberta	99
7.1.3 Validação com o sistema em malha fechada	100

7.2 Otimização numérica dos controladores <i>fuzzy</i> utilizando algoritmos genéticos	102
7.2.1 Definição da função objetivo	102
7.2.2 Controlador inicial e as estratégias de otimização	103
7.3 Resultados dos testes experimentais no domínio do tempo	105
7.4 Caracterização do controle no domínio da frequência	108
8 – Conclusão	112

Referências Bibliográficas

Apêndices I – II – III e IV

SIMBOLOGIA

A	-	Agregação de regras <i>fuzzy</i>
A	-	Conjunto elementar
A, B	-	Conjunto <i>fuzzy</i>
AG	-	Algoritmo genético
C	-	Complemento de um conjunto <i>fuzzy</i>
$\mathcal{F}$	-	Função de avaliação
G	-	Número de geração
G_C	-	Número da geração corrente
G_{MAX}	-	Número máximo de geração
Hgt	-	Altura de um conjunto <i>fuzzy</i>
I	-	Somatório de entrada do neurônio
IP_{MESA}	-	Índice de performance para o sistema com um grau de liberdade
IP_{VIGA}	-	Índice de performance para o sistema com infinitos graus de liberdade
J	-	Erro médio quadrático
M	-	Tamanho da cadeia binária de cada variável de otimização genética
M_p	-	Modificação lingüística
N	-	Número de iterações
NP	-	Variável lingüística negativo pequeno
NG	-	Variável lingüística negativo grande
N_r	-	Número de regras <i>fuzzy</i>
N_x	-	Número de variáveis do antecedente de uma regra
$P_1, \dots, P_n$	-	Peso ou ponderação de cada regra <i>fuzzy</i>
p	-	Parâmetro da modificação lingüística
PP	-	Variável lingüística positivo pequeno

PG	- Variável lingüística positivo grande
$\mathcal{P}_c$	- Probabilidade de cruzamento
$\mathcal{P}_M$	- Probabilidade de mutação
$\mathcal{P}_{RG}$	- Probabilidade de ranking
POP	- Tamanho da população
$\mathcal{P}$	- Probabilidade adotada em selecionar o melhor indivíduo (Roleta)
Q	- Probabilidade adotada em selecionar o melhor indivíduo (Ranking)
R	- Ranking do indivíduo
R	- Regra <i>fuzzy</i>
$rand_{1,...,8}$	- Números randômicos
S	- Norma <i>s</i> ou cotonorma <i>T</i>
Sup	- Suporte de um conjunto <i>fuzzy</i>
T	- Norma <i>t</i>
U	- Universo do discurso
U_A	- Defuzificação pelo método do centro de área
u_{CG}	- Defuzificação pelo método do centro de gravidade
u_{MM}	- Defuzificação pelo cálculo da média do máximo
u_{SUGENO}	- Defuzificação para regras do tipo Sugeno
V_R	- Valor numérico real
X, Y	Indivíduos para codificação por ponto flutuante
x, X	- Neurônios da camada de entrada
w, W	- Peso das interconexões entre neurônios
y, Y	- Neurônios da camada de saída
Yq	- Saídas desejadas
α	- Constante que determina o efeito prévio de mudança dos pesos
λ	- Parâmetro do operador complemento Sugeno
$\emptyset$	- Conjunto <i>fuzzy</i> vazio

ϕ	- Função de ativação
μ	- Função de pertinência
ε	- Erro médio quadrático
η	- Constante de proporcionalidade da taxa de aprendizado
ψ	- Função característica dos conjuntos ordinários
Φ	- Função implicação <i>fuzzy</i>
χ	- Representação do indivíduo por codificação binária

Teixeira, R. L., 2001, “Uma Metodologia de Projeto de Controladores Híbridos Inteligentes com Aplicações no Controle Ativo de Vibrações Mecânicas”, Dissertação de Mestrado, Universidade Federal de Uberlândia, Uberlândia, MG

RESUMO

Este trabalho propõe uma metodologia de projeto de controladores *fuzzy* para sistemas, que não oferecem ao projetista de controle informações intuitivas suficientes para orientá-lo na construção convencional dos controladores *fuzzy*.

A metodologia de projeto envolve a aplicação de técnicas de inteligência artificial, onde os controlares *fuzzy* são obtidos a partir de um processo de otimização que utiliza os algoritmos genéticos. Esse mecanismo de otimização, por sua vez, requer o conhecimento da dinâmica do sistema. Uma rede neural artificial é então treinada para, a partir de dados de entrada e saída do sistema, modelar o comportamento dinâmico da planta. São otimizados a base de regras, o peso das regras e as funções de pertinência de entrada.

A motivação do trabalho é o controle de vibrações em sistemas dinâmicos complexos, como, por exemplo, o controle de vibração em placas instrumentadas com vários sensores e atuadores piezelétricos.

O trabalho apresenta os fundamentos do controle *fuzzy*, das redes neurais artificiais e dos algoritmos genéticos. A metodologia de controle proposta é avaliada numérica e experimentalmente no controle de mesa vibratória de um grau de liberdade e de uma viga de aço engastada-livre. A mesa é controlada por atuadores eletromagnéticos e a viga por atuadores piezelétricos. Diversos controladores são avaliados no domínio do tempo e da frequência.

O trabalho conclui que, para os casos estudados, a metodologia proposta é eficiente e aponta, ainda, nas considerações finais, alguns desdobramentos futuros da pesquisa realizada.

Palavras chave: Lógica Fuzzy, Redes Neurais Artificiais, Algoritmos Genéticos, Controle Ativo de Vibrações

Teixeira, R. L., 2001, "A Design Methodology of Intelligent Hybrid Controllers with Applications to the Active Vibrations Control in Mechanical Systems", M. Sc. Dissertation, Universidade Federal de Uberlândia, Uberlândia, MG.

ABSTRACT

This work proposes a design methodology of fuzzy controllers for systems that do not offer any intuitive information to the control designer, in order to provide guidelines to construct the fuzzy conventional controllers.

The design methodology involves the application of artificial intelligence techniques, where the fuzzy controllers are obtained by an optimization process that uses genetic algorithms. For this optimization procedure, the knowledge of the system dynamics is required. From the experimental inputs and outputs of the system, an artificial neural network is trained and, in that way, it is possible to model the dynamic behavior of the plant. The rule base, the weights of the rules and the input membership functions are optimized.

The goal of this methodology is the control the vibrations of complex dynamic systems, such as vibration control of plates instrumented with piezoelectric sensors and actuators.

In this work, the fundamentals of fuzzy control, artificial neural networks and genetic algorithms are presented. The proposed control methodology is evaluated numerically and experimentally on the control of a vibratory system with one degree of freedom, and on a steel cantilever test beam. The first system is controlled by electromagnetic actuators and the beam is controlled by piezoelectric actuators. Many controllers are evaluated in time and frequency domains.

For the studied cases, it is concluded that the proposed methodology is efficient and also some considerations are made about the future works based of the presented search.

Keywords: Fuzzy Logic, Artificial Neural Networks, Genetic Algorithms, Active Vibrations Control

Capítulo I

Introdução

Os sistemas mecânicos, sejam máquinas industriais, construções civis, veículos de transporte etc., estão freqüentemente sujeitos as excitações internas ou externas, que provocam vibrações indesejáveis, perturbando operadores e usuários e, em alguns casos, colocando em risco a própria integridade estrutural do sistema. Este fenômeno tem mobilizado um conjunto expressivo de pesquisadores e são inúmeras as publicações especializadas na área.

Várias estratégias têm sido propostas para a atenuação de vibrações em sistemas mecânicos. Há soluções simples que empregam materiais viscoelásticos como amortecedores passivos (Nashif *et al.*, 1985) e propostas sofisticadas baseadas em materiais "inteligentes" e técnicas de controle moderno. De uma maneira geral as técnicas de controle utilizadas podem ser classificadas em dois grandes grupos: técnicas de controle passivo e técnicas de controle ativo.

Como técnicas passivas, destacam-se as que empregam os absorvedores dinâmicos de vibrações (ADVs) passivos (Den Hartog, 1956 ; Korenev e Reznikov, 1993). Recentemente, estudos de diversas configurações de ADVs passivos foram realizados por Cunha Jr. (1999) e Steffen e Rade (2000).

Uma limitação importante dos ADVs passivos está no fato de seus parâmetros construtivos serem definidos para operar em uma freqüência fixa (ou em uma faixa bem estreita). Quando ocorrem excitações fora da faixa de sintonia do ADV, como freqüentemente acontece na prática, esse perde grande parte de sua eficiência.

Esta limitação dos ADVs passivos deu origem aos absorvedores dinâmicos adaptativos que alteram as suas características (inércia, amortecimento, rigidez), adaptando-se a diferentes freqüências de excitação dentro de uma faixa de operação (Sun *et al.*, 1995).

Quando um ADV incorpora um elemento ativo, cuja ação depende do comportamento instantâneo do sistema, tem-se um ADV ativo. Olgac e Holsek (1997) propõem um ADV desta natureza, cujo atuador é realimentado por uma combinação dos sinais de velocidade e aceleração do sistema primário. Em Marques (2000) é feita uma ampla revisão deste tipo de absorvedor.

Os avanços tecnológicos observados nas últimas décadas na área de materiais abriram novas perspectivas para o controle de vibrações. Os materiais ditos "inteligentes", que podem ter algumas de suas propriedades modificadas mediante uma ação de controle, saíram dos laboratórios de pesquisa e ganharam o mundo comercial. Destaque especial para os materiais piezelétricos, que têm possibilitado a concepção de tipos inovadores de sensores e atuadores. A distribuição de sensores e atuadores piezelétricos, ao longo de amplas superfícies, permitindo o monitoramento e o controle dos níveis de vibração é hoje uma realidade concreta (Rogers, 1992; Banks *et al.*, 1996).

Nas últimas décadas, paralelamente ao desenvolvimento de materiais, surgiram novos elementos teóricos na área de controle. Consolidaram-se os controladores robustos e adaptativos, que apropriaram os conceitos matemáticos dos controladores clássicos e os enriqueceram com uma abordagem estocástica. Como exemplos desta estratégia, aplicadas ao controle de vibrações, temos os controladores H_∞ , LQG/LTR e H_2 , Ribeiro (1995).

Diante da necessidade de estabelecer instrumentos de análise e síntese de controladores voltados para sistemas não lineares, complexos e variantes no tempo, surgiu, mais recentemente, uma nova vertente de pesquisa e de desenvolvimento de controladores. Emergiram os sistemas híbridos inteligentes baseados no uso de duas ou mais tecnologias de inteligência artificial.

Esse novo campo é caracterizado por uma certa tolerância à imprecisão e ambigüidades decorrentes de variações paramétricas, perturbações dinâmicas, alterações ambientais, ignorância dos modelos. É, neste contexto, que se desenvolveu o conceito dos sistemas híbridos inteligentes. Tais sistemas são entendidos como aqueles que percebem alterações ambientais e/ou de funcionamento e ajustam-se às novas condições de trabalho, servindo-se, nesta adaptação, das denominadas "tecnologias de inteligência artificial". São exemplos destas tecnologias: as redes neurais artificiais, os algoritmos genéticos e a lógica *fuzzy* (Uhrig e Tsoukalas, 1997).

Muitas destas tecnologias, curiosamente, tiveram como base a observação de fenômenos biológicos. O controle *fuzzy*, por exemplo, imita a intuição e/ou a consciência humana a partir do conhecimento heurístico do sistema a ser controlado. As redes neurais artificiais, por sua vez, são baseadas nos modelos da estrutura do córtex cerebral, enquanto que os algoritmos genéticos baseiam-se na teoria Darwiniana da evolução das espécies.

No campo das vibrações, em especial, encontra-se um vasto número de aplicações que utilizam as tecnologias da inteligência artificial. Destacam-se: o controle de vibração estrutural de grande antenas espaciais (Hadaegh *et al.*, 1994), o controle de vibrações em

elevadores de alta velocidade (Ling e Xu, 1996), o controle de robôs com braços flexíveis (Moudgal *et al.* 1994), o controle de suspensão semi-ativa veicular (Wu e Xu, 1999; Yoshimura, 1998; Huang e Lian, 1996). Empregando a lógica *fuzzy* no controle de vibrações de um sistema massa-mola comando por atuadores eletromagnéticos, temos os trabalhos de Teixeira e Ribeiro (2000; 2001).

Atualmente, muitos pesquisadores investigam a possibilidade de integração de duas ou mais tecnologias de inteligência artificial na resolução prática de problemas complexos. É neste contexto que este trabalho se insere, qual seja, o da síntese de sistemas híbridos de controle.

Procura-se neste trabalho reunir algumas ferramentas da lógica *fuzzy*, das redes neurais artificiais e dos algoritmos genéticos para a solução do problema de controle de vibrações em sistemas mecânicos. Assim, são dois os objetivos principais deste trabalho:

- *propor uma metodologia de projeto de controladores híbridos inteligentes, centrados na lógica fuzzy, nos algoritmos genéticos e nas redes neurais artificiais, que dispense o modelo matemático e o conhecimento intuitivo do comportamento dinâmico do sistema a ser controlado, e*
- *avaliar a metodologia proposta, numérica e experimentalmente, quando aplicada no controle ativo de vibrações de sistemas dinâmicos.*

Para atender estes objetivos, esse trabalho está assim organizado: no Capítulo II é apresentada a metodologia de projeto do controlador; nos capítulos III, IV e V discutem-se os fundamentos do controle *fuzzy*, das redes neurais e dos algoritmos genéticos respectivamente. O Capítulo VI mostra a metodologia aplicada no controle de um sistema vibratório de um grau de liberdade. O Capítulo VII avalia a metodologia aplicada ao controle de uma viga de aço flexível engastada numa das extremidades e controlada por atuadores piezelétricos e o Capítulo VIII apresenta as conclusões e os desdobramentos futuros deste trabalho.

Capítulo II

Metodologia de projeto de controladores híbridos inteligentes

Na última década, o emprego de redes neurais artificiais e da lógica *fuzzy* na solução de problemas de Engenharia, cresceu de forma muito significativa. Mais recentemente têm sido propostas estratégias de solução que unem essas áreas de conhecimento e acrescentam outras como os algoritmos genéticos, os sistemas caóticos etc. Estas estratégias constituem, segundo Tsoukalas e Uhrig (1997), os sistemas híbridos, baseados em técnicas de inteligência artificial.

Estes sistemas operam com plantas complexas, não lineares e pobremente conhecidas. São capazes de "perceber" alterações ambientais e/ou de funcionamento e de se adaptarem a diferentes condições de operação.

A lógica *fuzzy* faz uso da teoria dos conjuntos *fuzzy*, proposta em 1965, por Lofti Zadeh. Foi Mamdani (1974,1977a,1977b), um dos pioneiros na formulação dos controladores *fuzzy*. Mamdani inspirou os seus trabalhos nos artigos de Zadeh (1968,1972,1973), que propunham uma forma lingüística para analisar sistemas baseada na teoria dos conjuntos *fuzzy*.

Como características gerais os controladores *fuzzy* são naturais e intuitivos na sua formulação, pois *procuram imitar* o comportamento consciente e a estratégia de controle de um operador humano. Dispensam o conhecimento detalhado do sistema físico. São de fácil implementação e aplicam-se a sistemas lineares e não lineares. Diversas aplicações práticas podem ser encontradas, como por exemplo: o monitoramento e o controle quantitativo de anestésias pré-operatórias (Linkens e Abbod, 1998), a operação automática de trens (Yasunobu,1985) e de guindastes para contêiner (Yasunobu,1986), o controle da qualidade de água (Yagishitta *et al.*,1985). Mendel (1995) reporta uma série de aplicações industriais como o controle de aeronaves (Rockweel Corporation), transmissão automática veicular (Subaru,Nissan), piloto automático (Nissan), a programação de elevadores prediais (Hitachi, Fujitech e Mitsubishi), auto-foco de câmeras fotográficas (Sanyo/Fisher e Canon), ajuste de imagem de televisores (Sony), reconhecimento de escrita manual (Sony Palm Top), plataforma de lançamentos de foguete (Nasa), etc.

Em muitas circunstâncias, no entanto, especialmente em se tratando de sistemas complexos e não lineares, não é possível estabelecer um comportamento intuitivo para o

sistema que se deseja controlar. Não há um operador a ser "*imitado*" pelo controlador *fuzzy* pois o operador "não sabe" tratar de forma intuitiva a complexidade do problema. Pode-se, neste caso, propor-se um controlador *fuzzy*. No entanto, a possibilidade dele ser eficiente é obviamente muito remota. O desafio que se coloca é como escolher um controlador *fuzzy* eficiente para um sistema complexo, na ausência de informações intuitivas sobre tal sistema. Neste ponto é que surgem os algoritmos genéticos.

Os algoritmos genéticos (Holland, 1992) procuram, de certa maneira, imitar o processo de evolução natural observado na natureza. Uma população de indivíduos, um conjunto de parâmetros, evolui num processo de aprendizado, onde os indivíduos mais aptos sobrevivem e transmitem, por reprodução, suas características hereditárias para a população seguinte. Esta evolução, que comporta cruzamentos e mutações, resulta numa população vencedora representando o conjunto ótimo de parâmetros que maximiza uma determinada função de custo. A formulação matemática desta evolução configura o algoritmo genético (Goldenberg, 1989).

Destacam-se algumas aplicações de algoritmos genéticos, tais como López (1999) que os utiliza no controle de trajetória de robô, Delben *et al.* (1996) aplica-os no restabelecimento ótimo de energia em sistemas de distribuição. Na otimização de parâmetros de controladores *fuzzy* destaca-se os trabalhos de Tsang e Yeung (1999), Hwang e Zein-Sabatto (1997), Heider e Drabe (1997), Carse *et al.* (1996), dentre outros.

Os algoritmos genéticos necessitam, no entanto, do modelo do sistema dinâmico para que seus indivíduos (as soluções propostas) possam ser avaliados. Tal modelo pode ser obtido a partir de dados conhecidos de entrada e saída do sistema físico. As redes neurais artificiais (RNAs), por exemplo, podem fazer o mapeamento entre a entrada e a saída, configurando-se como um modelo do sistema físico.

São muitas as áreas tecnológicas em que a teoria das redes neurais artificiais são empregadas. Wang e Mendel (1992) utilizam as redes neurais artificiais na identificação de sistemas não lineares. Raza *et al.* (1994) as utilizam para detectar falhas em superfícies de controle de aeronaves de alto desempenho. Uhrig *et al.* (1994) as empregam na análise e diagnóstico de falhas em reatores nucleares. Hanes (1994) as utilizam na robótica. Snyder e Tanaka (1995) e Abreu *et al.* (2000), usam as RNAs no controle ativo de vibrações e Carrara (1999) no controle de atitude de satélites. Tsoukalas *et al.* (1992) reportam dezenas de aplicações das redes neurais na área de reconhecimento e tratamento de imagens, na área médica, em engenharia de transportes, na área financeira, etc.

À luz do acima exposto propõe-se neste trabalho investigar a seguinte metodologia para a síntese de um controlador:

- dado um sistema a ser controlado e devidamente instrumentado, excita-se o mesmo com sinais de entrada conhecidos e registram-se os sinais de saída gerados;
- a partir dos sinais de entrada e saída constrói-se e valida-se um modelo neural do sistema dinâmico, utilizando a teoria das redes neurais artificiais;
- validado o modelo neural do sistema, define-se a arquitetura de um controlador *fuzzy*, observando: o número de entradas e saídas, o intervalo de variações dos sinais envolvidos, o número e a natureza (triangular, gaussiana etc) das funções de pertinência,
- define-se o comportamento dinâmico desejado para sistema, operando sob a ação do controlador a partir de certas condições de contorno especificadas (condições iniciais, tempo de simulação, etc.)
- define-se uma função objetivo que, uma vez minimizada, resulte no comportamento definido no item anterior. A função objetivo deve ser capaz de "medir" a distância entre o comportamento real do sistema e o comportamento desejado durante o intervalo de simulação;
- seleciona-se, via algoritmo genético, os controladores *fuzzy*, modificando alguns de seus parâmetros (regras, pesos, parâmetros das funções de pertinência etc), que minimizem a função objetivo definida anteriormente. Para isso simula-se o comportamento do sistema físico, sob a ação do controle, utilizando o modelo neural anteriormente validado e, finalmente,
- verifica-se experimentalmente o desempenho do(s) controlador(es) proposto(s)

Para uma melhor compreensão desta metodologia veremos, nos capítulos que se seguem os fundamentos teóricos sob os quais se assentam esta proposta. Os leitores familiarizados com os conceitos básicos dos controladores *fuzzy*, das redes neurais e dos algoritmos genéticos podem se reportar diretamente aos Capítulos VI e VII que avaliam e discutem numericamente e experimentalmente a metodologia de projeto proposta.

Capítulo III

Fundamentos da Lógica *Fuzzy*

Este capítulo apresenta uma visão geral da teoria dos conjuntos *fuzzy* e da teoria da lógica *fuzzy*. A aplicação destas teorias na formulação dos controladores *fuzzy* também é discutida e vários exemplos ilustram os conceitos teóricos apresentados.

3.1 Teoria dos Conjuntos *Fuzzy*

Como uma proposta de generalização da teoria clássica dos conjuntos ordinários, Lofti A. Zadeh introduziu, em 1965, os conceitos fundamentais da teoria dos conjuntos *fuzzy*.

Para compreender a proposta de Zadeh, suponha um conjunto A definido dentro de um universo de discurso U ¹.

Segundo a teoria elementar dos conjuntos, para qualquer conjunto A existe uma função característica denotada por $\Psi_A(x)$, tal que:

$$\Psi_A(x) = \begin{cases} 1 & \Leftrightarrow x \in A \\ 0 & \Leftrightarrow x \notin A \end{cases} \quad (3.1a)$$

ou,

$$\Psi_A(x): U \Rightarrow \{0,1\} \quad (3.1b)$$

Desta forma, um elemento é considerado membro do conjunto A se $\Psi_A(x)=1$ ou não membro se $\Psi_A(x)=0$. Entretanto, em muitas situações é difícil afirmar claramente se um elemento x é membro ou não de um determinado conjunto. Por exemplo, suponha que se queira classificar pessoas em três conjuntos: crianças, adolescentes e adultos. É claro que um indivíduo de 17 anos é um adolescente. Porém não há critérios precisos que definam a fronteira deste conjunto, ou seja, um indivíduo de 12 ou 25 anos é adolescente ou não? Seria o indivíduo de 12 anos uma criança e o indivíduo de 25 anos um adulto? Percebe-se a existência de um intervalo vago na fronteira destes conjuntos, que dificulta a classificação dos indivíduos.

¹ Universo do Discurso é o espaço ou domínio das variáveis de um conjunto.

Foi diante deste tipo de dificuldade que emergiram os conjuntos *fuzzy* que admitem para as funções características um índice (ou grau de pertinência), que pode variar no intervalo real: $[0,1]$.

Na teoria dos conjuntos *fuzzy*, o conjunto A passa a ser caracterizado por sua função característica, também chamada função de pertinência e denotada por $\mu_A(x)$, assim definida:

$$\mu_A(x): U \Rightarrow [0,1] \quad (3.2)$$

onde U é o universo do discurso de um elemento particular x . Uma representação possível para o conjunto *fuzzy* A é:

$$A = \{(x, \mu_A(x))\}, x \in U \quad (3.3)$$

onde cada par do conjunto A é formado por um elemento x e o seu respectivo valor de pertinência $\mu_A(x)$. Este par é denominado “*singleton*” por muitos autores.

Um conjunto *fuzzy* pode ser também representado pela seguinte notação discreta:

$$A = \sum_{i=1}^m \frac{\mu_A(x_i)}{x_i}, x_i \in U \quad (3.4a)$$

$$A = \frac{\mu_A(x_1)}{x_1} + \frac{\mu_A(x_2)}{x_2} + \dots + \frac{\mu_A(x_m)}{x_m} \quad (3.4b)$$

onde $\mu_A(x)$ é a função de pertinência e U o universo do discurso.

Quando U for definido contínuo e não finito, então a representação 3.4a, pode ser escrita como:

$$A = \int_U \frac{\mu_A(x_i)}{x_i} \quad (3.5)$$

onde a função de pertinência $\mu_A(x)$ é uma função contínua $\mu_A(x) = f(x)$ definida em U .

Salienta-se que nas equações 3.4 e 3.5, os sinais de divisão, somatório e integral são simbologias adotadas para descrição da teoria dos conjuntos *fuzzy*, ou seja, o símbolo de integral não possui o mesmo significado adotado pelo cálculo diferencial integral.

Como exemplo, considere três conjuntos *fuzzy* formados por crianças, adolescentes e adultos. Seja o universo do discurso definido para indivíduos até 40 anos, então é possível construir os seguintes conjuntos discretos:

$$Criança = \frac{1}{0} + \frac{1}{5} + \frac{1}{10} + \frac{0,75}{12} + \frac{0,25}{16} + \frac{0}{25} + \frac{0}{30} + \frac{0}{35} + \frac{0}{40} \quad (3.6a)$$

$$Adolescente = \frac{0}{0} + \frac{0}{5} + \frac{0}{10} + \frac{0,75}{16} + \frac{1}{18} + \frac{0,75}{21} + \frac{0,25}{27} + \frac{0}{30} + \frac{0}{35} + \frac{0}{40} \quad (3.6b)$$

$$Adulto = \frac{0}{0} + \frac{0}{5} + \frac{0}{10} + \frac{0}{18} + \frac{0,25}{21} + \frac{0,75}{27} + \frac{1}{30} + \frac{1}{35} + \frac{1}{40} \quad (3.6c)$$

Define-se o diagrama de Zadeh como a representação gráfica dos conjuntos *fuzzy*. A Figura 3.1 mostra a representação de Zadeh dos três conjuntos definidos por 3.6, tanto na forma discreta como na forma contínua.

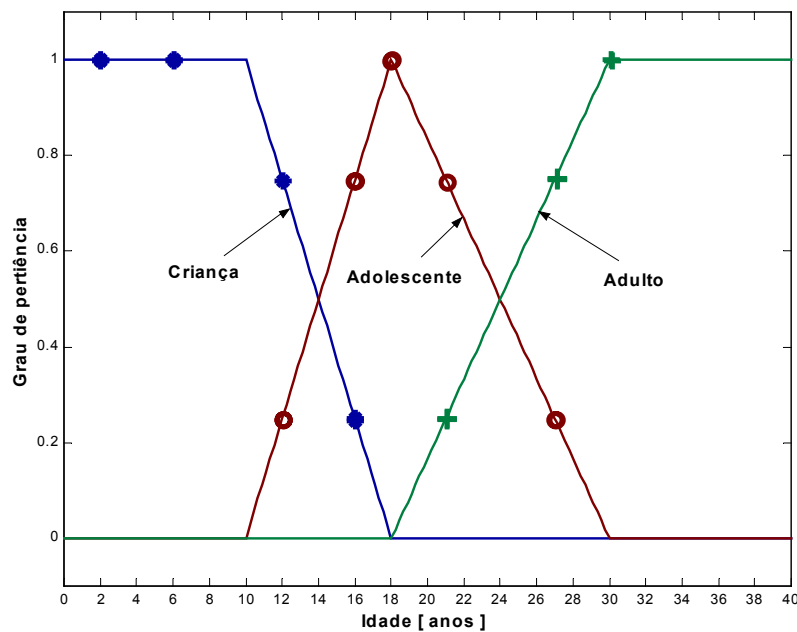


Figura 3.1: Diagrama dos conjuntos *fuzzy*: crianças, adolescentes e adultos

Da Figura 3.1 pode-se observar que um indivíduo de 21 anos, segundo a teoria dos conjuntos *fuzzy*, é 0.75 adolescente e 0.25 adulto.

3.1.1 Algumas definições em torno dos conjuntos *fuzzy*

A altura de um conjunto *fuzzy* A , $hgt(A)$, é definida pela relação:

$$hgt(A) = \max_{x \in U} [\mu_A(x)] \quad (3.7)$$

onde a designação *max* significa supremo ou o maior valor de pertinência. O conjunto *fuzzy* que apresentar pelo menos um elemento x_0 tal que $\mu_A(x_0)=1$, é chamado de conjunto *fuzzy normal* e neste caso $hgt(A)=1$. O conjunto caracterizado por $hgt(A)<1$ é denominado *subnormal*. Um conjunto *fuzzy* é chamado de *vazio* ($\emptyset$) se e somente se $\mu_A(x)=0$ para qualquer x pertencente ao universo do discurso.

O núcleo de um conjunto *fuzzy*, $nucl(A)$, é um subconjunto de A definido pela relação:

$$nucl(A) = \{x \in U \mid \mu_A(x) = 1\} \quad (3.8)$$

e o suporte de um conjunto *fuzzy*, $sup(A)$, é também um subconjunto de A definido pela relação:

$$sup(A) = \{x \in U \mid \mu_A(x) > 0\} \quad (3.9)$$

Se o suporte de um conjunto *fuzzy* for finito então a terminologia passa a ser suporte compacto. A figura 3.2 apresenta a altura, o núcleo e o suporte de um conjunto *fuzzy*.

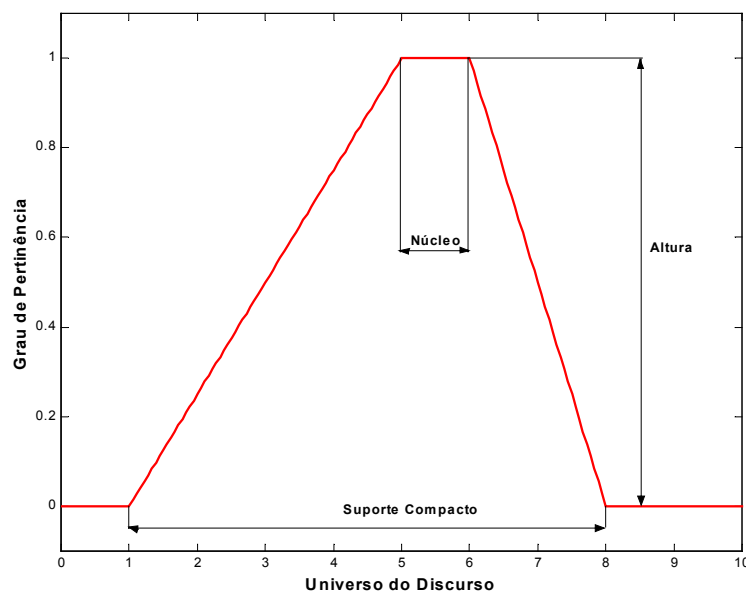


Figura 3.2: Altura, Núcleo e Suporte de um conjunto *fuzzy*

Os elementos do universo do discurso que possuírem $\mu_A(x)=0.5$ são chamados de pontos de cruzamento. O α_{cut} de um conjunto *fuzzy* A é o conjunto de todos elementos do universo do discurso U para os quais o valor de pertinência é maior ou igual a α :

$$\alpha_{cut}(A) = \{x \in X \mid \mu_A(x) \geq \alpha\} \quad (3.10)$$

onde α é um parâmetro definido no domínio $0 < \alpha \leq 1$.

Uma propriedade dos conjuntos *fuzzy* é a *convexidade*. Um conjunto A é convexo se:

$$\forall x_1, x_2, x_3 \in U \mid x_1 \leq x_2 \leq x_3 \Rightarrow \mu_A(x_2) \geq \min[\mu_A(x_1) ; \mu_A(x_3)] \quad (3.11)$$

Um exemplo importante de conjuntos *fuzzy* são os números *fuzzy*. O número *fuzzy* cinco, por exemplo, pode ser definido conforme o diagrama da Figura 3.3 (a). Os conjuntos dos números *fuzzy* possuem as seguintes características:

- (i) o conjunto *fuzzy* é *convexo*;
- (ii) o conjunto *fuzzy* é *normal*;
- (iii) as funções características são contínuas;
- (iv) o núcleo do conjunto *fuzzy* consiste em apenas um valor.

Ainda sobre os números *fuzzy* Dubois e Prade (1988), definem os intervalos dos números *fuzzy*, onde as características acima enunciadas são todas válidas exceto a que diz que o núcleo é restrito a um único ponto. Um exemplo de um intervalo *fuzzy* é mostrado no diagrama da Figura 3.3 (b).

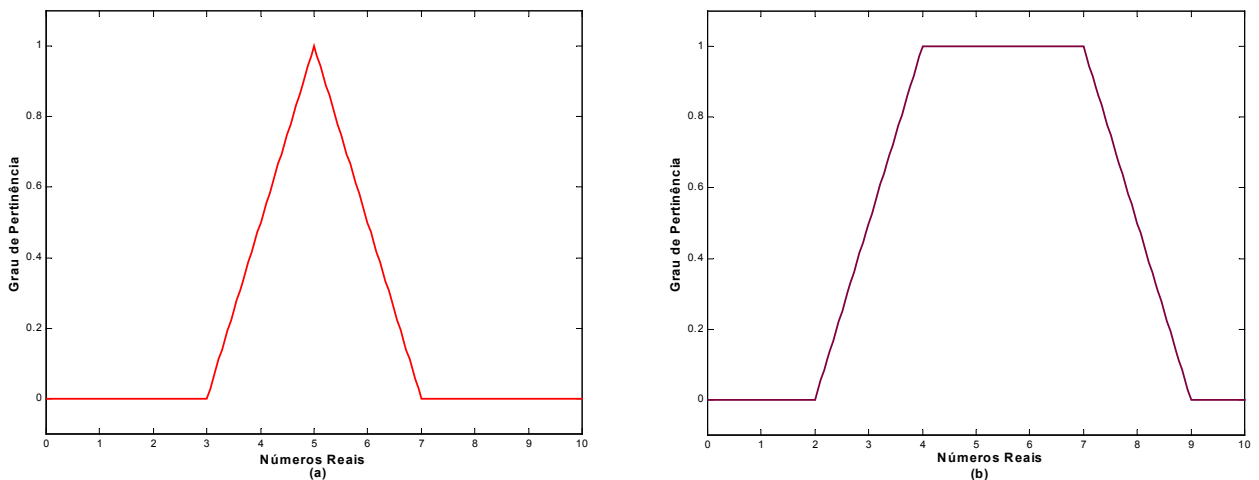


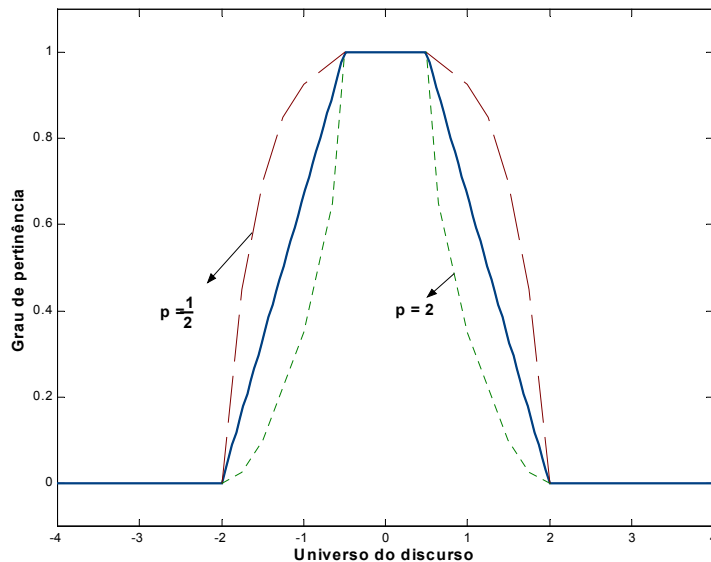
Figura 3.3 (a) Número *fuzzy* cinco e (b) Intervalo *fuzzy* entre os números *fuzzy* 4 e 7

3.1.3 Compensação de conjuntos *fuzzy* através de variáveis lingüísticas

Modificações lingüísticas podem ser utilizadas para alterar as características de um conjunto *fuzzy*. Por exemplo, a modificação lingüística *muito* pode ser usada para mudar o sentido de um conjunto *fuzzy* definido como *grande* para *muito grande*. Muitos autores (Hellendoorn, 1990; Zimmermann, 1985; Lakoff, 1973) têm definido essas modificações ou compensações de maneiras distintas. As compensações de potência operam com o grau de pertinência dos conjuntos *fuzzy* e são representadas por:

$$m_p(A) = \int_x \mu_A^p(x) / x \quad (3.12)$$

onde m_p é a modificação lingüística e $p \in (0,1]$ é o parâmetro específico de uma certa modificação lingüística. Para exemplificar suponha um conjunto *fuzzy* A definido como sendo os números reais *próximos da origem* e a função característica do tipo trapezoidal. Aplicando a compensação de potência pode-se obter outros conjuntos. Quando p for igual a 2 tem-se o conjunto dos números *muito próximos da origem* e quando $p = 1/2$ tem-se o conjunto dos números *mais ou menos próximos da origem*. Esses dois casos podem ser visualizados na Figura 3.4.



$$muito(A) = \int_x \mu_A^2(x) / x$$

$$mais\ ou\ menos(A) = \int_x \mu_A^{1/2}(x) / x$$

Figura 3.4: Conjunto dos números *próximos da origem* (curva contínua), *muito próximo da origem* ($p=2$) e *mais ou menos próximo da origem* ($p=1/2$) para compensação de potência.

Diante destas possibilidades de compensação, ou seja, variações do parâmetro p , algumas propriedades são derivadas:

$$\text{Conjunto fuzzy não modifica} \quad p = 1 : m_p(A) = A \quad (3.13a)$$

$$\text{Conjunto fuzzy sofre contração} \quad p > 1 : m_p(A) \subset A \quad (3.13b)$$

$$\text{Conjunto fuzzy sofre dilatação} \quad 0 < p < 1 : m_p(A) \supset A \quad (3.13c)$$

Uma característica importante desta compensação potencial é que tanto o suporte como o núcleo do conjunto *fuzzy* não se modificam, uma vez que $0^p = 0$ e $1^p = 1$.

Existem maneiras de compensar um conjunto *fuzzy* que modificam o núcleo e o suporte de um conjunto *fuzzy*. É o caso da compensação de deslocamento discutida em Pétié *et al.* (1990).

3.1.3 Operações com os conjuntos *fuzzy*: união, intersecção e complemento

Na teoria clássica dos conjuntos, são definidas as operações de união e intersecção de dois conjuntos, assim como o complemento de um conjunto.

Tabela 3.1: Operações com conjuntos segundo a teoria clássica dos conjuntos

A	B	$A \cup B$	$A \cap B$	$\overline{A}$
0	0	0	0	1
0	1	1	0	1
1	0	1	0	0
1	1	1	1	0

Nesta seção as operações de união e intersecção, bem como o complemento de um conjunto, serão estendidas para os conjuntos *fuzzy*.

A união e intersecção de conjuntos *fuzzy* são definidas a partir dos valores de pertinência das funções características dos conjuntos. Zadeh (1965) propõe as seguintes definições para a união e para a intersecção:

$$\mu_A \cup \mu_B = \max[\mu_A(x) ; \mu_B(x)] \quad (3.14a)$$

$$\mu_A \cap \mu_B = \min[\mu_A(x) ; \mu_B(x)] \quad (3.14b)$$

Caso se restrinja $\mu_A(x)$ e $\mu_B(x)$ a valores $\{0,1\}$, esses operadores passam a ter características da teoria dos conjuntos, o que revela que os conjuntos clássicos são na verdade um caso especial dos conjuntos *fuzzy*. Geralmente as operações de união e intersecção entre conjuntos *fuzzy* fazem uso, respectivamente, da norma triangular (*norma-T*) e da co-norma triangular (*conorma-T* ou *norma-S*).

A *norma-T* é uma função definida de $\Re^2$ para $\Re^1$, dentro dos domínios: $[0,1] \times [0,1] \rightarrow [0,1]$, e que satisfaz os seguintes axiomas:

$$\mathbf{T-1} \quad T(a,0) = 0 ; T(a,1) = a \quad \forall a \in [0,1] \quad (3.15a)$$

$$\mathbf{T-2} \quad T(a,b) = T(b,a) \quad \forall a,b \in [0,1] \quad (3.15b)$$

$$\mathbf{T-3} \quad T(T(a,b),c) = T(a,T(b,c)) \quad \forall a,b,c \in [0,1] \quad (3.15c)$$

$$\mathbf{T-4} \quad \text{Se } a \leq b \text{ então } T(a,c) \leq T(b,c) \quad \forall a,b,c \in [0,1] \quad (3.15d)$$

No primeiro axioma são apresentados as condições de fronteira da *norma-T*. Os dois axiomas subseqüentes são as leis da comutação e da associação. O último axioma exige a preservação da ordem e garante que a ordem de avaliação da norma não pode ser contraditória a uma terceira avaliação.

De forma quase análoga, a *norma-S* é uma função definida de $\Re^2$ para $\Re^1$, dentro dos domínios: $[0,1] \times [0,1] \rightarrow [0,1]$, e que satisfaz os seguintes axiomas:

$$\mathbf{S-1} \quad S(a,0) = a ; S(a,1) = 1 \quad \forall a \in [0,1] \quad (3.16a)$$

$$\mathbf{S-2} \quad S(a,b) = S(b,a) \quad \forall a,b \in [0,1] \quad (3.16b)$$

$$\mathbf{S-3} \quad S(S(a,b),c) = S(a,S(b,c)) \quad \forall a,b,c \in [0,1] \quad (3.16c)$$

$$\mathbf{S-4} \quad \text{Se } a \leq b \text{ então } S(a,c) \leq S(b,c) \quad \forall a,b,c \in [0,1] \quad (3.16d)$$

Vários autores propuseram diferentes fórmulas que aplicam as *norma-T* e *norma-S* para operações de união e intersecção respectivamente, como mostra a Tabela 3.2.

Tabela 3.2: Operações de união (*norma-T*) e de intersecção (*norma-S*) de dois conjuntos *fuzzy*

$T(a,b)$	$S(a,b)$	Condição	Referências
$\min(a,b)$	$\max(a,b)$		Zadeh (1965)
ab	$a+b-ab$		Bandler e Kohout (1980)
$\max(a+b-1,0)$	$\min(a+b,1)$		Lukasiewicz, Giles (1976)
$\frac{ab}{\gamma + (1-\gamma)(a+b-ab)}$	$\frac{a+b-(2-\gamma)ab}{1-(1-\gamma)ab}$	$\gamma > 0$	Hamacher (1978)
$\frac{ab}{\max(a,b,\alpha)}$	$\frac{a+b-ab-\min(a,b,1-\alpha)}{\max(1-a,1-b,\alpha)}$	$\alpha \in [0,1]$	Dubois e Prade (1986)
$\left[1 + \sqrt[\lambda]{\left(\frac{1}{a}-1\right)^{-\lambda} + \left(\frac{1}{b}-1\right)^{-\lambda}}\right]^{-1}$	$\left[1 + \sqrt[\lambda]{\left(\frac{1}{a}-1\right)^{\lambda} + \left(\frac{1}{b}-1\right)^{\lambda}}\right]^{-1}$	$\lambda > 1$	Dombi (1982)
$1 - \sqrt[p]{(1-a)^p + (1-b)^p - (1-a)^p(1-b)^p}$	$\sqrt[p]{a^p + b^p - a^p b^p}$	$p > 1$	Schweizer e Sklar (1961)
$\max\left(1 - \sqrt[p]{(1-a)^p + (1-b)^p}, 0\right)$	$\min\left(\sqrt[p]{a^p + b^p}, 1\right)$	$p \geq 1$	Yager (1980b)
$\max\left(\frac{a+b-1+\lambda ab}{1+\lambda}, 0\right)$	$\min(a+b+\lambda ab, 1)$	$\lambda \geq -1$	Weber (1983)

A representação original, segundo Zadeh, para o complemento ($\bar{A}$) de um conjunto *fuzzy* A é definido como:

$$\bar{A} = C(A) = 1 - \mu_A(x) \quad (3.17)$$

Porém, Sugeno (1977) sugere a inserção de uma variável $\lambda > 0$, chamada de *complemento* λ . Nesta formulação o complemento de um conjunto *fuzzy* A é definido como:

$$\bar{A}_\lambda = C(A) = \frac{1 - \mu_A(x)}{1 - \lambda \mu_A(x)} \quad (3.18)$$

A variação desse parâmetro λ resulta em diferentes complementos de conjuntos *fuzzy*.

Para ilustrar os operadores descritos, considere o conjunto A formado por números *fuzzy* maiores que 0,5 e o conjunto B formado por números *fuzzy* aproximadamente iguais a 0,707, definidos no universo do discurso $U \in \mathbb{R}^+$ e representados da seguinte forma:

$$A = \{ [x, \mu_A(x)] \mid x \in U \} \quad (3.19a)$$

onde:

$$\mu_A(x) = \begin{cases} 0 & \text{para } x < 0,5 \\ \frac{1}{1 + (x - 0,5)^2} & \text{para } x > 0,5 \end{cases}$$

e:

$$B = \{ [x, \mu_B(x)] \mid x \in U \} \quad (3.19b)$$

onde:

$$\mu_B(x) = \frac{1}{1 + (x - 0,707)^{-2}} \quad \text{para } x > 0$$

A Figura 3.5 mostra a representação gráfica dos conjuntos A e B , a união e intersecção entre conjuntos A e B e o complemento do conjunto B , utilizando os operadores de Zadeh.

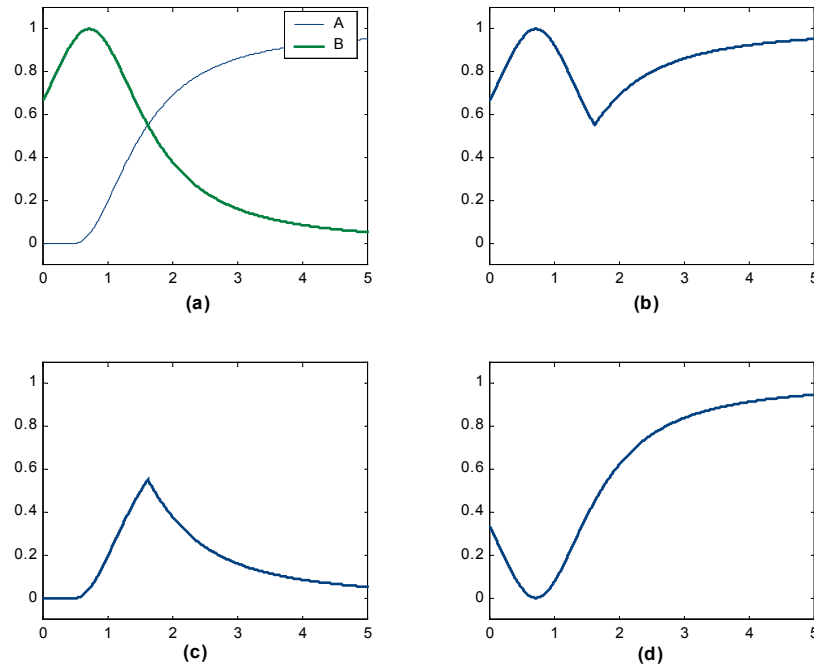


Figura 3.5 (a) Conjuntos *fuzzy* A e B ; (b) União entre os conjuntos *fuzzy* A e B ; (c) Intersecção entre os conjuntos *fuzzy* A e B e (d) Complemento do conjunto *fuzzy* B

Este exemplo demonstra que duas leis fundamentais da teoria dos conjuntos, a lei da contradição ($A \cup \bar{A} = U$) e a lei da exclusão intermediária ($A \cap \bar{A} = \emptyset$), não são válidas na teoria dos conjuntos *fuzzy*. Isto pode ser visto na Figuras 3.6 (a) e (d), onde para um mesmo ponto $x=0,5$ existe um grau de pertinência tanto para o conjunto B como para seu complemento: $\mu_B(0,5) \neq \mu_{\bar{B}}(0,5) \neq 0$, próprio da teoria dos conjuntos *fuzzy*.

3.2 Teoria da lógica *fuzzy* aplicada ao controle de sistemas

A lógica *fuzzy* é uma ferramenta matemática que permite operar com situações onde as fronteiras entre o verdadeiro ou falso são incertas. Isto é possível porque a lógica *fuzzy* faz uso dos conjuntos *fuzzy*. A partir da lógica *fuzzy* deriva-se o controle *fuzzy* que é uma técnica moderna de controle onde não é necessário o conhecimento preciso dos modelos matemáticos envolvidos. Em contrapartida é indispensável que se tenha um entendimento heurístico do sistema a ser controlado, conhecimento esse materializado em um conjunto de descrições lingüísticas entre conjuntos *fuzzy*. Através da descrição lingüística entre conjuntos *fuzzy*, transfere-se o conhecimento humano para um conjunto de regras *fuzzy*, do tipo:

SE um conjunto de condições é satisfeita, ENTÃO um conjunto de conseqüências pode ser inferida (3.20)

Descrições informais podem ser feitas através de variáveis lingüísticas *fuzzy* definidas por conjuntos *fuzzy*. As variáveis lingüísticas são portanto argumentos ou palavras (deslocamento, velocidade, controle, etc) que representam conjuntos *fuzzy* (positivo grande, negativo pequeno, zero, etc).

Assim, é possível enunciar um conjunto de regras que represente uma ação lógica global de todas as possibilidades e situações de um determinado problema. Em uma base de regras, denomina-se a parte do **SE** de antecedente da regra e a parte do **ENTÃO** de conseqüente da regra. Uma base com 3 regras pode assim ser enunciada:

	ANTECEDENTES	CONSEQÜENTES	
Regra 01:	SE o deslocamento é nulo e a velocidade é nula	ENTÃO o controle é nulo, SENÃO	(3.21a)
Regra 02:	SE o deslocamento é baixo e a velocidade é nula	ENTÃO o controle é baixo, SENÃO	(3.21b)
Regra 03:	SE o deslocamento é alto e a velocidade é pequena	ENTÃO o controle é alto.	(3.21c)

A descrição matemática da base de regra configura um algoritmo *fuzzy*. O algoritmo *fuzzy* utiliza a lógica *fuzzy* para definição dos significados de todos os conectivos (e, ou, se, então, senão, etc) presentes em uma base de regras.

3.2.1 Proposições *fuzzy* e conectivos lógicos

Um importante conceito em lógica *fuzzy* é a proposição *fuzzy*. Uma proposição *fuzzy* representa uma afirmação do tipo:

$$\text{"o deslocamento é baixo"} \quad (3.22)$$

onde “baixo” representa um rótulo lingüístico definido por um número *fuzzy* no universo do discurso da variável lingüística “deslocamento”.

A lógica *fuzzy* da negação de uma proposição *fuzzy* é definida similarmente à operação de complemento de um conjunto *fuzzy*. Um exemplo de uma proposição de negação *fuzzy* pode assim ser expressa:

$$\text{"o deslocamento não é baixo"} \quad (3.23)$$

E conforme definição do complemento proposta por Zadeh:

$$\mu_{\text{DESLOCAMENTO}_{\text{não é baixo}}} = 1 - \mu_{\text{DESLOCAMENTO}_{\text{baixo}}} \quad (3.24)$$

Tal como na lógica clássica, as proposições *fuzzy* podem ser combinadas pelo uso de conectivos lógicos “e” e “ou”, como por exemplo:

$$\text{O deslocamento é baixo “e” a velocidade é alta.} \quad (3.25)$$

Para combinar as preposições “e” e “ou”, a lógica *fuzzy* faz uso das *normas-T* e *normas-S*, respectivamente. Como já foi mostrado na Tabela 3.2 existem várias maneiras de aplicar estas normas.

Os operadores originais propostos por Zadeh, equações 3.26, têm a vantagem de não ignorar redundâncias, admitindo combinações entre duas proposições *fuzzy* iguais, representando uma mesma informação, o que não acontece com as propostas dos outros autores.

$$\mu_A \cup \mu_A = \max[\mu_A(x) ; \mu_A(x)] = \mu_A(x) \quad (3.26a)$$

$$\mu_A \cap \mu_A = \min[\mu_A(x) ; \mu_A(x)] = \mu_A(x) \quad (3.26b)$$

Entretanto, quando as proposições *fuzzy* não são iguais, porém correlacionadas, justifica-se o uso de outros operadores. Uma opção é o uso de operadores segundo as normas T e S de Lukasiewicz, equações 3.27, nos casos apropriados de completa interatividade.

$$\mu_A \cup \mu_B = \min[\mu_A(x) + \mu_B(x) ; 1] \quad (3.27a)$$

$$\mu_A \cap \mu_B = \max[\mu_A(x) + \mu_B(x) - 1 ; 0] \quad (3.27b)$$

Para exemplificar, suponha duas proposições representadas por:

$$\text{A velocidade é baixa} \quad \text{“ e ”} \quad \text{a velocidade é alta.} \quad (3.28)$$

Intuitivamente, espera-se que a combinação dessas duas proposições resulte num conjunto vazio, pois não é possível a velocidade ser alta e baixa simultaneamente. Segundo o formalismo até agora visto, o operador mais apropriado é o proposto por Lukasiewicz, como pode ser observado na Figura 3.6 (b).

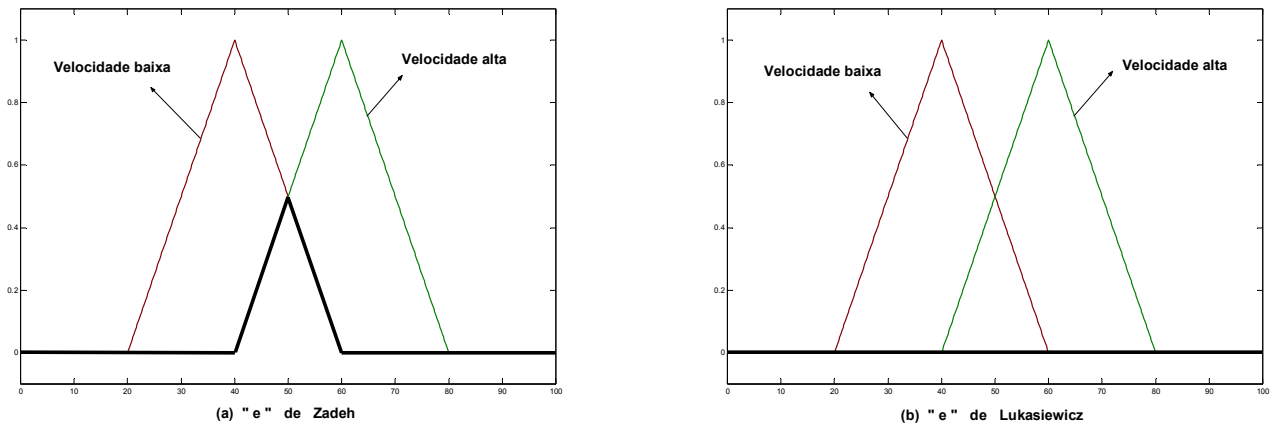


Figura 3.6 Representação gráfica dos operadores dos conectivos “e” segundo Zadeh e Lukasiewicz

Portanto a escolha das normas T e S , para representar os conectivos lógicos “e” e “ou” depende do contexto da relação entre as variáveis. Os operadores mais utilizados na lógica *fuzzy* são os mostrados na Tabela 3.3.

Tabela 3.3: Operadores comumente utilizados para os conectivos “e” e “ou” na lógica *fuzzy*

“e”	“ou”	Referência
$\min(a,b)$	$\max(a,b)$	Zadeh
$\max(a+b-1,0)$	$\min(a+b,1)$	Lukasiewicz
ab	$a+b-ab$	Probabilidade

3.2.2 Métodos de inferência da base de regras *fuzzy*

Nas descrições matemáticas elementares, uma entrada x pode ser inferida em uma função $y=f(x)$ para obtenção de y . Similarmente, em descrições lingüísticas *fuzzy*, uma entrada pode ser inferida para a obtenção da saída. O processo de avaliação de uma descrição lingüística, contida em uma base de regras, é chamado de inferência composicional *fuzzy*.

Existem dois tipos de inferência composicional *fuzzy*: *generalizado modus ponens* e *generalizado modus tolens* (Urih e Tsoukalas, 1997). Normalmente, a inferência composicional da base de regras *fuzzy* consiste em, num primeiro instante, avaliar as regras através de uma relação chamada de implicação da regra, e depois agregar todas elas.

Cada regra é representada por uma afirmação *se-então*, onde o antecedente e o conseqüente consistem de proposições combinadas pelos conectivos lógicos “e” e “ou”.

Considere a seguinte regra:

$$\text{Se } x_1 \text{ é } A_1 \text{ e } x_2 \text{ é } A_2 \text{ então } y \text{ é } B \quad (3.29)$$

onde A_1 , A_2 e B são conjuntos *fuzzy*. A seguinte relação *fuzzy* pode ser construída para representar essa regra:

$$R = \Phi (T (A_1, A_2), B) \quad (3.30)$$

onde T é uma *norma-T* e Φ é a função de implicação *fuzzy*. Assim como a *norma-T* modela o conectivo “e”, a implicação Φ modela a sentença “se-então”.

A regra *fuzzy* pode também ser representada através das funções de pertinência envolvidas:

$$\mu_R (x_1, x_2, y) = \Phi \{ T [\mu_{A_1}(x_1), \mu_{A_2}(x_2)]; \mu_B (y) \} \quad (3.31)$$

As funções de implicação *fuzzy* são normalmente denotadas por $\Phi(a,b)$ quando $a,b \in [0,1]$ e serão descritas com mais detalhes na próxima seção.

3.2.3 Implicações de regras *fuzzy*

Existem diversas maneiras de representar uma implicação *fuzzy*. Normalmente a implicação significa uma relação de casualidade, ditada pela sentença *se-então*, onde o sentido da proposição ou sentença tem que ser mantida para representar o conhecimento a respeito do sistema. Dubois e Prade (1991) relatam diferentes tipos de implicações *fuzzy*:

(i) Implicação clássica ou implicação-S: Define-se $a \rightarrow b$ a aplicação de alguma *norma*-S entre o complemento de a em intersecção com b :

$$\Phi(a,b) = S(C(a); b) \quad (3.32)$$

(ii) Implicação baseada na Lógica Quântica: $a \rightarrow b$ é definido como o complemento de a em intersecção com a união de a com b :

$$\Phi(a,b) = S(C(a); T(a,b)) \quad (3.33)$$

onde *norma*-T neste caso adota um critério de dualidade complementar, definido por:

$$T(a,b) = C(S(C(a), C(b))) \quad (3.34)$$

(iii) Implicação *fuzzy* utilizando somente a *norma*-T:

$$\Phi(a,b) = T(a,b) \quad (3.35)$$

(iv) Implicação residual *fuzzy* representa uma proposição parcial, na forma:

$$\Phi(a,b) = \begin{cases} 1, & \text{se } a \leq b \\ 0, & \text{se } a = 1 \text{ e } b = 0 \\ \in [0,1], & \text{caso contrário.} \end{cases} \quad (3.36)$$

A implicação (iii), que somente utiliza a *norma-T*, pode ser considerada como uma generalização da implicação clássica e é muito utilizada em problemas de controle de sistemas.

Lee (1990a,b) reporta mais de 40 maneiras distintas para a representação da função de implicação *fuzzy*. As implicações listadas na tabela 3.4, são freqüentemente encontradas em aplicações de engenharia, particularmente, no projeto de controladores *fuzzy*.

Tabela 3.4 Implicações *fuzzy* entre **a** e **b**

$\Phi(a,b)$	Referências
$\min[1-a; \min(a,b)]$	Zadeh (1975)
$\min[1-a+b; 1]$	Lukasiewicz
$\max[1-a; b]$	Kleene (1938)
$1-a+ab$	Reichenbach
b , se $a > b$ ou : 1 , caso contrário	Gödel
1 , se $a = 0$ ou : $\min[b \cdot a^{-1}; 1]$, caso contrário	Goguen (1969)
1 , se $a < b$ ou : 0 , caso contrário	Gaines (1976)
1 , se $a < b$ ou : $\min[1-a; b]$, caso contrário	Wu (1986)
b^a	Yager (1980a)
$\min\{\max(1-a;b); \max[a,1-b]; \min(b,1-a)\}$	Willmott (1980)
$\min(a,b)$	Mamdani (1974)
ab	Larsen (1980)

3.2.4 Agregação das regras *fuzzy*

Na seção anterior mostrou-se como compor uma sentença (se-então) por uma implicação *fuzzy*. O algoritmo *fuzzy* envolve, normalmente, mais de uma proposição *fuzzy*, composta por antecedentes e conseqüentes, formando uma base de regras da seguinte forma:

$$\begin{aligned}
 r_1 &: \text{se } x_1 \text{ é } A_{1,1} \quad \text{e } \dots \text{ e } \quad x_{N_x} \text{ é } A_{N_x,1} \quad \text{então} \quad y = B_1; \text{ senão} \\
 r_2 &: \text{se } x_1 \text{ é } A_{1,2} \quad \text{e } \dots \text{ e } \quad x_{N_x} \text{ é } A_{N_x,2} \quad \text{então} \quad y = B_2; \text{ senão } \dots \\
 r_{Nr} &: \text{se } x_1 \text{ é } A_{1,Nr} \quad \text{e } \dots \text{ e } \quad x_{N_x} \text{ é } A_{N_x,Nr} \quad \text{então} \quad y = B_{Nr}.
 \end{aligned} \tag{3.37}$$

onde N_r é o número de regras paralelas e N_x é o número de variáveis contidas nos antecedentes de todas as regras.

A combinação das regras *fuzzy* é feita em uma relação *fuzzy* que é chamada de agregação. É possível agregar as regras de diferentes maneiras. Para a implicação Mamdani, a agregação (senão) será adotada como sendo o operador de união das regras, que faz uso das *normas-S*, conforme a equação 3.38, onde a agregação (A) é a união de todas as N_r regras *fuzzy*, cada qual com uma relação de implicação R_k .

$$A = \bigcup_k R_k \quad (3.38)$$

De forma análoga, a agregação poderá também assumir o operador intersecção, não fazendo mais uso das *normas-S*, e sim das *normas-T*, como a equação 3.39:

$$A = \bigcap_k R_k \quad (3.39)$$

São inúmeras as possibilidades de escolha do operador agregação, uma vez que a agregação faz uso das *normas-S* e das *normas-T*.

Quando se utilizam relações de implicação Mamdani e Larsen, com inferência *generalizado modus ponens*, o conseqüente da regra resulta em um subconjunto *fuzzy* da saída. A agregação, nestes casos, poderá ser a união ou a intersecção dos subconjuntos dos conseqüentes de cada regra, resultando em um novo conjunto *fuzzy*. Serão mostrados a seguir exemplos gráficos ilustrando implicações e agregações de regras, quando submetidas a um algoritmo *fuzzy* de controle.

3.2.5 Propriedades das regras *fuzzy*

São apresentadas nesta seção as três propriedades principais que caracterizam a base de regras. Suponha, por exemplo, o caso de duas variáveis linguísticas nas entradas, erro e variação do erro (Δerro), e uma variável linguística na saída, o controle. Cada variável linguística é representada por cinco conjuntos *fuzzy* dos número NG (negativo grande), NP (negativo pequeno), Z (zero), PP (positivo negativo), PG (positivo grande). A base de regras podem ser:

(i) Base de regras completa: Essa propriedade distingue uma base de regras completa de outra não completa. Um conjunto de regras é completo se todas as combinações dos valores das entradas resultarem em valores de saída. Formalmente, tem-se que:

$$\forall erro, \Delta erro : hgt[controle(erro, \Delta erro)] > 0 \quad (3.40)$$

onde *hgt* é a altura do conjunto *fuzzy* da saída diante de uma entrada para o erro e outra para o $\Delta erro$. A Figura 3.7 ilustra uma base de regras incompleta onde as entradas para o erro e para o $\Delta erro$ não existem em algumas células e portanto não produz nenhuma saída ou ação de controle. Por exemplo, a seguinte regra:

$$\text{Se } erro \text{ é PG } \text{ e } \Delta erro \text{ é PG } \text{ então } controle = ? \quad (3.41)$$

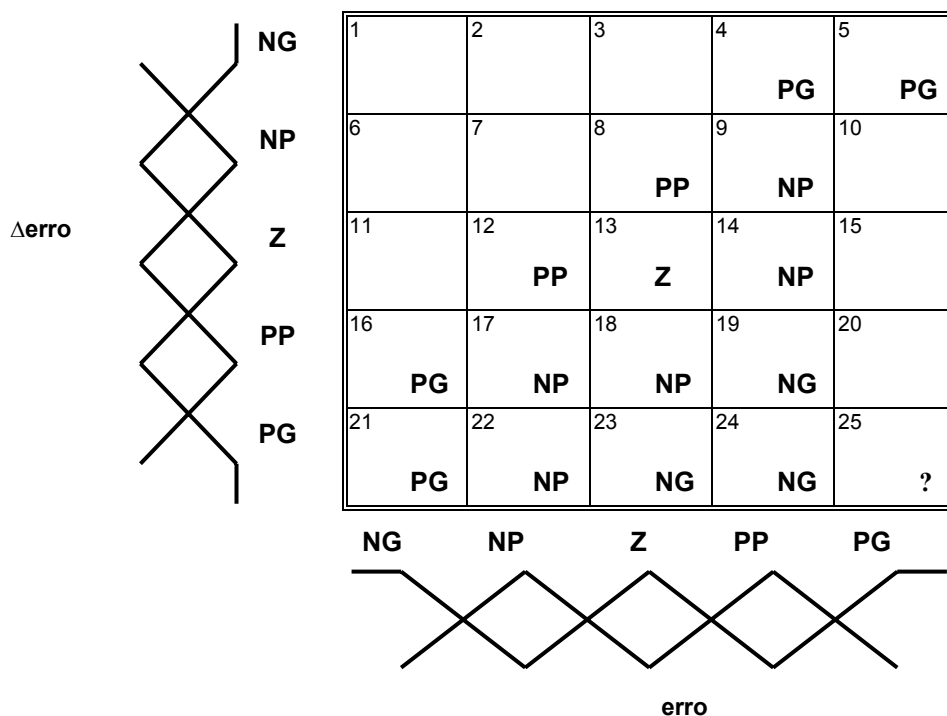


Figura 3.7 Exemplo de uma base de regra incompleta e descontínua, com 25 regras, cada uma representada por uma célula

(ii) Base de regras contínua: Para definição de uma base de regras contínua é necessário definir células ou regras vizinhas. Considere, a partir da Figura 3.9, a regra 17. Uma regra é vizinha a outra caso ocupe uma das posições adjacentes, no caso da regra 17, as regras com posições 12, 16, 18, 22. Define-se uma base de regras contínua quando não existe nenhuma

regra vizinha que apresente o conjunto de saída com interseção vazia. Por exemplo a regra 22 é NP e a regra vizinha 21 tem saída PG. Os conjuntos PG e NG da saída são definidos, normalmente, nos extremos do universo da saída, logo a interseção entre eles é vazia, concluindo-se que esta base é descontínua.

(iii) Base de regra consistente: A consistência de uma regra está relacionada com a capacidade de representação do conhecimento do sistema, ou seja, a construção adequada da base de regras. Seja a base de regras:

$$R_1: \text{Se erro é NP ou } \Delta\text{erro é Z então controle é PP} \quad (3.42a)$$

$$R_2: \text{Se erro é PP ou } \Delta\text{erro é PP então controle é Z} \quad (3.42b)$$

$$R_3: \text{Se erro é Z ou } \Delta\text{erro é Z então controle é NG} \quad (3.42c)$$

Estas regras prevêem conclusões ambíguas sobre controle, conforme mostra a Tabela 3.5, pois o controle poderá assumir dois valores PP e Z, tanto quando o erro for NP e Δ erro for PP como quando erro for PP e Δ erro for Z.

Tabela 3.5 : Exemplo de uma base de regras inconsistente.

Δ erro \ erro	NP	Z	PP
NP	PP	NG	Z
Z	PP	PP	PP,Z
PP	PP,Z	Z	Z

Quando a base de regras é mais complexa este fenômeno é de difícil identificação. Isto também ocorre com o uso operador complemento, por exemplo, a proposição: o erro não é negativo grande, ou também quando o controlador envolve múltiplas entradas e múltiplas saídas.

3.2.6 Algoritmo *fuzzy* de controle

Basicamente o algoritmo *fuzzy* de controle é uma aplicação de uma inferência composicional de regras. Dado que, R representa as regras do controlador e A' representa um espaço de entrada *fuzzy*, a saída *fuzzy* B' pode ser obtida pela seguinte relação entre A' e R :

$$B' = R \circ A' \quad (3.43)$$

onde o símbolo “ $\circ$ ” representa o método de inferência composicional utilizada no algoritmo *fuzzy*.

Entretanto, as entradas e saídas de controladores são normalmente valores numéricos. Portanto, há necessidade de transformar as entradas numéricas em entradas *fuzzy*, assim como transformar as saídas *fuzzy* em saídas numéricas, transformações estas que são chamadas de *fuzzificação* e *defuzzificação*, respectivamente.

O projeto do controlador *fuzzy* tem a finalidade de mapear os dados de entrada em dados de saída através da inferência composicional da base de regras. Para tanto são necessárias três etapas distintas: (a) a *fuzzificação* das entradas; (b) a aplicação de um mecanismo composicional de inferência de regras e (c) a *defuzzificação* das saídas.

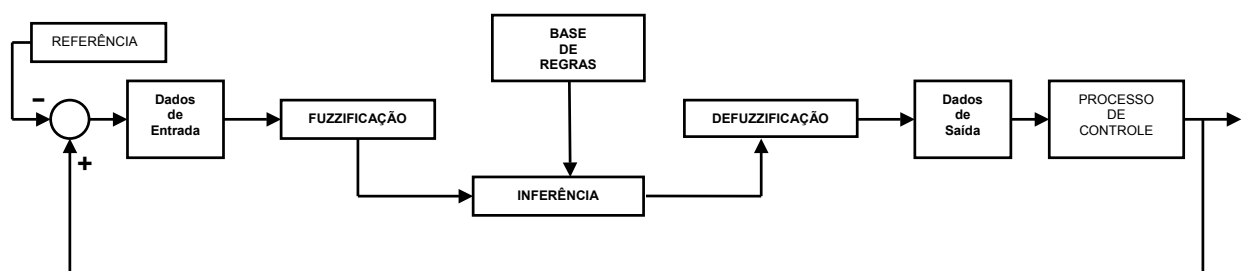


Figura 3.8 Representação de um controlador *fuzzy* com seus elementos básicos

Estas etapas interligadas configuram o algoritmo *fuzzy* completo, conforme a Figura 3.8. Para o entendimento do algoritmo, desde a entrada de dados até a saída do controlador, faz-se necessário a descrição de cada uma das etapas de um controlador:

(a) Etapa de *fuzzificação* das entradas

A etapa de *fuzzificação* de um controlador consiste em transformar um conjunto de dados numéricos de entrada, medidos ou estimados, em valores de pertinência de um conjunto *fuzzy*.

Considere um problema de duas entradas erro e Δ erro, cada qual descrita dentro dos seus respectivos universos do discurso por três funções de pertinências triangulares, denominadas negativo (N), zero (Z) e positivo (P). Sejam apenas duas regras com seus antecedentes e suas proposições:

$$R_1 : \text{ Se erro é N e } \Delta\text{erro é P então ...} \quad (3.44a)$$

$$R_2 : \text{ Se erro é Z e } \Delta\text{erro é P então ...} \quad (3.44b)$$

A *fuzzificação* é o cálculo dos antecedentes das duas regras.

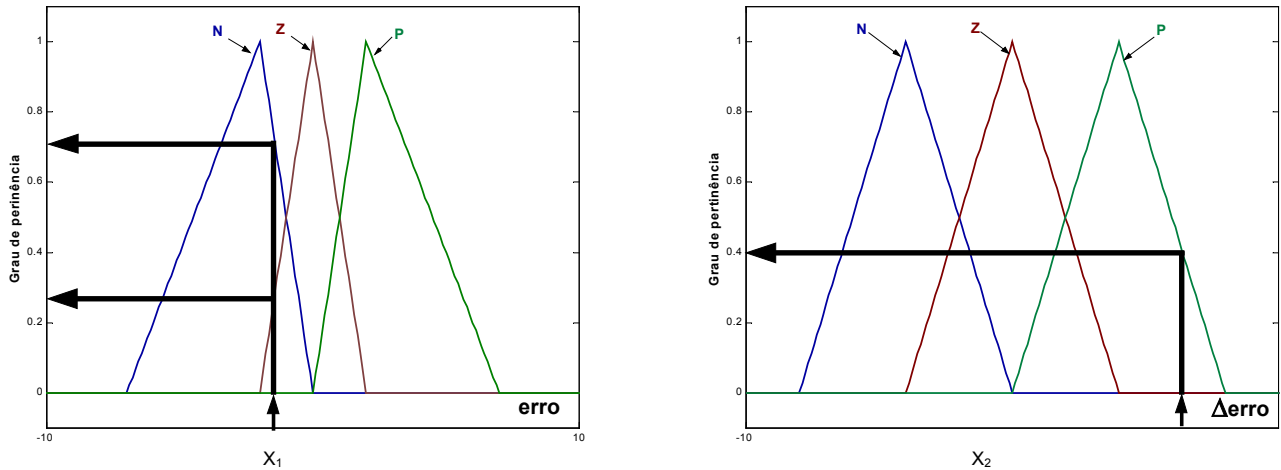


Figura 3.9 : Representação gráfica da *fuzzificação* de duas entradas x_1 e x_2

Suponha que o valor do erro é x_1 e da variação do erro é x_2 , a *fuzzificação* das entradas x_1 e x_2 quando aplicada na primeira regra, consiste no prolongamento vertical da entrada x_1 até o conjunto relativo à proposição relativa ao erro, o conjunto N. Calcula-se o valor de pertinência do ponto de cruzamento entre o prolongamento da entrada x_1 e o conjunto N envolvido na regra. Da mesma forma a segunda proposição relativa ao Δerro , a entrada x_2 é prolongada verticalmente até o conjunto *fuzzy* P. Novamente o ponto de cruzamento da prolongação da entrada x_2 com o conjunto envolvido na regra, o P, resulta em um valor de pertinência para o Δerro . De acordo com a Figura 3.9, tem-se:

$$R_1 : \text{ Se erro} = x_1 \text{ é N e } \Delta\text{erro} = x_2 \text{ é P então ...} \quad (3.45a)$$

$$R_1 : \text{ Se } \mu_N(x_1) = 0,7 \text{ e } \mu_P(x_2) = 0,4 \text{ então ...} \quad (3.45b)$$

Analogamente, o resultado da *fuzzificação* da segunda regra, para as mesmas entradas x_1 e x_2 é:

$$R_2 : \text{ Se erro é Z e } \Delta\text{erro é P então ...} \quad (3.46a)$$

$$R_2 : \text{ Se } \mu_P(x_1) = 0,25 \text{ e } \mu_Z(x_2) = 0,4 \text{ então ...} \quad (3.46b)$$

Assim, a etapa de *fuzzificação* das entradas é o cálculo dos antecedentes de todas as regras, através da conversão de valores numéricos em valores de pertinência do conjunto envolvido nas preposições das regras.

(b) Etapa de aplicação do mecanismo composicional de inferência de regras

Os mecanismos composicionais mais comuns de inferência de regras utilizando GMP, aplicados na construção de controladores *fuzzy*, são:

(i) Método composicional de inferência *max-min*: Desde a primeira concepção do controlador *fuzzy*, este método tem sido utilizado com bastante êxito. Ele consiste em escolher o operador *min* (Mandani) para as implicações *fuzzy* e o operador *max* (união) para a agregação.

Considere por exemplo, que um controlador possua duas entradas e uma saída, e a seguinte base de regras:

$$\mathbf{R_1 : \quad Se \quad } x_1 \text{ é P } \text{ e } x_2 \text{ é Z } \quad \mathbf{então \quad a saída \quad } u_1 = \text{N} \quad (3.47a)$$

$$\mathbf{R_2 : \quad Se \quad } x_1 \text{ é Z } \text{ e } x_2 \text{ é N } \quad \mathbf{então \quad a saída \quad } u_1 = \text{Z} \quad (3.47b)$$

$$\mathbf{R_3 : \quad Se \quad } x_1 \text{ é N } \text{ e } x_2 \text{ é P } \quad \mathbf{então \quad a saída \quad } u_1 = \text{P} \quad (3.47c)$$

Suponha que num dado momento seja necessário avaliar a saída para uma entrada x_1 e x_2 . É feita inicialmente a *fuzzificação* de todas proposições de cada antecedente. Como as implicações são do tipo Mandani, para modelar o conectivo “e” utiliza-se o mínimo dos graus de pertinência envolvidos. O resultado da inferência destas três regras é obtido de acordo com a Figura 3.10:

$$\mathbf{R_1 : \quad Se \quad } \mu_P(x_1) = 0,75 \text{ e } \mu_Z(x_2) = 0,35 \quad \mathbf{então \quad a saída \quad } u_1 \text{ é } 0,35 \text{ de } \mu_N(u_1) \quad (3.48a)$$

$$\mathbf{R_2 : \quad Se \quad } \mu_Z(x_1) = 0,6 \text{ e } \mu_N(x_2) = 0,3 \quad \mathbf{então \quad saída \quad } u_1 \text{ é } 0,30 \text{ de } \mu_Z(u_1) \quad (3.48b)$$

$$\mathbf{R_3 : \quad Se \quad } \mu_N(x_1) = 0,0 \text{ e } \mu_P(x_2) = 0,7 \quad \mathbf{então \quad a saída \quad } u_1 \text{ é } 0,0 \text{ de } \mu_P(u_1) \quad (3.48c)$$

Por exemplo, a implicação da primeira regra resulta em um subconjunto $\mu_{N'}(u_1)$ para a saída, truncado na altura igual à mínima pertinência das preposições do antecedente, no caso 0,35. Assim, o procedimento é repetido para todas as regras.

A Figura 3.11 ilustra a agregação de todos os conseqüentes, ou seja, a união dos subconjuntos obtidos em cada regra pelo método de inferência composicional *max-min*.

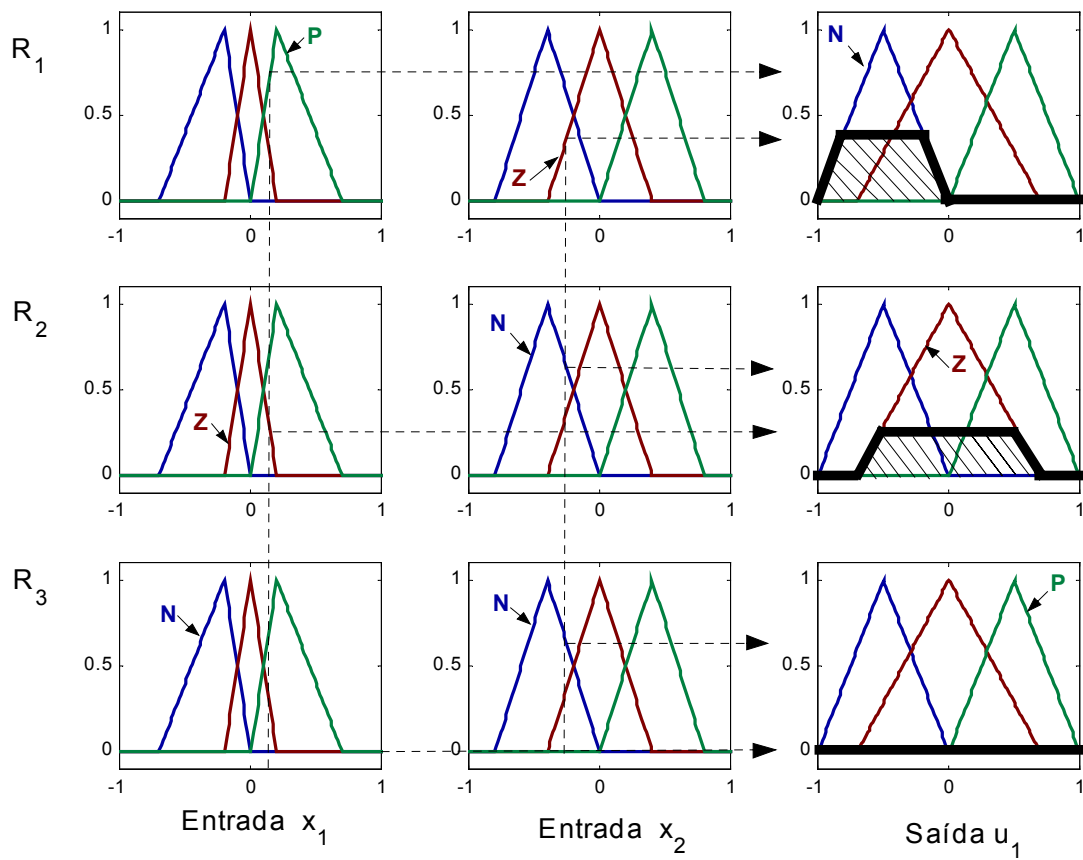


Figura 3.10 : Cálculo dos antecedentes e dos consequentes de cada regra pelo método de inferência composicional *max-min*

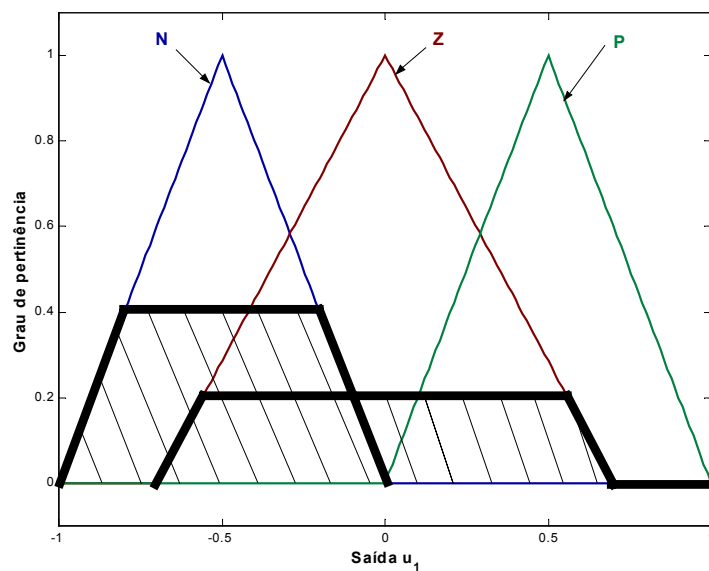


Figura 3.11 : Agregação dos consequentes das três regras pelo método de inferência composicional *max-min*.

(ii) Método composicional de inferência *max-prod*: Esse método composicional de inferência representado pelo produto foi proposto por Larsen (1980). Tão comumente utilizado, o *max-prod* é um método que emprega a implicação de Larsen para escalonar o conseqüente por um certo grau de pertinência, resultando em um subconjunto envolvido no conseqüente da regra. Esses subconjuntos de cada regra são agregados, pelo operador união, para obtenção de um único conjunto *fuzzy*.

Suponha novamente que o controlador possua duas entradas e uma saída, e que para obter a lei de controle são necessárias as mesmas três regras aqui reescritas:

$$R_1 : \text{ Se } x_1 \text{ é P e } x_2 \text{ é Z então a saída } u_1 \text{ é N} \quad (3.49a)$$

$$R_2 : \text{ Se } x_1 \text{ é Z e } x_2 \text{ é N então a saída } u_1 \text{ é Z} \quad (3.49b)$$

$$R_3 : \text{ Se } x_1 \text{ é N e } x_2 \text{ é P então a saída } u_1 \text{ é P} \quad (3.49c)$$

Suponha que se deseja avaliar a saída para uma entrada x_1 e x_2 . Inicialmente é feita a *fuzzificação* de todas proposições de cada antecedente. Como as implicações são do tipo Larsen, para o modelar o “e” utiliza-se o produto dos graus de pertinências envolvidos. O resultado da inferência destas três regras é obtido de acordo com a Figura 3.12:

$$R_1 : \text{ Se } \mu_P(x_1) \text{ é } 0,75 \text{ e } \mu_Z(x_2) \text{ é } 0,35 \text{ então a saída é } 0,2625 \cdot \mu_N(u_1) \quad (3.50a)$$

$$R_2 : \text{ Se } \mu_Z(x_1) \text{ é } 0,60 \text{ e } \mu_N(x_2) \text{ é } 0,30 \text{ então a saída é } 0,180 \cdot \mu_Z(u_1) \quad (3.50b)$$

$$R_3 : \text{ Se } \mu_N(x_1) \text{ é } 0,00 \text{ e } \mu_P(x_2) \text{ é } 0,70 \text{ então a saída é } 0,0 \cdot \mu_P(u_1) \quad (3.50c)$$

Por exemplo, a implicação da primeira regra resulta em um subconjunto N para a saída, escalonado pelo produto dos valores de pertinência das proposições da primeira regra, no caso $0,75 \cdot 0,35 = 0,2625$, ou seja, o conjunto de saída *fuzzy* N é multiplicado por 0,2625 para formar o subconjunto representando o conseqüente da regra. Assim se sucede para todas as regras.

A Figura 3.13 ilustra a agregação de todos os conseqüentes, ou seja, união dos subconjuntos obtidos em cada regra pelo método de inferência composicional *max-prod*.

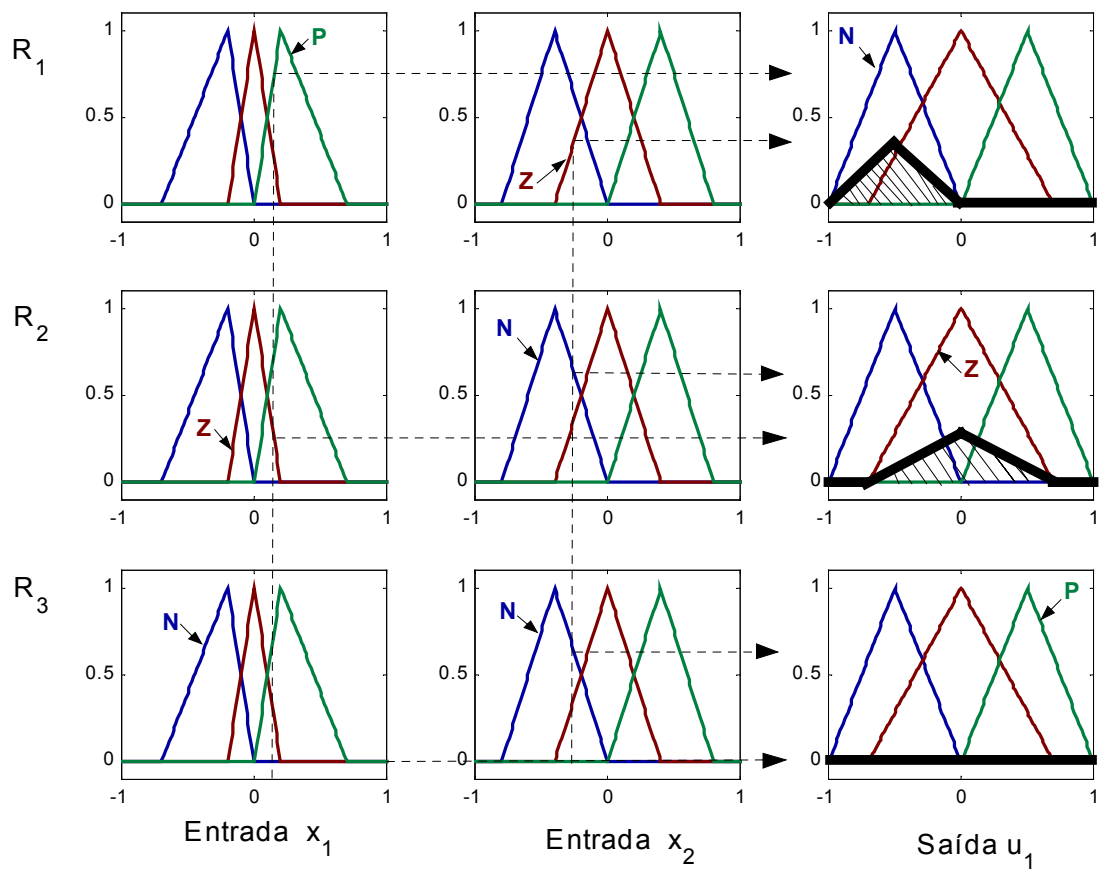


Figura 3.12 : Cálculo do antecedente e do conseqüente de cada regra pelo método de composicional de inferência *max-prod*

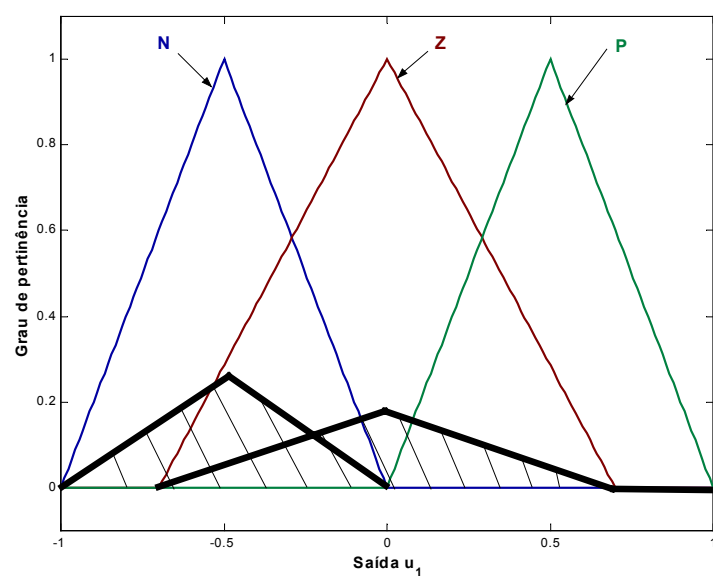


Figura 3.13 : Agregação dos conseqüentes das três regras pelo método de inferência composicional *max-prod*.

(c) Etapa de *defuzzificação* das saídas

Após a inferência das entradas, resta ao algoritmo de controle fazer a *defuzzificação*, isto é, transformar o conjunto *fuzzy* obtido na agregação das regras em saídas numéricas. Muitos autores (Terano *et al.*, 1992; Pedrycz, 1993; Yager e Filev, 1993) têm sugerido diferentes métodos para o cálculo da *defuzzificação*. A escolha pelo método tem um impacto significativo na velocidade e acuracidade do controlador *fuzzy*. Os métodos mais comumente empregados são:

(i) Método do centro de gravidade (CG): Este método consiste na técnica do cálculo do centróide de figuras planas. O cálculo é feito baseado na região da figura resultante da agregação de todas as regras, tais como as Figuras 3.13 e 3.15. O método do centro de gravidade é definido como:

$$u_{CG} = \frac{\sum_{i=1}^N u_i \cdot \mu_u(u_i)}{\sum_{i=1}^N \mu_u(u_i)} \quad (3.51)$$

onde o somatório é tomado sob os valores discretos do universo do discurso de u_i amostrados em N pontos, $\mu_u(u_i)$ é o valor da pertinência para um dado valor de controle u_i . O valor de controle u_{CG} tem a propriedade de dividir a figura resultante da agregação em duas áreas iguais, conforme mostra a Figura 3.14.

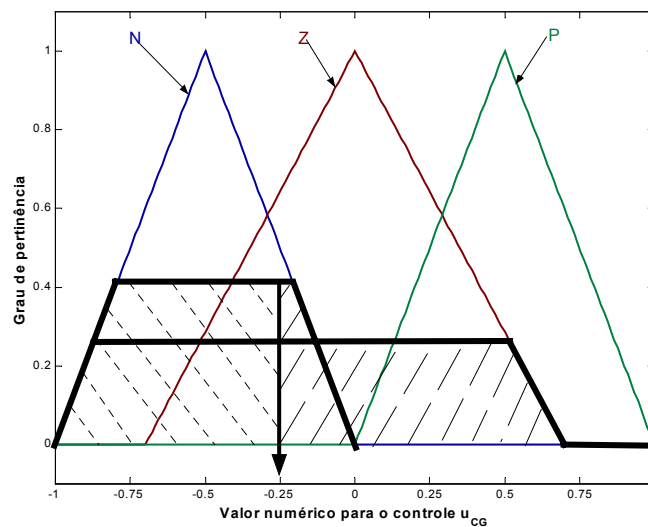


Figura 3.14 Representação Gráfica da *defuzzificação* pelo métodos do centro de gravidade

(ii) Método do centro de área (CA): Este método considera a área obtida do conseqüente de cada regra. O método de *defuzzificação* pelo centro da área é definido por:

$$u_{CA} = \frac{\sum_{i=1}^N u_i \cdot \sum_{k=1}^n \mu_{Bk}(u_i)}{\sum_{i=1}^N \sum_{k=1}^n \mu_{Bk}(u_i)} \quad (3.52)$$

onde valores discretos do universo do discurso de u_i amostrados em N pontos para todas n regras são somados, $\mu_{Bk}(u_i)$ é o valor da pertinência para um dado valor de controle u_i do conseqüente da k -ésima regra. Por exemplo, na Figura 3.15 o valor de controle u_{CA} é obtido somando-se os valores para os dois trapézios resultantes da agregação. Neste método o resultado é modificado quando existem regiões comuns agregadas por duas ou mais regras, como mostra a área hachurada da Figura 3.15, utilizada no cálculo do centro de área duas vezes.

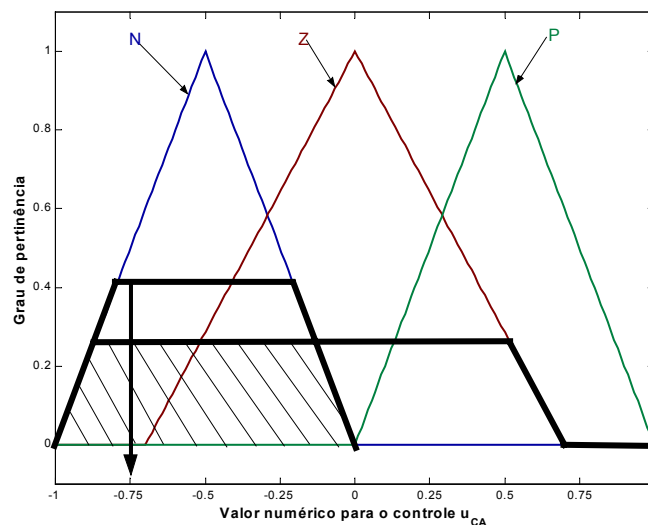


Figura 3.15 Representação gráfica da *defuzzificação* pelo método do centro de área

(iii) Método de *defuzzificação* pelo cálculo da média do máximo (MM): Neste método, o cálculo da *defuzzificação* consiste em atribuir ao valor u_{MM} de controle o maior valor de pertinência da figura resultante da agregação das regras.

Muitas vezes o máximo valor ocorre em mais de um ponto do universo do discurso, como mostra a Figura 3.16. Nestes casos o método do centro de gravidade considera a média de todos valores de u_i que possuem função de pertinência máxima, resultante da agregação de todas as regras.

Este método de *defuzzificação* pela média do máximo é definido por:

$$u_{MM} = \sum_{i=1}^M \frac{u_m}{M} \quad (3.53)$$

onde o somatório é tomado sob os valores discretos do universo do discurso de u_i amostrados em M pontos, u_m são as maiores alturas *hgt* que existem no conjunto resultante da agregação.

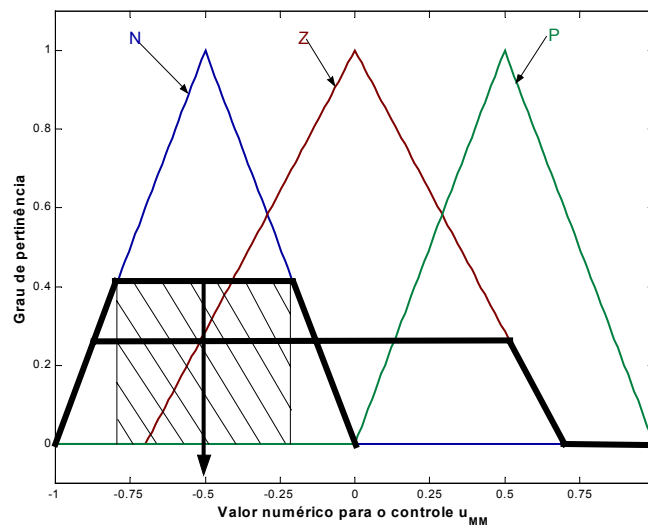


Figura 3.16 Representação gráfica da *defuzzificação* pelo método da média do máximo

Percebe-se que os métodos do centro de gravidade e média do máximo dificilmente atinge os extremos do universo do discurso da variável de saída.

3.2.7 Regras *fuzzy* para controle

Dois tipos de regras são comumente utilizadas em algoritmos *fuzzy* para controladores: regras do tipo Mamdani e Sugeno. As regras adotadas nesta dissertação são do tipo Mamdani.

(i) Regras *fuzzy* do tipo Mamdani: Este tipo de regra foi usado nas primeiras aplicações dos controladores *fuzzy* (Mamdani,1974; Assilian,1974; Mamdani e Assilian,1975), e possui a seguinte forma geral:

$$r_k : \text{Se } x_1 \text{ é } A_1^k \text{ e } \dots \text{ e } x_{Nx} \text{ é } A_{Nx}^k \text{ então } y_1 \text{ é } B_1^k \text{ e } \dots \text{ e } y_{Ny} \text{ é } B_{Ny}^k \quad (3.54)$$

onde k é o número de regras, N_x é o número de variáveis contidas nos antecedentes de todas as regras, N_y é o número de variáveis contidas nos conseqüentes de todas as regras.

Uma regra do tipo Mamdani para o exemplo em que se tem duas entradas, o *erro* e a mudança do erro Δerro , e somente uma saída de controle u , pode ser assim escrita:

$$\text{Se erro é Positivo Grande e } \Delta\text{erro é Negativo Grande então } u \text{ é Positivo Médio} \quad (3.55)$$

Normalmente uma regra Mamdani utiliza o operador *min* para a implicação entre as proposições das regras, e o operador *max* para a agregação, isto é, a união dos subconjuntos *fuzzy* de cada conseqüente.

Ao considerar um controlador *fuzzy* com uma única entrada e uma saída, o resultado do controlador é limitado aos pontos característicos entre a entrada e a saída. Quando o controlador possui múltiplas entradas e múltiplas saídas, o controlador torna-se uma superfície multidimensional. No caso de duas entradas e uma saída, o controlador fica restrito a uma superfície de controle tridimensional, que depende do número de regras envolvidas no projeto do controlador, como pode ser observado na Figura 3.17.

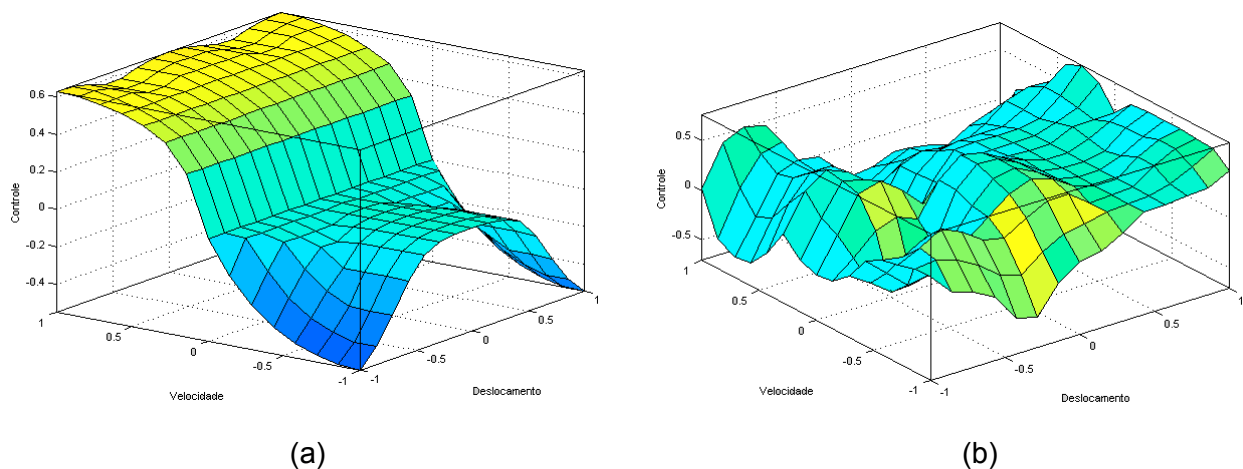


Figura 3.17 (a) Superfície de controle obtidas a partir de 6 regras e (b) Superfície de controle obtidas a partir de 50 regras. Ambas com controlador de duas entradas e uma saída.

Jager *et al.* (1991) propuseram uma outra possibilidade de construção de regras, multiplicando os conseqüentes por pesos, por exemplo:

$$\text{Se erro é Positivo Pequeno e } \Delta\text{erro é Positivo Pequeno então o controle } u \text{ é 90\% Zero e 10\% Positivo} \quad (3.56)$$

(ii) Regras fuzzy do tipo Sugeno:

As regras do tipo Sugeno (Takagi e Sugeno, 1983), tem a seguinte forma geral:

$$r_1: \text{ Se } x_1 \text{ é } A_{1,k} \text{ e } \dots \text{ e } x_{N_x} \text{ é } A_{N_x,k} \text{ então } u_1 = f_{1,k}(x_1, \dots, x_{N_x}), \dots f_{N_x,k}(x_1, \dots, x_{N_x}) \quad (3.57a)$$

⋮

$$r_k: \text{ Se } x_1 \text{ é } A_{1,k} \text{ e } \dots \text{ e } x_{N_x} \text{ é } A_{N_x,k} \text{ então } u_k = f_{1,k}(x_1, \dots, x_{N_x}), \dots f_{N_x,k}(x_1, \dots, x_{N_x}) \quad (3.57b)$$

onde $f_{N_x,k}(x_1, \dots, x_{N_x})$ é uma função que modela uma combinação linear das entradas, ou seja:

$$r_k: \text{ Se } x_1 \text{ é } A_{1,k} \text{ e } \dots \text{ e } x_{N_x} \text{ é } A_{N_x,k} \text{ então } u_k = b_{0,k} + \sum_{i=1}^{N_x} b_{i,k} x_i \quad (3.58)$$

onde $b_{i,k}$ e $b_{0,k}$ são parâmetros constantes.

Neste caso, a inferência segue a seguinte metodologia. Inicialmente, as entradas x_1 , x_2 , ..., x_{N_x} , referentes a um dado estado do sistema a ser controlado, são inferidas para cada proposição dos antecedentes das regras. Então os graus de pertinência das proposições de cada regra é obtido. Para cada regra escolhe-se o valor mínimo das funções de pertinências inferidas, ou seja, a primeira regra tem uma pertinência resultante $\mu_{u1}(x_1 \dots x_{N_x})$ e assim por diante até a última regra com pertinência resultante $\mu_{uk}(x_1 \dots x_{N_x})$. Logo, quando a base de regras é Sugeno, não é necessário a agregação dos conjuntos resultantes das implicações de cada regra. O resultado da *defuzzificação* é obtido da seguinte forma:

$$u_{Sugeno} = \frac{\sum_{i=1}^k \mu_{u_i} . u_i}{\sum_{i=1}^k \mu_{u_i}} \quad (3.59)$$

Para exemplificar, suponha que um controlador possua duas entradas e uma saída, e que a lei de controle seja obtida a partir do conjunto de regras:

$$R_1: \text{ Se } x_1 \text{ é N e } x_2 \text{ é P então a saída } u_{R1} = x_1 - 5 . x_2 \quad (3.60a)$$

$$R_2: \text{ Se } x_1 \text{ é N e } x_2 \text{ é Z então a saída } u_{R2} = 2 . x_1 - 4 . x_2 \quad (3.60b)$$

$$R_3: \text{ Se } x_1 \text{ é Z e } x_2 \text{ é Z então a saída } u_{R3} = x_1 + 2 . x_2 - 1 \quad (3.60c)$$

Deseja-se avaliar a saída para uma entrada $x_1 = -0,18$ e $x_2 = 0,40$. É feita inicialmente a *fuzzificação* de todas proposições de cada antecedente, de acordo com a Figura 3.18.

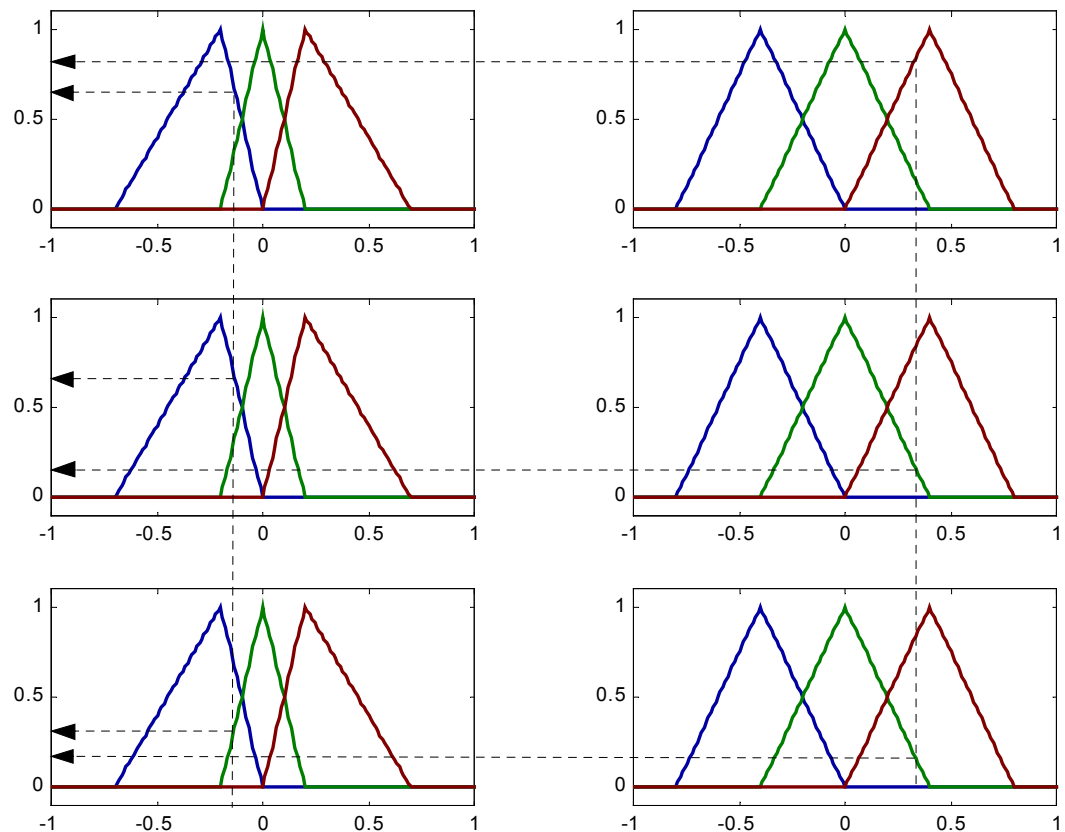


Figura 3.18 Procedimento de *fuzzificação* dos antecedentes das regras do tipo Sugeno.

Podem ser escolhidas implicações do tipo Mamdani ou do tipo Larsen, porém neste exemplo é adotado Mamdani. Então para modelar o “e” entre as proposições utiliza-se o mínimo dos graus de pertinência envolvidos. O resultado da inferência das três regras é o seguinte:

$$R_1 : \text{ Se } \mu_N(x_1) \text{ é } \mathbf{0,70} \text{ e } \mu_Z(x_2) \text{ é } 0,85 \text{ então a saída é } u_{R1} = 0,18-5.0,40 \quad (3.61a)$$

$$R_2 : \text{ Se } \mu_N(x_1) \text{ é } 0,70 \text{ e } \mu_Z(x_2) \text{ é } \mathbf{0,20} \text{ então a saída é } u_{R2} = 2.0,18 - 4. 0,40 \quad (3.61b)$$

$$R_3 : \text{ Se } \mu_Z(x_1) \text{ é } 0,35 \text{ e } \mu_P(x_2) \text{ é } \mathbf{0,20} \text{ então a saída é } u_{R3} = 0,18 + 2. 0,40 - 1 \quad (3.61c)$$

Assim o resultado da *defuzzificação*, segundo a equação 3.60, é o seguinte:

$$u_{Sugeno} = \frac{0,70[-1,82] + 0,20.[1,60] + 0,20.[-1,62]}{0,70 + 0,20 + 0,20} = 1,6782 \quad (3.62)$$

Logo para a entrada $x_1 = -0,18$ e $x_2 = 0,40$ resulta, segundo as regras Sugeno propostas, em um controle $u = -1,6782$.

Comparando os dois tipos de regras descritos, conclui-se que a base de regras do tipo Mamdani é mais intuitiva, representando melhor a idéia humana. Por outro lado, a base de regra tipo Sugeno apresenta uma maior eficiência computacional, uma vez não há necessidade de implementação numérica do cálculo de áreas.

Uma similaridade que pode ser feita e discutida, é o efeito mútuo existente quando se utiliza controladores construídos com regras Mamdani com os controladores com regras Sugeno com somente um termo constante, $b_{0,k}$ da equação 3.59 (Batur e Kaparian,1991; Matsuoka,1991; Harris e Moore,1989).

No projeto de controladores *fuzzy* é importante analisar o espaço de estados do problema. Geralmente o desenvolvimento da base de regras é uma das maiores dificuldades da lógica *fuzzy*. Isto porque cada regra contribui com a saída em um dado instante de tempo e cada regra é avaliada em função da definição do método de *fuzzificação*, método de *defuzzificação* e os operadores de implicação, como por exemplo mínimo ou produto.

Quanto à *defuzzificação*, se for necessária maior precisão para o valor da saída para a base de regras do tipo Mamdani, é preferível escolher o método do centro de gravidade (CA) ao método da média do máximo (MM), conforme Urih e Tsoukalas (1997).

Capítulo IV

Fundamentos das Redes Neurais Artificiais

Um dos aspectos mais fascinantes da espécie humana é a sua capacidade de aprendizagem. O mecanismo exato de aprendizagem do Homem ainda é desconhecido, porém sabe-se que o aprendizado é realizado no cérebro. O cérebro é formado por uma rede de neurônios interconectados configurando um sistema complexo capaz de pensar, lembrar e resolver problemas. As redes neurais artificiais foram desenvolvidas, na década de 40, pelo neurologista McCulloch e pelo matemático Walter Pitts, os quais fizeram uma analogia entre os neurônios do cérebro e o processo eletrônico. O trabalho consistia num modelo de resistores variáveis e amplificadores representando conexões sinápticas de um neurônio.

4.1 O neurônio biológico

O neurônio biológico é a unidade celular fundamental do sistema nervoso cerebral. Um neurônio pode ser funcionalmente classificado em neurosensor e neurotransmissor. Os neurosensores percebem estímulos externos (temperatura, luminosidade, pressão, etc.) e os neuro-transmissores transmitem essas informações até o córtex cerebral, onde existem cerca de 10 bilhões de neurônios interconectados. Um neurônio é capaz de criar 10.000 sinapses, isto é, estabelecer até 10.000 conexões com neurônios adjacentes. Um neurônio, basicamente, capta o estímulo nos dendritos e os transmite pelo axônio para que se atinja dendritos de outros neurônios, como mostra o esquema de um neurônio biológico ilustrado na Figura 4.1.

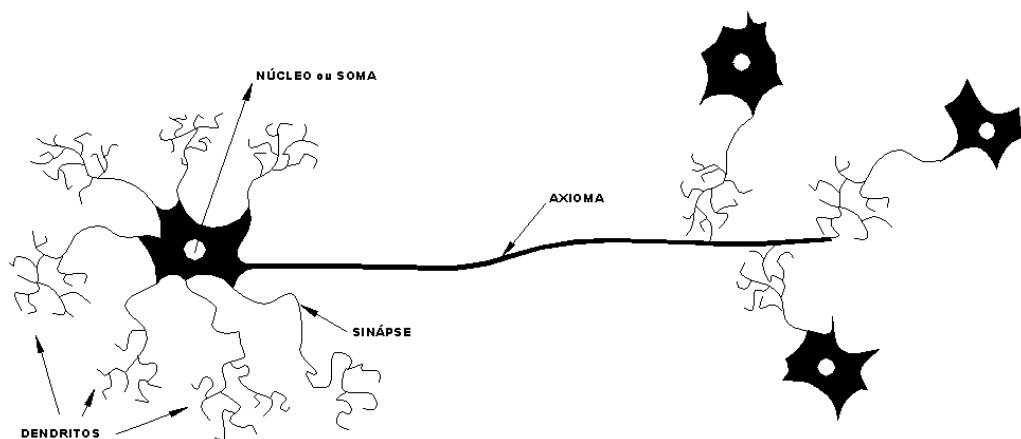


Figura 4.1 Neurônio Biológico

Uma vez captado o estímulo nas terminações nervosas, ele é transferido em um impulso elétrico. Esse impulso chamado potencial em ação, percorre o axônio e atinge outros neurônios. Os neurônios proporcionam um fluxo preciso das informações. Um forte (fraco) potencial em ação entre os contatos sinápticos, que ocorrem no núcleo ou soma de um neurônio, estimula um fluido neurotransmissor a produzir uma carga elétrica que acelera (retarda) o fluxo das informações a outros dendritos de milhares de neurônios. Quando existem fortes combinações sinápticas, é dito que as condutâncias ou impedâncias das cargas elétricas estão ajustadas, o que permite ao cérebro aprender e armazenar as informações processadas.

4.2 O neurônio artificial

Atentos aos estudos de neurociência, McCulloch e Pitts (1943) desenvolveram um modelo matemático análogo ao neurônio biológico, denominado neurônio artificial McCulloch-Pitts, que possui múltiplas entradas e uma única saída, onde cada uma das entradas possui um peso associado.

O neurônio artificial possui duas fases de processamento. Na primeira fase calcula-se o somatório do produto das entradas pelos pesos associados. Na segunda fase é atribuída uma função não linear, chamada de função de ativação, que é aplicada ao somatório resultante da primeira fase.

Veja na Figura 4.2 que a saída do neurônio é a imagem de uma função de ativação não linear tipo degrau.

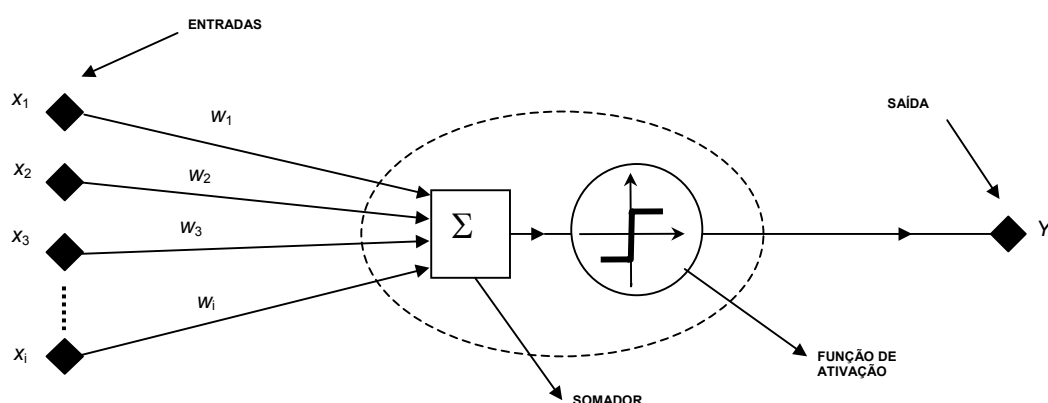


Figura 4.2 Neurônio artificial

A formulação matemática deste modelo, com função de ativação tipo degrau com imagem $\{0,1\}$ ou $\{-1,1\}$ pode ser vistas na Figuras 4.3 e é descrita por:

$$I = \sum_{i=1}^N w_i \cdot x_i \quad (4.1a)$$

$$Y = \phi(I) \quad (4.1b)$$

onde Y é a saída do neurônio, $\phi(I)$ é a função de ativação não linear e w_i são os pesos correspondentes às entrada x_i .

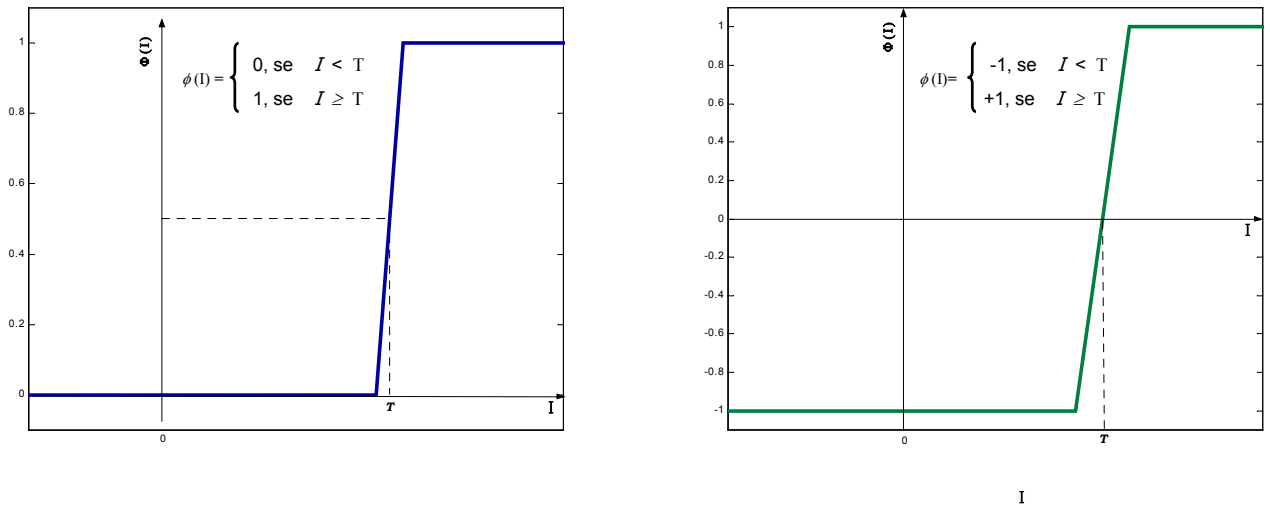


Figura 4.3 Funções de ativações de neurônios artificiais

Este modelo se aplica fundamentalmente aos problemas de classificação, onde as variáveis de entrada devem ser separadas em duas classes.

Porém, as funções de ativação mais utilizadas são as funções sigmóides, como o arco-tangente, a tangente hiperbólica, etc., que variam gradualmente entre as assíntotas: $\phi(I) = 0$ e $\phi(I) = 1$ ou $\phi(I) = -1$ e $\phi(I) = 1$. Entretanto, podem ser utilizadas funções de ativação do tipo linear: $\phi(I) = I$. Salienta-se que o uso de uma função de ativação linear remove a não linearidade do neurônio artificial, impossibilitando às redes neurais artificiais a identificação de fenômenos não lineares (Uhrig e Tsoukalas, 1997).

4.3 Redes neurais artificiais

Uma rede neural artificial consiste em um sistema de processamento de dados, com muitos neurônios artificiais interconectados e organizados em seqüências de camadas. Este arranjo entre camadas de neurônios configura a arquitetura de uma rede neural artificial, inspirado na estrutura cerebral do córtex. As camadas de uma rede neural são interconectadas através de parâmetros internos denominados pesos (w). A camada de entrada somente

apresenta os dados à rede neural, ela não possui neurônios de processamentos e a camada de saída resulta nos valores de saída da rede. As outras camadas são chamadas de intermediárias ou ocultas. A arquitetura da rede neural mostrada na Figura 4.4, por exemplo, é composta por uma camada de entrada com três neurônios, uma camada de saída com dois neurônios e uma camada oculta com cinco neurônios.

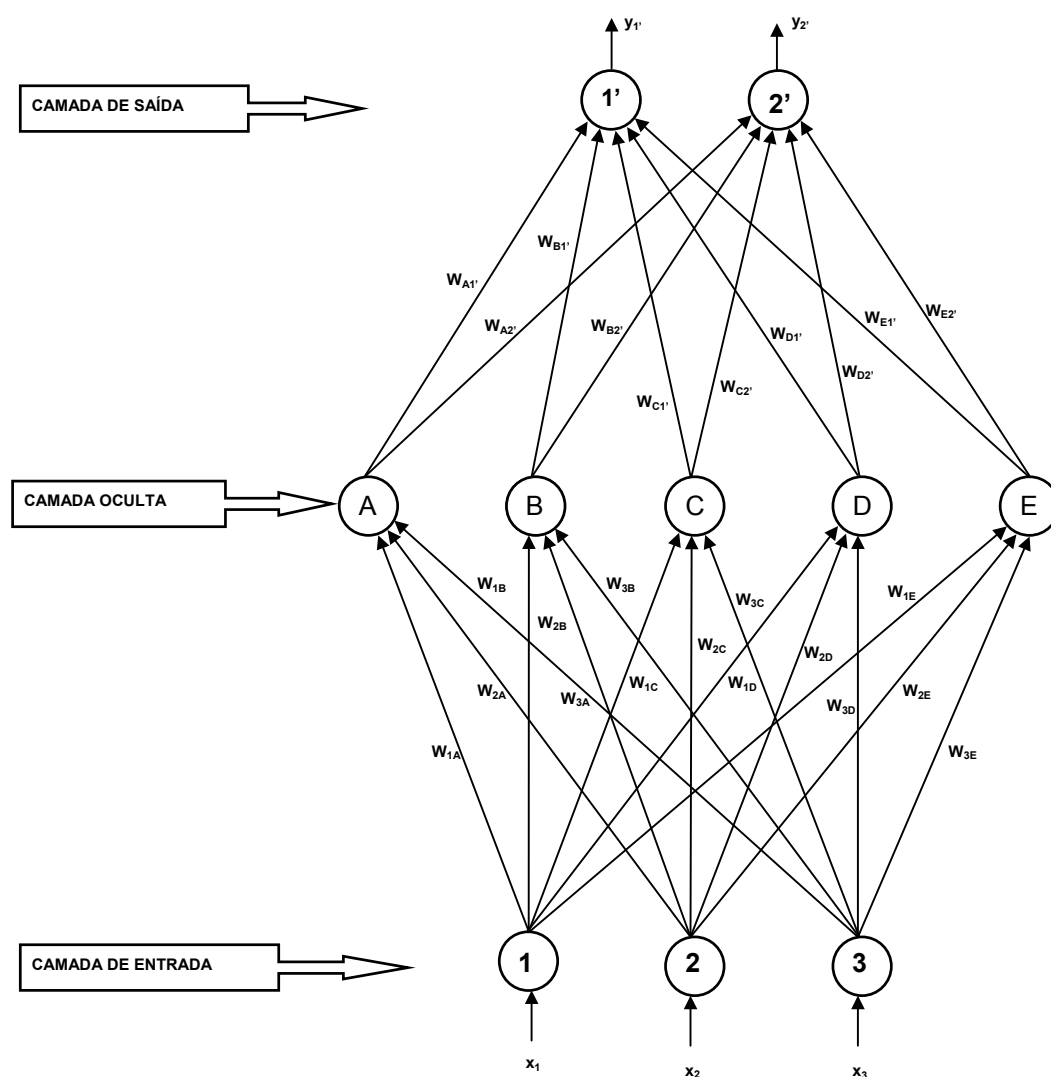


Figura 4.4 Arquitetura de uma rede neural *feedforward*

Há dois tipos gerais de redes neurais. As redes com memórias auto-associativas mapeiam parte dos dados relacionados a eles mesmos, memorizando as informações específicas mais relevantes. As redes hetero-associativas mapeiam um conjunto de padrões (entradas) relacionados a outros padrões (saídas). Nesta dissertação são utilizadas as redes hetero-associativas que são totalmente interconectadas por pesos.

4.3.1 Redes neurais: *feedforward* x *feedback*

Quando a rede neural apresenta arquitetura com conexões de realimentações (*feedback*), tanto entre neurônios de uma mesma camada como entre neurônios de camadas anteriores, o processo de obtenção das operações matemáticas para as saídas é mais complexo, porém após o treinamento as saídas são calculadas instantaneamente. Uma rede realimentada é chamada de rede recorrente, conforme mostra a Figura 4.5, onde a saída da rede tem uma conexão de realimentação para camada anterior, através do neurônio B. São exemplos de redes recorrentes as redes de Hopfield (1982) e Kohonen (1984). A rede recorrente apresenta uma realimentação após a propagação, o que favorece o treinamento. Deste modo a rede é treinada com propagações e realimentações, até que haja um equilíbrio entre as entradas, pesos e saídas ou algum critério de convergência seja atingido. Estas redes recorrentes não serão tratadas neste trabalho.

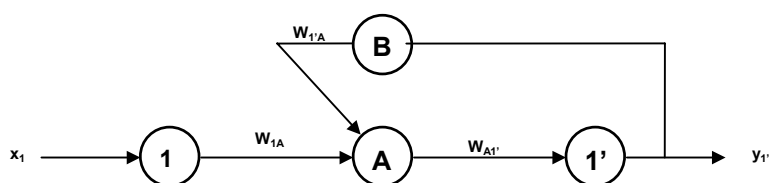


Figura 4.5 Arquitetura de uma rede neural recorrente

Quando a rede neural não apresenta em sua arquitetura interconexões entre os neurônios de uma mesma camada ou interconexão de realimentação (*feedback*) com neurônios de camadas anteriores, e a rede é caracterizada por um fluxo unidirecional recebe o nome de rede neural *feedforward*. Nessa rede, inicialmente o vetor de entradas é aplicado à camada de entrada. Então as funções de ativação são rapidamente calculadas e o processo flui da camada de entrada para a camada oculta e desta para a camada de saída. O aprendizado da rede, ou seja, o ajuste dos pesos, é realizado através de treinamento supervisionado. Após o treinamento, a rede é capaz de mapear uma simples entrada em uma saída desejável. São exemplos clássicos de redes *feedforward* as rede *adeline*, *madeline* e *perceptron*.

4.3.2 Redes neurais de associação linear

Uma das redes neurais mais elementares são as redes com associação linear, que possuem a arquitetura de uma camada de entrada interconectada a uma camada de saída, conforme ilustra a Figura 4.6. A função de ativação da camada de saída é linear. Inicialmente,

a partir de um conjunto de entradas e saídas padrões, retiram-se pares de entradas padrões com sua saídas correspondentes. A rede aprende quando existe entrada ativa, que ocorre quando uma entrada padrão produz, para um determinado conjunto de pesos, uma saída padrão. Para que todas as entradas se tornem ativas, é necessário o treinamento da matriz de pesos. A vantagem das redes com associação linear é a habilidade que apresentam para aprender mais de um padrão de saída simultaneamente.

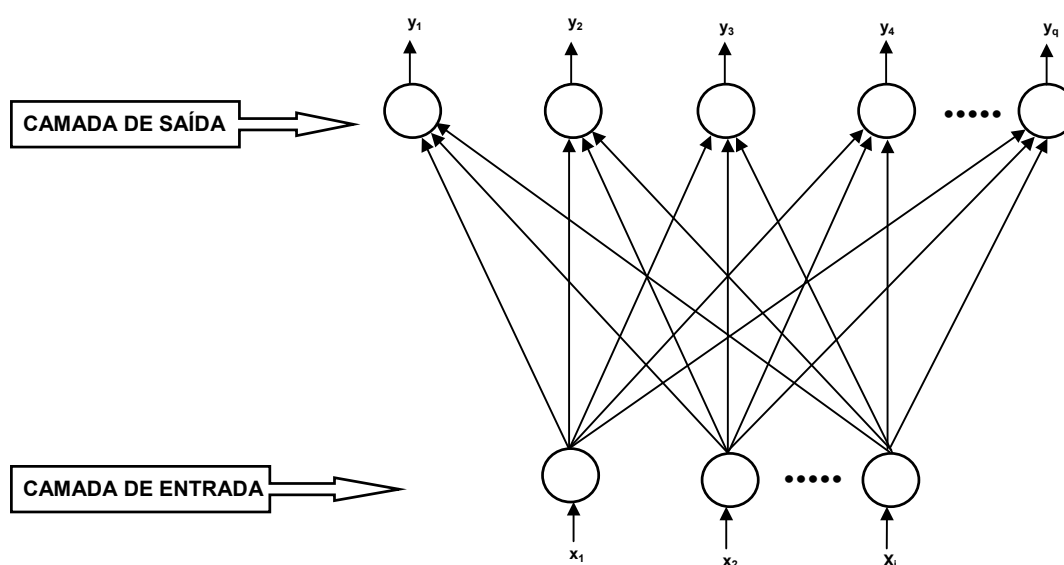


Figura 4.6 Rede neural com associação linear

4.3.3 Treinamento das redes neurais artificiais

O treinamento da rede é o processo iterativo de determinação dos pesos até que a rede neural produza a saída desejada para uma determinada entrada. Existem dois tipos distintos de treinamento:

(i) Treinamento não supervisionado ou auto-organizado: neste treinamento a rede não recebe nenhuma informação sobre como classificar as entradas e ajustar os pesos, a rede “*por si mesma*” faz as devidas correlações e desenvolve seu próprio esquema de classificação das entradas. As redes seriais de memória associativas (Kohonen, 1984), são exemplos de treinamento não supervisionado. Esta abordagem não será tratada nesta dissertação.

(ii) Treinamento supervisionado ou associativo: As redes comumente utilizadas em problemas de engenharia envolvem treinamento supervisionado. Este treinamento recebe este nome pois a rede neural artificial depende de um sistema auxiliar para supervisionar o treinamento, ou seja, é necessário um conjunto de dados de saída exemplares ou um certo conjunto de dados

experimentais confiáveis, ambos utilizados na avaliação da capacidade da rede em mapear a entradas em saídas. No treinamento supervisionado o vetor das variáveis de entrada padrão, que possui um correspondente vetor de variáveis de saída padrão, é aplicado à rede neural. Os pesos são adotados, num primeiro momento, randomicamente e com pequenos valores reais. O vetor de entrada produz o vetor de saída da rede pela propagação das entradas pelas camadas ocultas e pela camada de saída. A saída da rede é comparada com o vetor de saída padrão e a diferença (erro) entre a saída da rede e a saída desejada é utilizada no ajuste dos pesos da rede neural. Com os pesos ajustados um novo par de entradas e saídas padrões é novamente apresentado à rede, onde calcula-se novamente a saída da rede para um novo ajuste dos pesos. O treinamento se repete até que a rede neural produza saídas próximas às saídas desejadas. O treinamento supervisionado é esquematizado na Figura 4.7, onde é incluído um vetor padrão de saídas desejadas para cada entrada de treinamento.

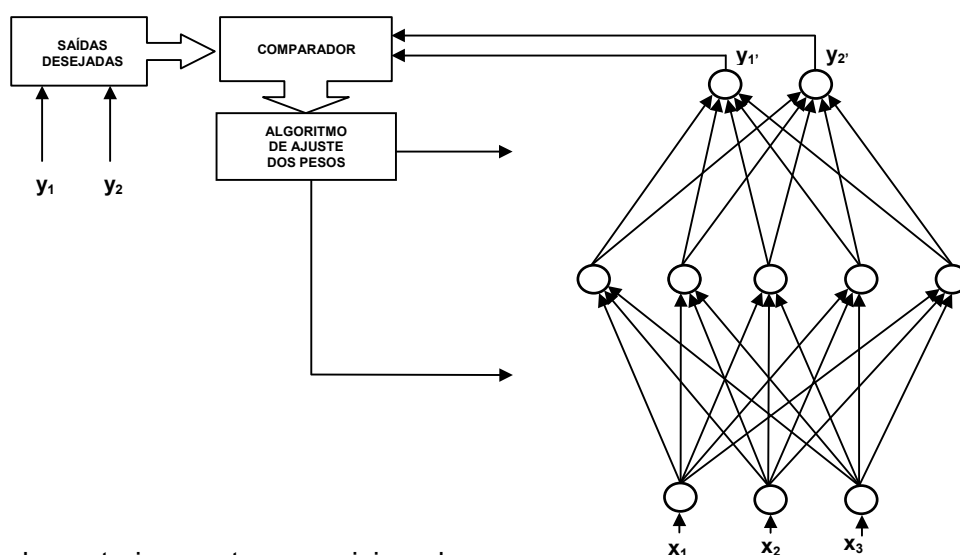


Figura 4.7 Rede neural com treinamento supervisionado

4.4 Histórico das redes neurais artificiais

A primeira experiência prática de aplicações de redes neurais se deve a Minsky e Dean que, em 1951, construíram no MIT (Massachusetts Institute of Technology), uma máquina baseada em pistões e motores que apresentava certa capacidade de aprendizado, com uma rede de 40 elementos de processamento. Após a primeira Conferência Internacional de Inteligência Artificial, realizada em Dartmouth (1956), Roseblatt (1958) apresentou os conceitos fundamentais do *perceptron*, que adiciona uma metodologia de aprendizado para as redes compostas por neurônios de McCulloch-Pitts. Simultaneamente ao desenvolvimento do *perceptron*, Widrow e Hoff (1960) introduziram as redes com elemento linear adaptativo (*adaline*).

4.4.1 As redes neurais *adaline* e *madaline*

As redes *adalines*, proposta por Windrow e Hoff (1960), possuem entradas $+1$ ou -1 , logo são redes classificatórias binárias, isto é, um elemento pode classificar as entradas em dois padrões. Esta rede possui neurônios com função de ativação não linear. O conjunto de pesos são inicializados com pequenos valores gerados aleatoriamente. É inserido um neurônio extra de valor unitário chamado *bias* (b_0), que é um termo constante independente do vetor das entradas da rede.

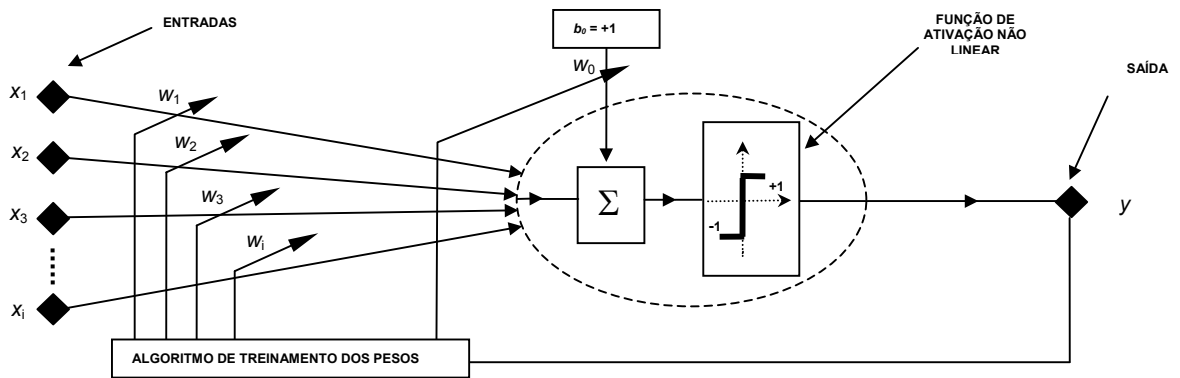


Figura 4.8: Diagrama de um neurônio da rede *adeline*

A rede *adeline* forma uma saída binária $+1$ ou -1 , conforme a Figura 4.8 e a equação:

$$y = \begin{cases} +1 : z = \sum_{i=1}^N (w_i x_i) + w_0 \cdot b_0 \geq 0 \\ -1 : z = \sum_{i=1}^N (w_i x_i) + w_0 \cdot b_0 < 0 \end{cases} \quad (4.2)$$

onde N é o número de entradas, w_i são os pesos da rede. A regra de atualização dos pesos, proposta por Windrow-Hoff, utiliza o algoritmo do gradiente descendente (*steepest-descent*), que minimiza a seguinte função:

$$J = \frac{1}{2} \varepsilon^2 \quad (4.3a)$$

$$\varepsilon = D - z = D - \left[\sum_{i=1}^N (w_i x_i) - w_0 \cdot b_0 \right] \quad (4.3b)$$

onde J representa o erro médio quadrático (função objetivo) entre uma dada entrada propagada pela rede e uma resposta exemplar D ; w_i e x_i são, respectivamente, os pesos associados às entradas, para $i=1,2,\dots,N$. A derivada da função objetivo em relação a cada w_i é:

$$\frac{\partial J}{\partial w_i} = \frac{\partial J}{\partial y} \cdot \frac{\partial y}{\partial w_i} = -[D - z]x_i \quad (4.4)$$

A lei de recorrência para atualização dos pesos é dada por:

$$\hat{w}_i = w_i - \eta \frac{\partial \mathcal{E}}{\partial w_i} = w_i - \eta \cdot \mathcal{E} \cdot x_i \quad (4.5)$$

onde $\hat{w}_i$ é o valor do peso atualizado e η é a taxa de aprendizado que define o tamanho do passo do algoritmo do gradiente descendente. Normalmente, este valor é obtido por inspeções partindo de pequenos valores constantes.

Os pesos são ajustados para cada exemplar do conjunto de treinamento. O neurônio deve ser treinado repetitivamente até que atinja os pontos corretos de classificação e encontre um conjunto de pesos ótimos.

Uma extensão das redes *adelines* é a rede *madaline*. Estas redes apresentam a camada de entrada, a camada oculta (composta de elementos *adelines*) e a camada de saída, conforme mostra a Figura 4.9. A saída pode ser calculada de diversas maneiras, por exemplo, se as saídas dos elementos *adelines* são na maioria +1 então a saída da *madaline* é +1. Uma vez que a saída das redes *madalines* são binárias, estas redes somente são utilizadas em problemas de classificação em duas classes.

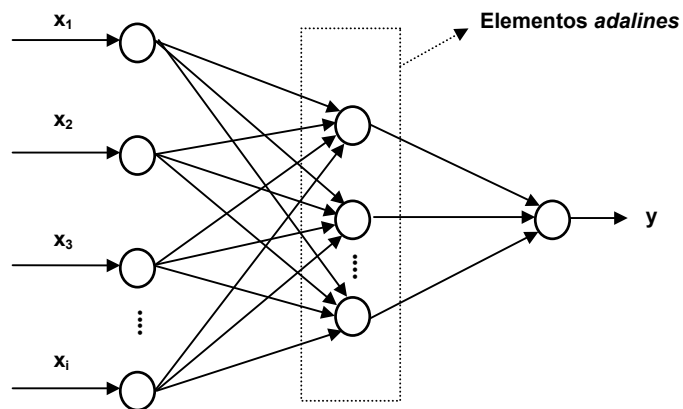


Figura 4.9: Arquitetura de uma rede neural *madalines* com múltiplos neurônios *adelines* na camada oculta.

4.4.2 A rede neural *perceptron*

Após a Conferência de Dartmouth, Rosenblatt (1958) desenvolveu um modelo computacional para modelar a retina dos olhos, chamado de *perceptron*, que é uma rede neural *feedforward* com uma única camada composta de neurônios de McCulloch-Pitts operando em paralelo, cada qual com sua saída e todos com as mesmas entradas, conforme mostra a Figura 4.10.

As entradas também são valores binários unitários. Para atualização dos pesos, o *perceptron* faz uso do aprendizado de Hebb.

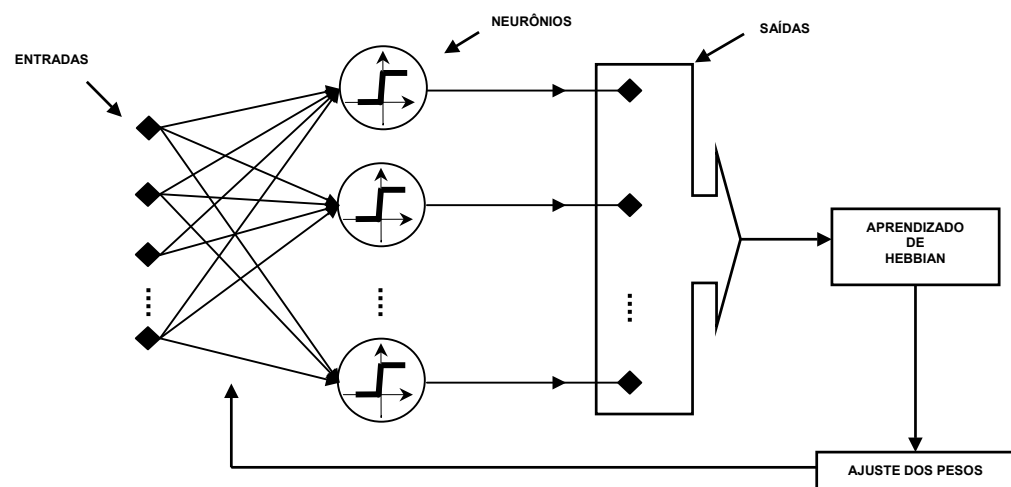


Figura 4.10 A rede neural *perceptron* de Rosenblatt's

Este aprendizado, proposto pelo biólogo Hebb, em 1949, pode assim ser escrito:

“...se um neurônio A estimula repetidamente um neurônio B e o neurônio B fornece a saída desejada, então o peso da sinapse entre A e B deve manter a mesma magnitude...”.

Este método de aprendizado pode assumir três concepções de atualização dos pesos:

$$\text{se } Y_q = Y_{\text{desejado}} \quad \text{então } \hat{w}_i = w_i, \text{ senão} \quad (4.6a)$$

$$\text{se } Y_q = 0 \quad \text{então } \hat{w}_i = w_i + x_i, \text{ senão} \quad (4.6b)$$

$$\text{se } Y_q = 1 \quad \text{então } \hat{w}_i = w_i - x_i \quad (4.6c)$$

onde $\hat{w}_i$ é o peso atualizado associado à entrada x_i e à saída Y_q .

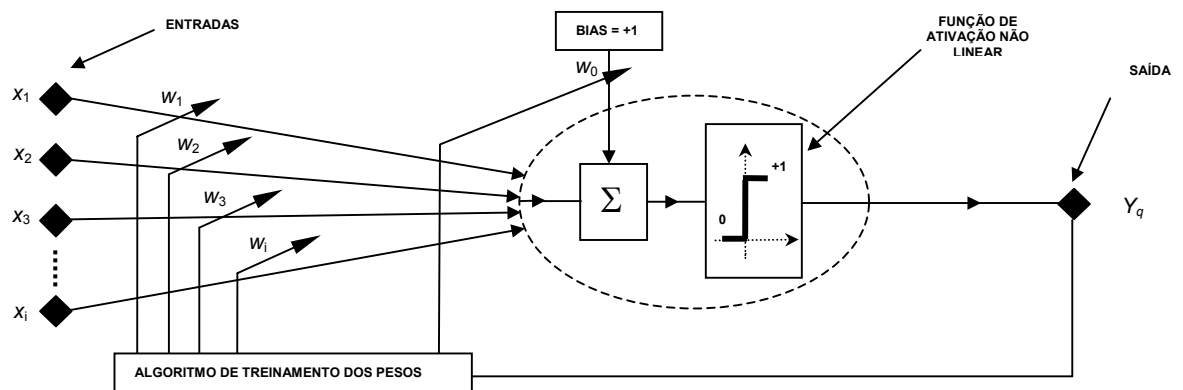


Figura 4.11: Diagrama de um neurônio da rede *perceptron*

Para atualização dos pesos segundo o aprendizado Hebbiano, basta calcular os valores das saídas Y_q de todos os neurônios da rede da Figura 4.11, da seguinte forma:

$$Y_q = \text{sgn} \left[\sum_{i=1}^N (w_i x_i) + w_0 \cdot 1 \right] = \begin{cases} 0, & \text{se positivo} \\ 1, & \text{se negativo} \end{cases} \quad (4.7)$$

onde sgn é o sinal do resultado da propagação de uma entrada pela função de ativação do neurônio da camada oculta.

O que diferencia o aprendizado de Hebb em relação ao aprendizado pela regra delta de Windrow-Hoff, é o critério de atualização dos pesos. Enquanto a rede não convergir para todos os padrões de entradas, novas entradas são apresentadas à rede e os pesos são atualizados, e caso alguma saída coincida com a saída desejada, os pesos não são atualizados.

Minsky e Papert (1969), com a publicação do livro *Perceptron*, provaram as limitações para um *perceptron* de uma simples camada, pois operam somente com problemas que apresentam o espaço de soluções linearmente separáveis. O *ou-exclusivo* (XOR) é citado como um problema elementar que o *perceptron* não resolve. Provaram, também, que uma rede *feedforward* com uma camada oculta soluciona este problema elementar do *ou-exclusivo*, porém não apresentaram uma solução para atualização dos pesos de redes multi-camadas para problemas mais complexos.

Diante da ausência de uma solução para atualização dos pesos de redes multi-camadas, as pesquisas com redes neurais cessaram por um período. O responsável pelo reaparecimento das redes neurais artificiais foi Jonh Hopfield, que em 1982 propôs um sistema neural composto de diversos elementos de processamento interconectados, que buscam

atingir um estado de energia mínima. Hopfield define um critério de convergência para as saídas da rede, baseado na estabilidade da energia mínima dos elementos da rede que, em operação, representa o modelo físico sob um determinado nível de excitação. Surge então, o método *backpropagation* para treinamento das redes neurais multi-camadas que foi desenvolvido por Parker (1982). Curiosamente, com o passar dos anos, descobre-se que Werbos (1974) também desenvolverá o método *backpropagation* em sua tese de Ph.D.. A partir desta metodologia de treinamento as redes neurais invadiram os centros de ciências e tecnologia.

4.5 O treinamento *backpropagation* e seu algoritmo

O *backpropagation* é um método sistemático de treinamento de redes neurais multi-camadas, que tenta solucionar a atualização dos pesos. O método só teve comprovação experimental no trabalho de Rumelhart *et al.* (1986).

A idéia principal do método *backpropagation*, consiste no cálculo do erro médio quadrático, entre as saídas da rede e as saídas desejadas, que se propaga atualizando tanto os pesos da interconexão entre a camada oculta e a camada de saída, como os pesos das interconexões entre a camada de entrada e a camada oculta.

A Figura 4.12 ilustra uma rede multi-camadas com arquitetura do tipo *feed-forward*, com uma camada de entrada, duas ocultas e uma camada de saída. Normalmente adiciona-se nas camadas de entrada e nas ocultas, um neurônio extra chamado *bias* de valor unitário. Todos os neurônios são interconectados através dos pesos.

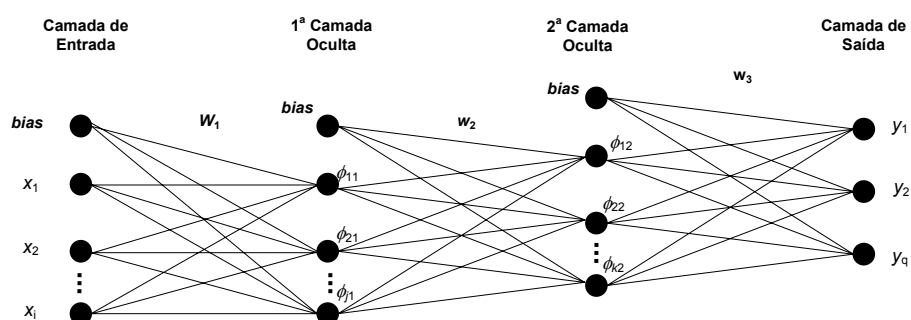


Figura 4.12 Arquitetura de uma rede neural artificial com quatro camadas

O modelo de um típico neurônio utilizado no algoritmo *backpropagation*, pode ser visto na Figura 4.13, onde as x_i entradas são conectadas pelos seus respectivos pesos w_i para processamento da saída por intermédio da função de ativação $\phi(I)$.

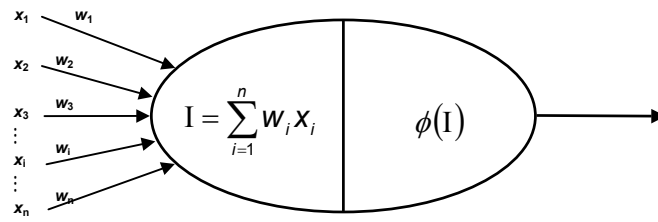


Figura 4.13 Neurônio de processamento do método *backpropagation* para redes multi-camadas

4.5.1 Funções de ativação para o *backpropagation*

As entradas de uma função de ativação são: o somatório dos produtos dos pesos pelas respectivas entradas ou o somatório dos produtos dos pesos pelos respectivos valores das funções de ativação das camadas ocultas, conforme equações mostradas a seguir:

$$I = w_1 x_1 + w_2 x_2 + w_3 x_3 + \dots + w_n x_n = \sum_{i=1}^n w_i x_i \quad (4.8a)$$

ou,

$$I = w_1 \phi_1 + w_2 \phi_2 + w_3 \phi_3 + \dots + w_n \phi_n = \sum_{i=1}^n w_i \phi_i \quad (4.8b)$$

O algoritmo *backpropagation* requer funções de ativação monotonicamente crescentes e diferenciáveis. Uma função sigmoidal não linear muito utilizada é a função logística.

A Figura 4.14 apresenta a formulação e as curvas das funções logísticas, bem como suas derivadas.

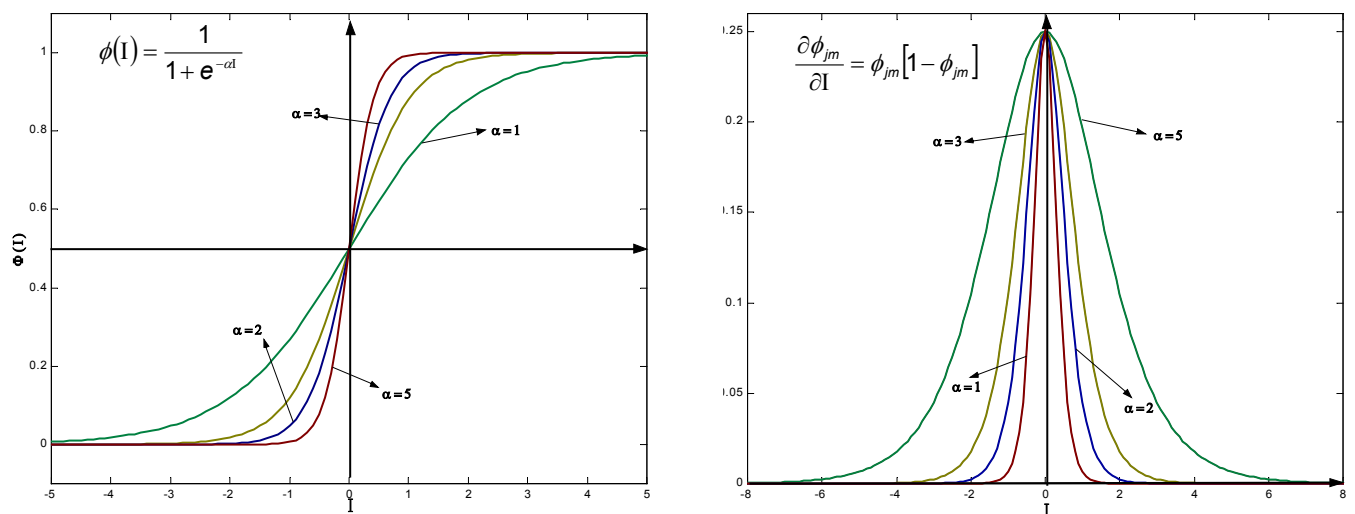


Figura 4.14 Funções de ativação logística e suas derivadas

O arco-tangente é também uma função sigmoideal que atende aos critérios do algoritmo e é muito explorada. A formulação e a curva da função de ativação arco-tangente e da sua primeira derivada são mostradas na Figura 4.15.

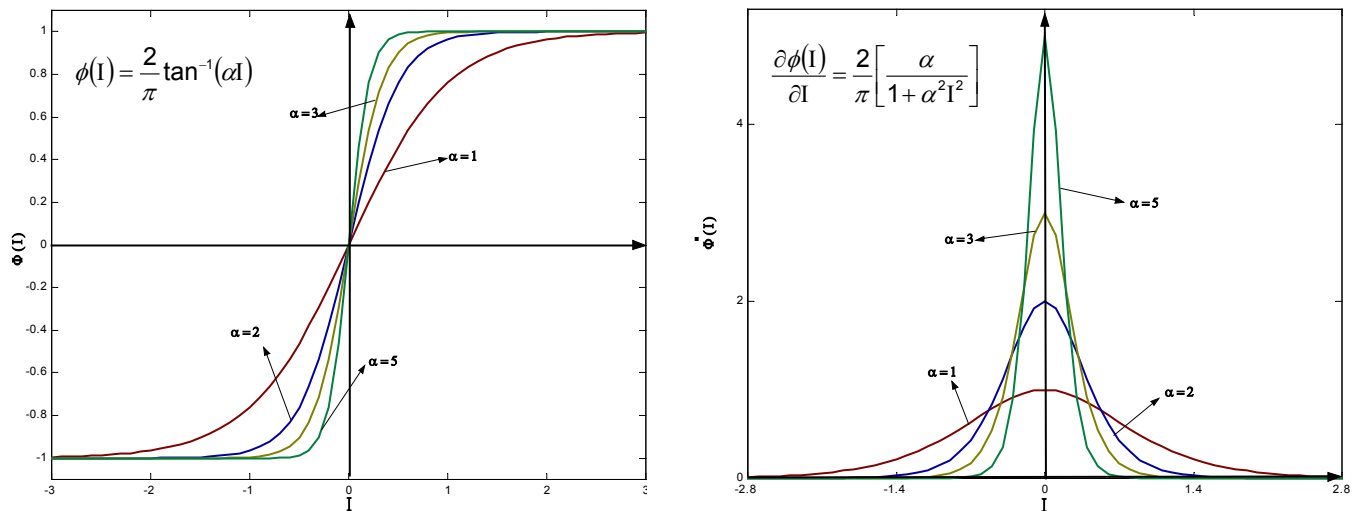


Figura 4.15 Funções de ativação arco-tangente e suas derivadas

Pode ser visto nas Figuras 4.14 e 4.15 que o maior gradiente ocorre para $I=0$. Hines (1997) afirma que a velocidade de aprendizado é parcialmente dependente da magnitude da derivada. Então, a ativação interna de todos os neurônios deve manter valores pequenos para acelerar o treinamento. Por esse motivo deve-se escalonar as variáveis de entrada assim como inicializar os pesos com pequenos valores randômicos. A adição dos neurônios *biases* de valor unitário também busca acelerar o treinamento.

Um dos parâmetros importantes no projeto de uma rede neural é a escolha adequada das funções de ativação dos neurônios, assim como o número de neurônios por camadas e a quantidade de camadas ocultas.

Para a realização do treinamento de uma rede neural multi-camadas é necessário um conjunto de padrões de entradas e saídas, geralmente resultantes de ensaios experimentais. Cada padrão de treinamento, entradas e saídas, apresentado à rede é processado em dois estágios que são descritos a seguir.

4.5.2 Estágio de propagação das entradas para o treinamento *backpropagation*

Para um melhor entendimento do primeiro estágio, suponha uma rede neural utilizada para o mapeamento de um problema que tenha x_i entradas e y_q saídas, conforme mostra a Figura 4.16 com apenas uma camada oculta.

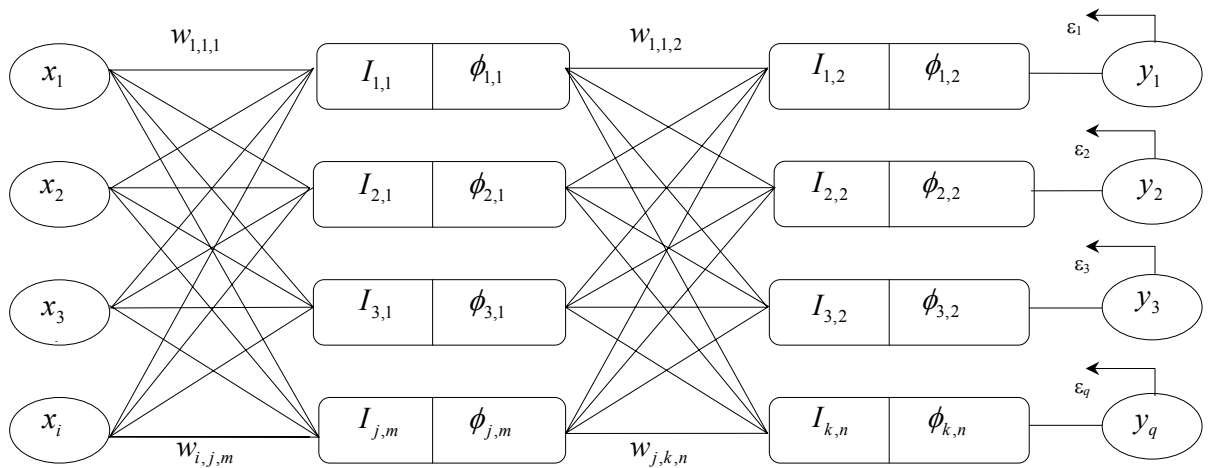


Figura 4.16 Esquema de uma rede neural com uma camada oculta mostrando os símbolos e índices para a compreensão dos dois estágios

O primeiro estágio se inicia com a geração aleatória de todos os pesos com pequenos valores reais e o fluxo de propagação parte da apresentação das entradas aos neurônios de processamento da camada oculta. Os neurônios da camada oculta inicialmente calculam os somatórios em $I_{1,1} \dots I_{j,m}$ das entradas $x_1 \dots x_i$ multiplicadas pelos seus respectivos pesos $w_{111} \dots w_{ijm}$, conforme mostra a equação 4.8a, e aplicam aos resultados as funções de ativação $\phi(I_{11}) \dots \phi(I_{j,m})$. As funções de ativação $\phi(I_{j,m})$ devem ser diferenciáveis e monotonicamente crescentes. Neste momento a rede se propaga para a camada de saída a partir dos resultados obtidos na camada oculta. Desta forma, todas as saídas dos neurônios da camada oculta são propagados para os neurônios de processamento da camada de saída, que calculam os somatórios em $I_{1,1} \dots I_{k,n}$ das entradas $\phi(I_{11}) \dots \phi(I_{j,m})$ multiplicadas pelos seus respectivos pesos $w_{112} \dots w_{jkn}$, conforme mostra a equação 4.8b, e aplica aos resultados as funções de ativação $\phi(I_{12}) \dots \phi(I_{k,n})$. O primeiro estágio se encerra com a propagação das saídas $\phi(I_{12}) \dots \phi(I_{k,n})$.

4.5.3 Estágio de atualização dos pesos para o treinamento *backpropagation*

Uma vez propagadas as entradas e obtidas as saídas da rede, define-se o erro médio quadrático como:

$$\varepsilon^2 = [Y_q - \phi_{kn}]^2 \quad (4.9)$$

onde Y_q são as saídas desejadas e ϕ_{kn} são as saídas propagadas da rede. A atualização dos pesos configura o segundo estágio do treinamento.

Os pesos que estão entre a camada oculta e a camada de saída da rede são atualizados em função da mudança do erro médio quadrático em relação aos respectivos pesos, da seguinte forma:

$$\Delta w_{jkn} = -\eta_{jkn} \frac{\partial \varepsilon^2}{\partial w_{jkn}} \quad (4.10)$$

onde η_{jkn} é uma constante de proporcionalidade associada à taxa de aprendizado dos pesos w_{jkn} . Para resolver esta derivada parcial utiliza-se a regra da cadeia, ou seja:

$$\Delta w_{jkn} = \eta_{jkn} \cdot \frac{\partial \varepsilon^2}{\partial \phi_{kn}} \cdot \frac{\partial \phi_{kn}}{\partial I_{kn}} \cdot \frac{\partial I_{kn}}{\partial w_{jkn}} \quad (4.11)$$

De forma análoga, a atualização dos pesos que estão entre a camada de entrada e a camada interna, é proporcional à taxa de mudança do erro médio quadrático em relação aos respectivos pesos, assim:

$$\Delta w_{ijm} = -\eta_{ijm} \frac{\partial \varepsilon^2}{\partial w_{ijm}} \quad (4.12)$$

onde η_{ijm} é outra constante de proporcionalidade da taxa de aprendizado relativo à atualização dos pesos w_{ijm} . Novamente utilizando a regra da cadeia, a equação 4.12 pode assim ser escrita:

$$\Delta w_{ijm} = -\eta_{ijm} \cdot \frac{\partial \varepsilon_q^2}{\partial \phi_{kn}} \cdot \frac{\partial \phi_{kn}}{\partial I_{kn}} \cdot \frac{\partial I_{kn}}{\partial \Phi_{jm}} \cdot \frac{\partial \Phi_{jm}}{\partial I_{jm}} \cdot \frac{\partial I_{jm}}{\partial w_{ijm}} \quad (4.13)$$

Para maior clareza destas atualizações, será feito um cálculo para o caso específico de uma arquitetura que apresenta na camada oculta funções tipo logística e na camada de saída funções lineares. Neste caso a atualização dos pesos das conexões entre a camada oculta e camada de saída é dado por:

$$\varepsilon^2 = [Y_q - \phi_{kn}]^2 \quad ; \quad \frac{\partial \varepsilon^2}{\partial \phi_{kn}} = -2[Y_q - \phi_{kn}] \quad (4.14a)$$

$$\phi_{kn} = I_{kn} \quad ; \quad \frac{\partial \phi_{kn}}{\partial I_{kn}} = 1 \quad (4.14b)$$

$$I_{kn} = \sum_{j=1}^m w_{jkn} \phi_{jm} \quad ; \quad \frac{\partial I_{kn}}{\partial w_{jkn}} = \phi_{jm} \quad (4.14c)$$

A substituição dessas derivadas na equação 4.11, resulta em:

$$\Delta w_{jkn} = -\eta_{jkn} \cdot (-2)[Y_q - \phi_{kn}] \cdot 1 \cdot \phi_{jm} = \eta_{jkn} \cdot \delta_{jkn} \cdot \phi_{jm} \quad (4.15a)$$

onde:

$$\delta_{jkn} = 2[Y_q - \phi_{kn}] \quad (4.15b)$$

Logo, os pesos w_{jkn} são atualizados segundo a seguinte lei de recorrência:

$$w_{jkn}(N+1) = w_{jkn}(N) + \eta_{jkn} \cdot \delta_{jkn} \cdot \phi_{jm} \quad (4.16)$$

onde N é o número de iteração.

Para o cálculo de atualização dos pesos que estão entre a camada de entrada e a oculta utiliza-se a equação 4.13. Relembrando que as três primeiras derivadas parciais resultantes já foram calculadas, conforme as equações 4.14 a-c, resta calcular as duas últimas derivadas parciais da equação 4.13. Elas são dadas por:

$$\phi_{jm}(I) = \frac{1}{1 + e^{-I_{jm}}} \quad ; \quad \frac{\partial \phi_{jm}}{\partial I} = \phi_{jm} [1 - \phi_{jm}] \quad (4.17a)$$

$$I_{jm} = \sum_{i=1}^m w_{ijm} x_i \quad ; \quad \frac{\partial I_{jm}}{\partial w_{ijm}} = x_i \quad (4.17b)$$

Substituindo as equações 4.14 e 4.17 em 4.13, tem-se que:

$$\Delta w_{ijm} = -\eta_{ijm} (-2) [Y_q - \phi_{kn}] \cdot 1 \cdot w_{jkn} \cdot \phi_{jm} [1 - \phi_{jm}] x_i \quad (4.18a)$$

ou :

$$\Delta w_{ijm} = -\eta_{ijm} \cdot \delta_{jkn} \cdot w_{jkn} \cdot \phi_{jm} [1 - \phi_{jm}] x_i \quad (4.18b)$$

onde :

$$\delta_{ijm} = \delta_{jkn} w_{jkn} \frac{\partial \Phi_{jm}}{\partial I_{jm}} \quad (4.18c)$$

Logo, os pesos w_{ijm} são atualizados segundo a seguinte lei de recorrência:

$$w_{ijm}(N+1) = w_{ijm}(N) + \eta_{ijm} \cdot \delta_{ijm} \cdot x_i \quad (4.19)$$

onde N é o número da iteração.

Para cada um dos pesos das interconexões entre a camada de entrada e a camada oculta ou entre a camada oculta e a camada de saída, é aplicado o processo descrito de ajuste dos pesos.

Nas duas leis de recorrência para a atualização dos pesos, equações 4.16 e 4.19, aparece um termo η referente à taxa de aprendizado. Esta variável η , inicialmente estimada, também é atualizada, de iteração a iteração, segundo o seguinte critério definido por Adeli e Hung (1995):

Se o erro calculado está se aproximando do erro desejado (4.20a)

a taxa de aprendizado é reduzida, senão (4.20b)

a taxa de aprendizado é aumentada. (4.20c)

A idéia deste procedimento é manter a mudança dos pesos proporcional às suas variações. Na prática, escolhe-se uma taxa de aprendizado de elevada magnitude que proporciona poucas oscilações na direção de busca. Uma maneira de evitar bruscas oscilações

sem a necessidade de escolher uma taxa de aprendizado pequena, é a inserção do termo *momentum*. Assim as equações de recorrência 4.16 e 4.19, ficam assim reescritas:

$$w_{jkn}(N+1) = \alpha \cdot w_{jkn}(N) + \eta_{jkn} \cdot \delta_{jkn} \cdot \phi_{jm} \quad (4.21a)$$

$$w_{ijm}(N+1) = \alpha \cdot w_{ijm}(N) + \eta_{ijm} \cdot \delta_{ijm} \cdot x_i \quad (4.21b)$$

onde α é uma constante que determina o efeito prévio de mudança dos pesos.

Quando o termo de *momentum* não é utilizado, o mínimo demora a ser encontrado se a taxa de aprendizado for pequena. Caso contrário, se a taxa de aprendizado for grande, o mínimo nunca é atingido devido às grandes oscilações de atualização. Por outro lado, quando se adiciona o termo de *momentum*, o mínimo é encontrado mais rapidamente.

4.3 Metodologia de treinamento e validação de uma rede neural

Para realização do treinamento e da validação são necessários os seguintes passos:

1. Coleta de dados: Inicialmente os dados são coletados ou estimados, e separados em conjunto de dados de treinamento e conjunto de dados de validação.

A validação de uma rede neural consiste em apresentar o conjunto de validação à rede treinada, ou seja, com os pesos ajustados, e numa única propagação obter os resultados esperados. Portanto, os dados de treinamento devem cobrir todo o espaço das entradas onde se espera que a rede opere. Caso contrário, é possível que não se consiga a validação da rede para uma determinada entrada não presente nos espaços de entradas utilizadas no treinamento.

O tamanho do conjunto de treinamento e validação depende do sistema que se quer mapear, modelar ou identificar. Quanto maior o número de pontos de treinamento, maior o tempo computacional despendido no treinamento. Por outro lado, uma quantidade pequena de pontos pode ser insuficiente para o mapeamento de todas as possibilidades.

2. Normalização ou escalonamento dos dados: As redes neurais são extremamente sensíveis a valores absolutos em diferentes escalas. Portanto, é necessário algum tipo de normalização ou escalonamento dos dados de treinamento e validação. Salienta-se que este procedimento prevê, também, um bom condicionamento numérico.

3. Escolha da arquitetura da rede neural: Uma vez normalizados os dados coletados para treinamento e validação da rede neural, é necessária a seleção da arquitetura da rede, ou seja, os parâmetros tais como quantidade de camadas ocultas, tipo de função de ativação, número de neurônios por camada e *bias*.

Normalmente nos problemas de engenharia, as redes com treinamento *backpropagation* possuem três camadas porém, conforme afirmam Tsoukalas e Uhrig (1997), sob determinadas circunstâncias é mais vantajoso o uso de quatro camadas, isto é, optar por uma rede de duas camadas ocultas pequenas a uma única grande camada oculta.

Os números de neurônios nas camadas de entrada e saída são determinados pela natureza do problema. Os números de neurônios nas camadas ocultas são obtidos a partir de experimentações. Bailey e Thompson (1990) dizem que na prática é comum às redes com somente uma camada oculta, utilizar aproximadamente dois terços do número de neurônios da camada da entrada.

4. Treinamento: Após configurada a arquitetura da rede, os pesos são inicializados e a rede é treinada até o ajuste dos pesos. Caso não atinja um erro pré-estabelecido uma das estratégias é retomar mudando a arquitetura e/ou seus parâmetros de inicialização.

A existência do erro elevado pode ter várias razões:

- (i) os pesos podem estar desajustados;
- (ii) uma escolha inadequada para as funções de ativação;
- (iii) escalonamento inadequado dos dados coletados.
- (iv) a quantidade de pesos a serem ajustados pode estar inadequada e o algoritmo de busca encontra um mínimo local. O algoritmo *backpropagation* é um procedimento de otimização baseado no gradiente descendente que, matematicamente, pode ser interpretado como a minimização de um superfície multidimensional complexa sujeito a vales e cristas. Quando ocorre um mínimo local o treinamento tem que ser reiniciado;
- (vi) as informações presentes nos dados de treinamento são insuficientes e não representativas do fenômeno como um todo.

O treinamento é terminado quando a rede atingir o erro pré-estabelecido.

5. Validação: Após o treinamento a rede necessita se validada para que possa entrar em operação. Uma das precauções que devem ser tomadas é quanto à possível ocorrência de *overfitting*, isto é, quando o erro atingido com os dados de treinamento é significativamente superior ao erro com os dados de validação.

4.6 Identificação de sistemas dinâmicos

Uma das aplicações mais interessantes de redes neurais em engenharia, é na área de identificação de sistemas dinâmicos. O algoritmo *backpropagation* é o método mais comumente utilizado no treinamento e projeto de redes neurais. Porém, a maior limitação do algoritmo *backpropagation* é que ele somente aprende dados mapeados em saídas de forma estática. Entretanto, sabe-se que em sistema dinâmicos o tempo é importante na maioria de aplicações práticas. O tempo está implícito na velocidade, no processamento de sinais etc. Para solucionar esta limitação do *backpropagation* é necessário tornar as redes neurais dinâmicas, e para isso é necessário inserir na rede memória das variáveis de entrada (Elman,1990). Uma maneira de introduzir memória é incluir atrasos no tempo das variáveis de entrada e saída da arquitetura da rede neural. O uso de atrasos no tempo é neurologicamente motivado (Braitenberg,1990; Miller,1987), pois o cérebro possui informações passadas que são utilizadas no processamento de informações neurobiológicas futuras.

Para identificar um sistema é necessário treinar uma rede neural em paralelo com o sistema físico, aplicando tanto para o sistema físico como para a rede neural um mesmo sinal de entrada. Utilizando a resposta do sistema físico como a saída desejada para a rede neural, a rede é treinada até que o erro entre a saída do sistema físico e a saída da rede seja aceitável. A Figura 4.17 mostra a configuração de uma rede neural em paralelo com o sistema físico a ser identificado.

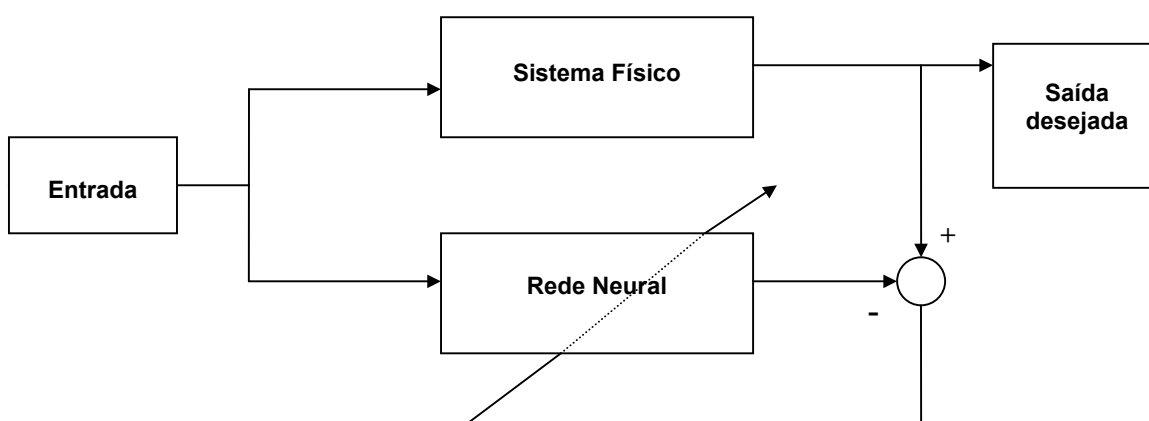


Figura 4.17 Diagrama esquemático da identificação de sistemas dinâmicos via redes neurais artificiais

Após o treinamento, a expectativa é que tanto o sistema físico como a rede neural apresentem uma mesma saída, diante de uma mesma entrada. Isto será verdade se as variáveis escolhidas para treinamento estiverem dentro de um domínio apropriado para um problema específico. Por exemplo, considere um sistema mecânico com uma massa sujeita a

vibrações. Suponha que para o treinamento, um dos dados de entrada da rede seja o sinal do deslocamento da massa excitada externamente numa faixa de frequência de 0,1 até 30 Hz. Após a rede treinada, a validação somente será possível com dados do sinal de deslocamento dentro da faixa de frequência adotada no treinamento, isto é, a rede não conseguirá validar dados fora desta faixa de frequência.

Conforme já foi dito, os sistemas dinâmicos envolvem de uma forma geral atrasos entre os sinais de entrada e os sinais de saída. Este comportamento deve ser considerado na definição da arquitetura de uma rede neural que pretenda modelar tais sistemas dinâmicos. Uma configuração possível é mostrada na Figura 4.18.

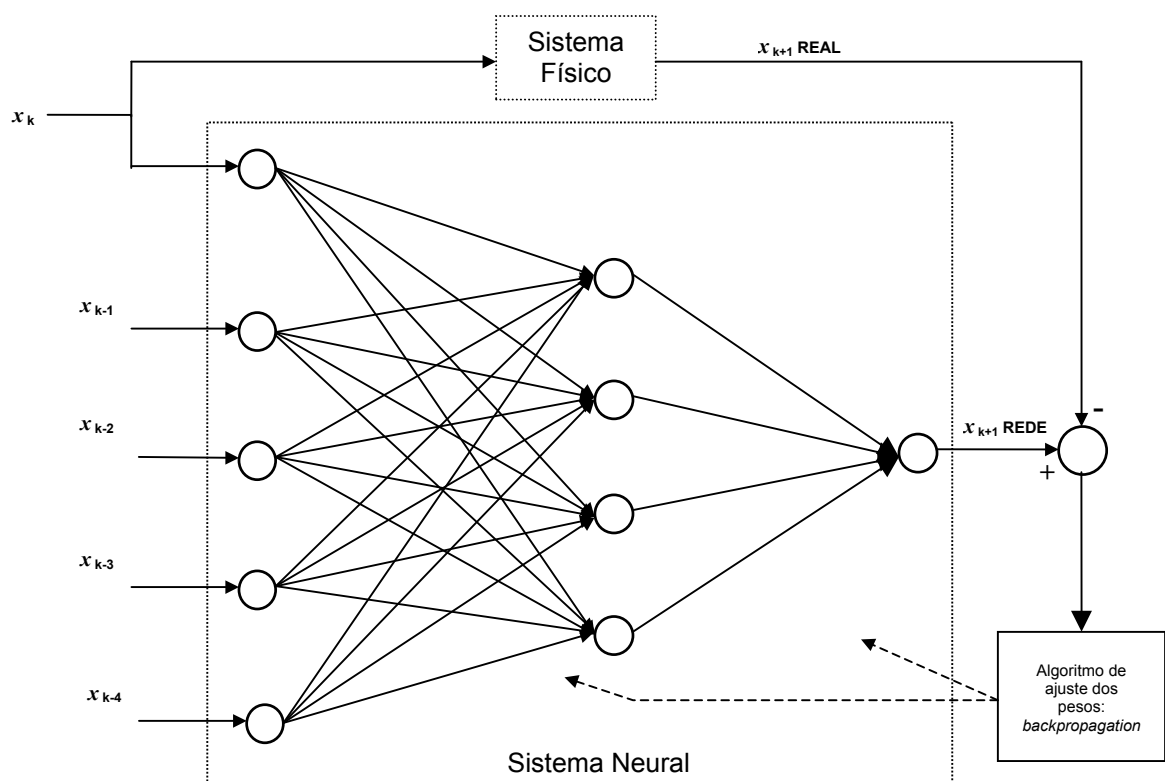


Figura 4.18 Rede neural dinâmica para da identificação de um sistema físico real

Na Figura 4.18 a rede identifica o sistema físico a partir de entradas atrasadas, no caso quatro atrasos. A quantidade de atrasos necessários está diretamente relacionada com a ordem e a complexidade do sistema que se deseja identificar. O atraso no tempo fornece informações anteriores ($x_{k-1}, x_{k-2}, \dots$), ou seja, esses atrasos são significantes justamente para que a rede possa prever a resposta do sistema em um instante de tempo futuro (x_{k+1}). Então se x_k é um deslocamento, logo x_{k-1} tem informações sobre a velocidade, x_{k-2} sobre a aceleração etc.

Capítulo V

Fundamentos dos Algoritmos Genéticos

Os algoritmos genéticos são métodos matemáticos pseudo-aleatórios de otimização. O nome atribuído a esta técnica deve-se ao fato de que os algoritmos genéticos, no seu mecanismo de busca, imitam as leis de Darwin e Mendel, que tratam sobre a evolução das espécies encontrada na natureza.

Holland (1975,1992) propôs e formulou os algoritmos genéticos. A idéia é armazenar todas as variáveis de um problema em um determinado código, de tal forma que esse código represente tanto um indivíduo do algoritmo como uma possível solução para o problema. Após sucessivas gerações, a evolução resulta em uma população vencedora, com o conjunto ótimo de indivíduos. A formulação dessa evolução configura os algoritmos genéticos.

Neste capítulo é feita uma descrição dos parâmetros utilizados para implementação dos algoritmos genéticos, ou seja, a maneira pela qual um indivíduo é codificado, selecionado, reproduzido, cruzado, mutado e extinto. No término do capítulo são apresentadas algumas estratégias para o uso dos algoritmos genéticos na otimização de parâmetros de controladores fuzzy.

5.1 Codificação de um indivíduo

O primeiro parâmetro em um projeto de otimização, via algoritmos genéticos, é a escolha da codificação de um indivíduo, que intrinsecamente significa uma possível solução do problema de otimização. Assim, os algoritmos genéticos não trabalham diretamente com as variáveis de projeto e sim com a codificação dessas variáveis.

O método clássico proposto por Holland (1975) utiliza a codificação binária para representar os valores das variáveis de projeto, porém existem outras formas de codificação como números com pontos flutuantes, números inteiros, escala de cinza, letras do alfabeto, etc. Veremos a seguir algumas dessas codificações.

5.1.1 Codificação binária

O indivíduo, neste caso, é representado por uma cadeia de bits 0 ou 1, que recebe o nome de cromossomo, conforme esquematizado na Figura 5.1. No código do cromossomo as variáveis x_i de um problema são alocadas em pequenas estruturas chamadas genes. Cada gene é formado por alelos que são a menor parte formadora do cromossomo, no caso os bits 0 ou 1.

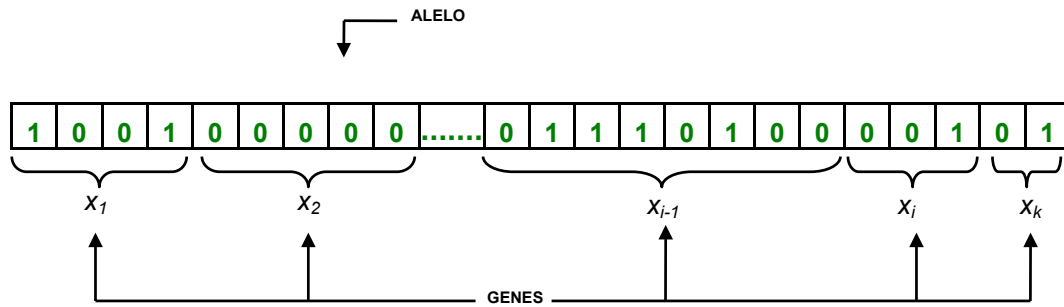


Figura 5.1 Cromossomo de um indivíduo representado por uma cadeia binária

É interessante atribuir ao cromossomo todas as variáveis que se deseja otimizar. A má codificação do indivíduo pode desencadear buscas inadequadas, ou seja, ou no cromossomo pode estar faltando um gene importante ou o tamanho dos genes estão incorretos.

Os tamanhos dos genes são determinados pela precisão que se deseja atingir na resolução do problema. Por exemplo, suponha o problema em que se deseja minimizar uma função ff com k variáveis com uma precisão de seis casas decimais:

$$\text{minimizar}(ff) = \text{maximizar}(-ff) \quad (5.1a)$$

$$ff(x_1, x_2 \dots x_k) : \mathbb{R}^k \rightarrow \mathbb{R} \quad (5.1b)$$

onde cada variável x_i pode assumir valores dentro do domínio:

$$\mathcal{D}_i = [a_i, b_i] \subseteq \mathbb{R} \quad (5.2a)$$

e que:

$$ff(x_1, x_2 \dots x_k) > 0 \text{ para todo } x_i \in \mathcal{D}_i \quad (5.2b)$$

É evidente que para atingir a precisão desejada, o domínio $\mathcal{D}_i$ tem que ser dividido por $(b_i - a_i)10^6$.

Denota-se por m_i o menor inteiro que satisfaça a seguinte inequação:

$$(b_i - a_i) \cdot 10^6 \leq 2^{m_i} - 1 \quad (5.3)$$

onde m_i é o tamanho da cadeia binária de cada variável x_i .

O somatório de todos os tamanhos m_i das variáveis x_i de um problema define a quantidade M de alelos no cromossomo de um indivíduo, e é dada por:

$$M = \sum_{i=1}^k m_i \quad (5.4)$$

onde m_i é o número de bits para mapear a variável x_i com as restrições laterais definidas pelo domínio $\mathcal{D}_i = [a_i, b_i]$.

Os indivíduos codificados são utilizados no desenvolvimento das operações de reprodução, cruzamento e mutação. Quando decodificado, o indivíduo é utilizado para avaliação das variáveis do problema, ou seja, saber o quão apto é o indivíduo codificado. Portanto são necessárias algumas transformações dos códigos de base. Por exemplo, suponha que um gene de tamanho m_i necessite ser convertido da base binária para o seu respectivo valor na base real dentro do domínio $\mathcal{D}_i = [a_i, b_i]$. Inicialmente é feita a transformação da base binária para base decimal, da seguinte forma:

$$decimal(bit_n, bit_{n-1}, \dots, bit_2, bit_1, bit_0)_2 = \left(\sum_{i=0}^n bit_i \cdot 2^i \right)_{10} \quad (5.5)$$

onde n é o número de alelos do gene. Por exemplo, seja uma variável x_i de projeto representada por um gene com 13 alelos, 0000001010010. Aplicando a equação 5.5, tem-se:

$$decimal(0000001010010)_2 = \left(\sum_{i=0}^{13} bit_i \cdot 2^i \right)_{10} = 0 \cdot 2^0 + 1 \cdot 2^1 + 1 \cdot 2^5 + 1 \cdot 2^7 = 161 \quad (5.6)$$

É necessário saber quanto o valor decimal corresponde ao valor real do gene ou a da variável x_i definida no domínio $\mathcal{D}_i$. A transformação do valor decimal para o real definido em um domínio $\mathcal{D}_i$, é dado por :

$$x_i = a_i + decimal(100011...001)_2 \cdot \frac{b_i - a_i}{2^{mi} - 1} \quad (5.7)$$

Retornando ao exemplo, suponha que o gene de 13 alelos, $(0000001010010)_2 = (161)_{10}$, representa uma variável de projeto definida no domínio real $[-5,5]$. Então, quando essa variável for utilizada na avaliação do problema, o seu valor real v_r com a precisão de 3 casas decimais é obtido aplicando a equação 5.7:

$$v_r = -5 + decimal(0000001010010)_2 \cdot \frac{5 - (-5)}{2^{13} - 1} = -4,803 \quad (5.8)$$

O alfabeto binário oferece grandes possibilidades de codificar diferentes parâmetros, porém esta representação, tradicionalmente utilizada em algoritmos genéticos, possui uma desvantagem quando se deseja obter soluções de problemas multivariáveis com alta precisão numérica. Por exemplo, para 50 variáveis definidas no domínio de $[-1000,1000]$, com a precisão de seis dígitos, o tamanho para o cromossomo, segundo as equações 5.3 e 5.4, é de 1500 bits. Nestas condições, em que o espaço de busca se torna complexo, a performance dos algoritmos genéticos é baixa quando se escolhe a codificação binária.

5.1.2 Codificação por ponto flutuante

A representação de indivíduos por ponto flutuante, isto é, por valores reais foi desenvolvida por Michalewicz (1994).

Nesta abordagem, o indivíduo apresenta um cromossomo na forma de um vetor χ_i de números reais r_i definidos nos domínios $\mathcal{D}_i = [a_i, b_i]$:

$$\chi_i = \{ r_1, r_2, r_3, \dots, r_i, \dots, r_k \} \quad (5.9)$$

onde K é a quantidade de variáveis do projeto de otimização. Logo, neste caso, para avaliar um indivíduo, as variáveis não sofrem mudança de base, ou seja, a avaliação é direta.

5.2 População, geração e extinção em algoritmos genéticos

Para resolução de um problema de otimização utilizando algoritmos genéticos, é necessária a simulação de uma evolução entre os indivíduos. Para tanto define-se uma população (P) de indivíduos, conforme a equação:

$$P(pop) = \{\chi_1^G, \chi_2^G, \dots, \chi_i^G, \dots, \chi_{pop}^G\} \quad (5.10)$$

onde pop é o tamanho da população e G é a geração na qual a população está inserida.

Todos os indivíduos χ_i da população inicial dos algoritmos genéticos são gerados aleatoriamente e cada um pode representar uma solução para o problema.

As gerações futuras são obtidas a partir da geração inicial pela aplicação de quatro operadores genéticos: seleção natural, reprodução, cruzamento e mutação, conforme ilustra Figura 5.2. Vejamos no que consiste cada um destes operadores:

- (i) Seleção natural: consiste em avaliar todos os indivíduos da população inicial, ou seja, saber o quão importante os indivíduos são na solução do problema. Os indivíduos mais aptos são selecionados para reprodução, e os demais são extintos.
- (ii) Reprodução: consiste na escolha aleatória de indivíduos sobreviventes da seleção natural, para sofrerem o processo de duplicação. Há casos em que a quantidade de indivíduos a serem reproduzidos é a mesma quantidade de indivíduos extintos na seleção natural, ou seja, o tamanho da população se mantém constante durante toda a geração. Porém é possível desenvolver gerações com populações de tamanhos diferentes de indivíduos.
- (iii) Cruzamento: consiste na escolha aleatória de pares de indivíduos sobreviventes na seleção natural juntamente com os reproduzidos, para sofrerem o processo de cruzamento, ou seja, os pares de indivíduos são entrelaçados para gerar os filhos. Os pais e os filhos são avaliados, onde os mais aptos sobrevivem e os demais são extintos;
- (iv) Mutação: consiste na escolha aleatória de alguns indivíduos sobreviventes do cruzamento, para sofrerem o processo de troca de um gene ou um alelo do cromossomo, mediante o tipo de codificação adotada. Os indivíduos que sofreram mutação e os mutantes são avaliados, e os mais aptos sobrevivem e os demais são extintos.

Com isso, o processo se repete através das gerações sucessivas. Para cada geração são feitas as seleções naturais, as reproduções, os cruzamentos, as mutações e as extinções. Todavia, é possível excluir algum dos operadores acima descritos. Por exemplo, podem ser eliminados a reprodução e o cruzamento.

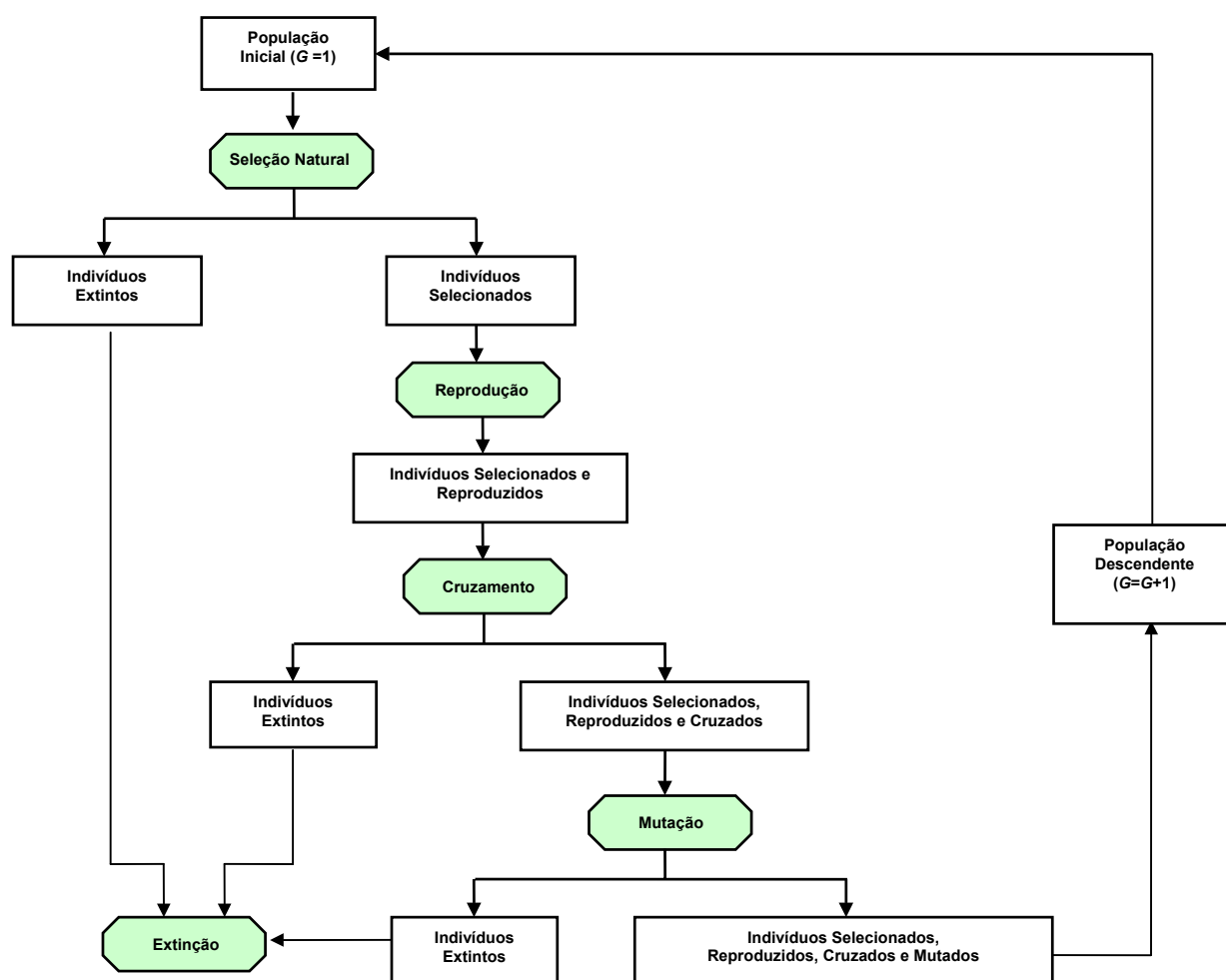


Figura 5.2 Geração inicial e os operadores genéticos.

É possível ter gerações com o tamanho da população variante, ao longo do processo de otimização. Nesta dissertação, porém, a abordagem se restringirá à população com o número de indivíduos constante.

O tamanho da população inicial é estipulado de acordo com o problema de otimização. Quanto mais indivíduos formarem a população, mais diversificada será a evolução, ou seja, serão maiores as possibilidades de encontrar-se a solução ótima em poucas gerações. Por outro lado, quando a população for pequena, as gerações se concluem com poucas modificações entre a população inicial e a população resultante do algoritmo. Então, quando a população for pequena, para atingir a solução ótima são necessárias muitas gerações.

Assim como na natureza, onde a evolução demanda geralmente milhares de anos, nos algoritmos genéticos há também um tempo computacional que é dependente dos parâmetros que definem o tamanho da população (*Pop*) e o número de gerações (*G*). Com os atuais computadores, em poucos segundos simula-se a evolução de milhares de gerações.

5.3 Processo de seleção natural

Uma vez definidos a forma de codificação do indivíduo, o número de indivíduos na população e a quantidade de gerações, é necessário definir o processo de seleção dos indivíduos para reprodução a cada geração, através da avaliação de uma função que qualifica o indivíduo, denominada de função avaliação ou função custo. Os biólogos utilizam a terminologia indivíduo com alto *fitness* para designarem os indivíduos que maximizam a função de avaliação, ou seja, os que possuem maior adaptação. A seleção é probabilística baseada neste índice de aptidão do indivíduo. Existem vários métodos para o processo de seleção, os mais comuns são: o método da roleta, o método das elites, o método do ranking linear e geométrico e o método do torneio. Vejamos alguns destes métodos.

5.3.1 Método da roleta

O método da roleta, desenvolvido por Holland (1975), consiste em calcular o valor da função de avaliação ($\mathcal{F}$) para todos os indivíduos da população. Determina-se primeiramente a probabilidade ($\mathcal{P}$) do indivíduo ser escolhido para sobreviver e reproduzir, conforme a expressão:

$$\mathcal{P}_i = \frac{\mathcal{F}_i}{\sum_{j=1}^{Pop} \mathcal{F}_j} \quad (5.11)$$

onde $i = 1, 2, \dots, Pop$, representa o i -ésimo indivíduo da população.

Após o cálculo da probabilidade $\mathcal{P}_i$ de cada indivíduo é possível a construção de uma roleta, conforme a Figura 5.3.

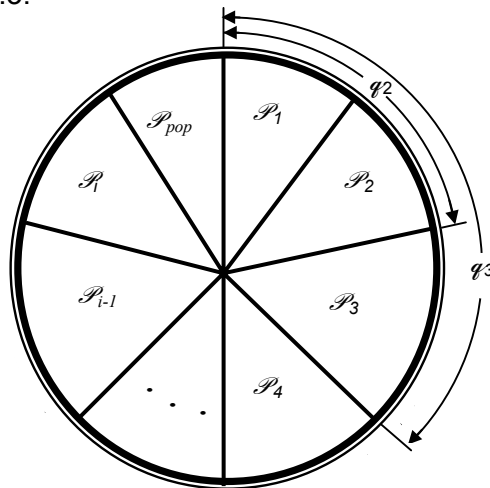


Figura 5.3 Divisão dos indivíduos em uma roleta segundo as suas respectivas probabilidades.

Os indivíduos mais aptos ocupam um espaço maior nos campos da roleta, pois sua probabilidade é maior. Para a simulação matemática de um sorteio pela roleta, calcula-se inicialmente a probabilidade cumulativa (q) de cada indivíduo:

$$q_i = \sum_{j=1}^i \mathcal{P}_j \quad (5.12)$$

O procedimento de seleção ou extinção do primeiro indivíduo parte da geração de um número aleatório $rand_1$ entre zero e a unidade. Se $rand_1 < q_1$ então o indivíduo índice 1 é selecionado para sobreviver, caso contrário, seleciona-se o i -ésimo indivíduo tal que:

$$q_{i-1} < rand_1 \leq q_i \quad (5.13)$$

Para cada seleção é gerado um novo número aleatório $rand_1$ e o processo é repetido até que atinja o número de seleções previamente estabelecido. Uma maneira prática de definir a quantidade de vezes que girar-se-á a roleta, é selecionando metade da população e extinguindo a outra metade. Assim, o número de seleções será metade do tamanho da população.

Como consequência do método da roleta, surgiu o método do elitismo que só admite indivíduos para compor a roleta com probabilidade acima da probabilidade das elites ($\mathcal{P}_e$), por exemplo, $\mathcal{P}_e = 0,80$ ou $0,90$. Os demais indivíduos são extintos.

5.3.2 Método do *ranking*

O método do *ranking* linear apenas calcula a função avaliação para todos os indivíduos e os ordena segundo um ranking. É estipulada uma probabilidade $\mathcal{R}$ que permite aos melhores indivíduos serem selecionados, por exemplo se $\mathcal{R}$ for 0,4 então 40% da população é selecionada e 60% é extinto. Desta forma, é garantido que os 40% selecionados são indivíduos que apresentam os melhores resultados na avaliação da função objetivo.

Houck *et al.* (1994) propuseram o método do ranking geométrico normalizado, que define uma probabilidade $\mathcal{P}_{Rg}$ para cada indivíduo, representando a probabilidade do indivíduo ser selecionado ou não, da seguinte forma:

$$\mathcal{P}_{Rg} = q' (1 - q)^{r-1} \quad (5.14a)$$

onde:

$$q' = \frac{q}{1 - (1 - q)^{Pop}} \quad (5.14b)$$

e q é a probabilidade adotada ao selecionar o melhor indivíduo, Pop é o tamanho da população e r é o ranking de um indivíduo. Para $r = 1$ tem-se o melhor indivíduo. Os indivíduos da população que apresentarem maiores probabilidades $\mathcal{P}_{Rg}$, são selecionados.

Outro método também utilizado na seleção de indivíduos é o método do torneio. Este método é semelhante ao ranking somente diferenciando-se por não definir probabilidades e sim realizar uma seleção randômica para escolha de pares de indivíduos. É feito um torneio entre os indivíduos que formam os pares e o vencedor é o indivíduo que apresenta melhor resultado para função avaliação. Os vencedores do torneio são selecionados e os demais são extintos.

5.4 Reprodução dos indivíduos

Durante a seleção natural, é possível que os indivíduos mais adaptados sejam selecionados mais de uma vez, devido a suas altas probabilidades, ou seja, a tendência na seleção natural é manter as melhores soluções do problema e extinguir as piores.

Se a população se mantém constante por todas as gerações, então o número de indivíduos que sofrerão reprodução é o mesmo número de indivíduos que se extinguirão na seleção natural, por exemplo, se tem 300 indivíduos na população e 50 foi extinto, então 50 indivíduos dos 250 restantes, isto é, os 250 selecionados são reproduzidos. Esses 50 indivíduos que serão reproduzidos são normalmente os melhores dentre os 250 selecionados, uma vez que na seleção, cada indivíduo é avaliado. Uma estratégia é extinguir metade da população e reproduzir a outra selecionada.

Para qualquer que seja a codificação do indivíduo, a reprodução será a duplicação do cromossomo do indivíduo.

5.5 Cruzamento dos indivíduos

Os indivíduos selecionados e reproduzidos sofrem o cruzamento aos pares e ao acaso. O procedimento de aplicação deste operador genético depende da maneira pela qual o indivíduo é codificado. Inicialmente será visto o cruzamento binário e depois dois tipos de cruzamentos para codificação com ponto flutuante: o cruzamento aritmético e o cruzamento heurístico.

5.5.1 Cruzamento binário

No cruzamento binário é necessário definir um parâmetro que define a probabilidade de cruzamento ($\mathcal{P}_c$). O procedimento para o cruzamento, parte da geração de um número aleatório $rand_2$ entre zero e a unidade para cada indivíduo. Somente os indivíduos que apresentarem a relação $rand_2 < \mathcal{P}_c$ são escolhidos para o cruzamento, os demais são extintos.

Caso for escolhida uma quantidade ímpar de indivíduos para o cruzamento, é necessário inserir um indivíduo extra ou retirar algum, uma vez que o cruzamento é feito aos pares.

Para cada par de cruzamento, é gerado pelo menos um número aleatório inteiro entre a unidade e $(M - 1)$ para definir as posições de cruzamento, onde M é o número de bits do indivíduo.

Suponha que se deseja cruzar com a probabilidade de cruzamento $\mathcal{P}_c$ os indivíduos de uma população de 10 indivíduos com 25 bits. A Figura 5.4 ilustra um possível casal escolhido para o cruzamento. Escolhe-se um número inteiro aleatório entre 1 e 24. Suponha que tenha escolhido o número 10, então nesta posição será feita a troca do material genético entre os pais, ou seja, os dois cromossomos dos pais serão divididos em duas cadeias, a esquerda do 1º ao 9º alelo e outra à direita do 10º ao 25º alelo. O cruzamento é a troca dessas divisões, o lado direito do pai é unido ao lado esquerdo da mãe e vice-versa. Os indivíduos descendentes são também avaliados pela função de avaliação. Diante desta nova avaliação, ou os pais ou os filhos são extintos, para manter o tamanho da população.



Figura 5.4 Esquema utilizado para realização dos cruzamentos de dois indivíduos

5.5.2 Cruzamento aritmético

Sejam dois vetores X_i e Y_i de números reais definidos nos domínios $\mathcal{D}_i = [a_i, b_i]$, que representam um casal de indivíduos da população selecionada e reproduzida.

O cruzamento aritmético consiste em gerar um número randômico $rand_3$, entre zero e a unidade, tal que os descendentes são assim calculados:

$$X' = rand_3 \cdot X + (1 - rand_3) \cdot Y \quad (5.15a)$$

$$Y' = rand_3 \cdot Y + (1 - rand_3) \cdot X \quad (5.15b)$$

5.5.3 Cruzamento heurístico

Seja também dois vetores X_i e Y_i de números reais definidos nos domínios $\mathcal{D}_i = [a_i, b_i]$, representando um casal de indivíduos da população selecionada e reproduzida.

O cruzamento heurístico produz uma extrapolação linear entre dois indivíduos. Um novo indivíduo é criado utilizando a equação 5.16, onde $rand_4$ é um número randômico gerado entre zero e a unidade. Para realizar o cruzamento é necessário que X_i apresente uma maior adaptação quando comparado com Y_i .

$$X_i' = X_i + rand_4 \cdot (X_i - Y_i) \quad (5.16)$$

Caso X_i' resultante estiver na região não factível ou viável, isto é, $X_i' \notin [a_i, b_i]$, então um novo $rand_4$ é gerado e o processo do cruzamento é repetido. Um critério de parada pode ser estabelecido caso o X_i' resultante estiver sempre na região não factível, neste caso faz-se o descendente igual a um dos pais, por exemplo:

$$Y_i' = X_i \quad (5.17)$$

5.6 Mutação dos indivíduos

Os indivíduos selecionados, reproduzidos e cruzados estão aptos a sofrerem mutação, o último operador genético de uma geração. A mutação também depende da maneira pela qual o indivíduo é codificado. Inicialmente será vista a mutação binária e depois três tipos de mutações para codificação com ponto flutuante: a mutação uniforme, a mutação de fronteira e a mutação não uniforme.

5.6.1 Mutação binária

Feito os cruzamentos binários e obtido a aptidão dos descendentes, extinguindo-os ou não, aplica-se o operador mutação para configurar a geração. Sabemos que o número de alelos em uma população é o produto do número de indivíduos pelo número de alelos em cada indivíduo. A mutação binária consiste na troca de um ou mais alelos da população e ocorre segundo uma probabilidade de mutação $\mathcal{P}_m$. O procedimento para a mutação parte da geração de um número aleatório $rand_5$ entre zero e a unidade para cada alelo da população. Somente os alelos que apresentarem a relação $rand_5 < \mathcal{P}_m$ são escolhidos para sofrerem a mutação.

Retornemos ao exemplo visto no cruzamento binário com uma população de 250 alelos, ou seja, os 10 indivíduos com 25 alelos cada um. Suponha que para uma determinada probabilidade de mutação $\mathcal{P}_m$ dois alelos, o 4º e o 22º, de um mesmo indivíduo tenham sido escolhidos para fazer mutação em dois de seus alelos. Assim, para efetuar a mutação, basta trocar esses alelos, se for 1 troca-se por 0 e vice e versa, conforme mostra a Figura 5.5.

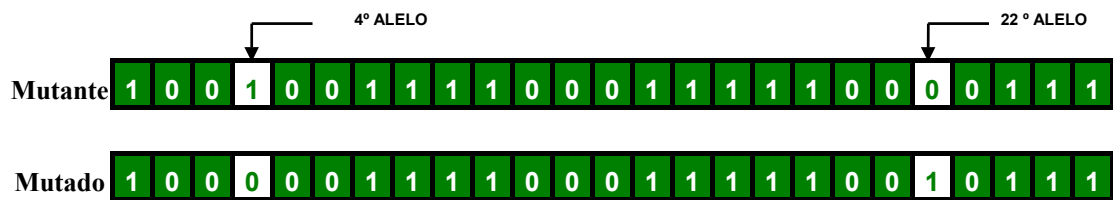


Figura 5.5 Esquema utilizado para realização das mutações dos indivíduos

5.6.2 Mutação uniforme

A mutação uniforme, que é aplicada à codificação com ponto flutuante, consiste em escolher um número inteiro randômico $rand_6$, entre a unidade e M , e substituir o indivíduo X_i com índice $i = rand_6$, tal que o indivíduo mutante fique definido conforme equação:

$$X_i' = \begin{cases} rand[a_i, b_i], & \text{se } i = rand_6 \\ X_i, & \text{caso contrário.} \end{cases} \quad (5.18)$$

onde o $rand[a_i, b_i]$ é o número randômico, dentro dos universos $[a_i, b_i]$, que fará a mutação de um indivíduo.

5.6.3 Mutação de fronteira

Outra mutação para ponto flutuante é a mutação de fronteira, que consiste em escolher um número inteiro randômico $rand_7$, entre a unidade e M , e substituir o indivíduo X_i , com índice $i = rand_7$, pelos limites inferiores e superiores, de acordo com outro número randômico $rand_8$, entre zero e a unidade, tal que o indivíduo mutante fique assim definido:

$$X_i' = \begin{cases} a_i, & \text{se } i = rand_7 \text{ e } rand_8 < 0,5 \\ b_i, & \text{se } i = rand_7 \text{ e } rand_8 \geq 0,5 \\ X_i, & \text{caso contrário.} \end{cases} \quad (5.19)$$

5.6.4 Mutação não uniforme

A mutação não uniforme consiste em escolher um número inteiro randômico $rand_9$, entre a unidade e M , e substituir o indivíduo X_i com índice $i = rand_9$, através do cálculo não uniforme que leva em conta dois números randômicos $rand_{10}$ e $rand_{11}$, escolhidos entre zero e a unidade, conforme as equações:

$$X_i' = \begin{cases} X_i + (b_i - X_i).h(G_C), & \text{se } i = rand_9 \text{ e } rand_{10} < 0,5 \\ X_i - (X_i - a_i).h(G_C), & \text{se } i = rand_9 \text{ e } rand_{10} \geq 0,5 \\ X_i, & \text{caso contrário.} \end{cases} \quad (5.20a)$$

onde,

$$h(G_C) = \left[rand_{11} \left(1 - \frac{G_C}{G_{MAX}} \right) \right]^b \quad (5.20b)$$

e G_C é o número da geração corrente, G_{MAX} é o número da máxima geração da evolução e o parâmetro b é um potencial. Uma extensão dessa mutação para codificação com ponto flutuante é a multi-mutação não uniforme, onde aplica-se o cálculo da mutação não uniforme para todos os elementos do indivíduo.

Após a descrição das possibilidades de codificação dos algoritmos genético por números binários e com ponto flutuante, bem como os operadores genéticos: seleção natural, reprodução, cruzamento e mutação, vejamos como usar esta ferramenta de otimização no projeto de controladores fuzzy.

5.7 AG como ferramenta de otimização de parâmetros de controladores fuzzy

Os algoritmos genéticos são ferramentas largamente utilizadas nos problemas de otimização que envolvem principalmente funções não lineares e não diferenciáveis. Encontram, também, grandes aplicações tanto nas redes neurais artificiais como na lógica fuzzy.

Harp *et al.* (1989) desenvolveram uma metodologia para determinação do número de neurônios em uma rede neural com somente uma camada oculta, baseado no processo de otimização dos algoritmos genéticos. Whitely e Starkwerther (1990) utiliza-os na busca dos pesos no treinamento *backpropagation*, substituindo o método do gradiente na tentativa de evitar os mínimos locais. Moricic and Nikolov (1990) desenvolveram um otimizador genético que seleciona a arquitetura de rede neural, dados alguns parâmetros específicos do problema que se quer otimizar.

Nesta dissertação, são utilizados os algoritmos genéticos na otimização de controladores fuzzy. A eficiência dos modelos fuzzy, ou seja, a capacidade de mapeamento entre as entradas e saídas de um sistema, depende da escolha de uma série de parâmetros que definem, entre outras coisas, a natureza dos conjuntos fuzzy envolvidos, o universo de variação das grandezas de entrada e saída, a base de regras empregada, os mecanismos de inferência utilizados, etc.

Karr (1991) e Homaifar *et al.* (1995) aplicam algoritmos genéticos no projeto de controladores com o objetivo de otimizar as funções de pertinência mantendo o conjunto de regras fixo. De maneira inversa, Thrift (1991) descreve o projeto de controladores fuzzy com duas entradas e uma saída, otimizando o conjunto de regras ou as relações fuzzy e mantendo as funções de pertinência constantes, assim como nos trabalhos de Baitinger *et al.* (1993) e Túpac *et al.* (1999). Kinzel *et al.* (1994) desenvolveu uma estratégia evolutiva em dois estágios, primeiro otimiza as regras e depois as funções de pertinência.

Uma questão importante, em qualquer problema de otimização, é a formulação adequada das funções de avaliação. Vejamos, a seguir, quais as funções de avaliação normalmente empregadas nos controladores fuzzy, quando aplicado à sistemas dinâmicos.

5.7.1 Funções de avaliação para controladores fuzzy

Existem muitas maneiras de definir o índice de performance para avaliação dos indivíduos de uma população. Em sistemas dinâmicos é usual definir funções objetivo que consideram: o comportamento do sistema no transiente, o comportamento em regime estacionário, os níveis de energia envolvidas (cinética e potencial), o esforço de controle, etc..

Uma possibilidade atraente, no caso de sistemas dinâmicos, é a definição de uma “curva de resposta desejada” para o comportamento do sistema a partir de uma condição inicial dada. Seja, por exemplo, a função $\zeta_{ótimo}$ que descreve um comportamento desejável, então uma função objetivo possível seria:

$$E(t) = \int_0^T (\zeta_{AG}(t) - \zeta_{ótimo}(t))^2 dt \quad (5.21)$$

onde T representa a definição do tempo de amostragem, $\zeta_{ótimo}$ é a resposta ótima e ζ_{AG} é a resposta desejada pelo controlador em análise.

5.7.2 Otimização dos pesos das regras de sistemas fuzzy

No projeto de um controlador fuzzy uma grande dificuldade é o estabelecimento do conjunto de regras, que normalmente são construídas com base na experiência e intuição de um especialista. Estas regras relacionam as entradas com as saídas e, em sistemas não lineares, variantes no tempo e instáveis, com muitas entradas e muitas saídas, a definição do conjunto de regras se torna uma tarefa não muito trivial. Uma questão crítica é o grande número de combinações que podem ocorrer no projeto de controladores fuzzy, especialmente quando o sistema apresenta conjuntos fuzzy com muitas variáveis lingüísticas e muitas funções de pertinência.

Uma estratégia interessante de otimização de regras em controladores fuzzy pode ser construída a partir da definição de um parâmetro de ponderação de cada regra, que retrata a importância de cada uma dentro do conjunto de regras, ou seja:

$$\mathbf{R}_1 : \text{Se o erro é PM e o } \Delta\text{erro é Z então } u \text{ é NM} \quad (\mathbf{P}_1) \quad (5.22a)$$

$$\mathbf{R}_2 : \text{Se o erro é Z e o } \Delta\text{erro é PM então } u \text{ é NP} \quad (\mathbf{P}_2) \quad (5.22b)$$

$$\mathbf{R}_3 : \text{Se o erro é NG e o } \Delta\text{erro é PG então } u \text{ é PM} \quad (\mathbf{P}_3) \quad (5.22c)$$

...

$$\mathbf{R}_n : \text{Se o erro é PG e o } \Delta\text{erro é Z então } u \text{ é NG} \quad (\mathbf{P}_n) \quad (5.22d)$$

Os pesos ($\mathbf{P}_1$, $\mathbf{P}_2$, ..., $\mathbf{P}_n$) das regras ponderam o conseqüente de cada regra, por exemplo, se o resultado do antecedente da primeira regra resultar em um valor de pertinência igual a μ_1 , então o conjunto do conseqüente dessa regra será inferido no valor $\mu_1.P_1$, e assim, sucessivamente, por todas as regras.

Através dos pesos das regras é possível avaliar, após evolução do algoritmo, a importância de cada uma na eficiência do controlador. Uma forma de diminuir o tamanho da base de regras é eliminar as regras que apresentarem pesos inferiores a um determinado valor escolhido. Com esta estratégia torna-se mais rápido o cálculo da saída do controlador.

5.7.3 Otimização dos conseqüentes das regras fuzzy

Um mecanismo de otimização que pode ser usado é o que relaciona o indivíduo aos conseqüentes de uma base de regras.

Suponha uma codificação binária para representação do controlador com sete variáveis lingüísticas no conseqüente, cada uma com três bits, conforme a Figura 5.6. Com três bits são possíveis 8 combinações. Neste caso, é necessário definir um conseqüente auxiliar (*AUX*) para completar todas as possibilidades. Este conseqüente *AUX* é uma variável lingüística representada por um conjunto fuzzy vazio, de tal forma que não influencia nos valores de controle.

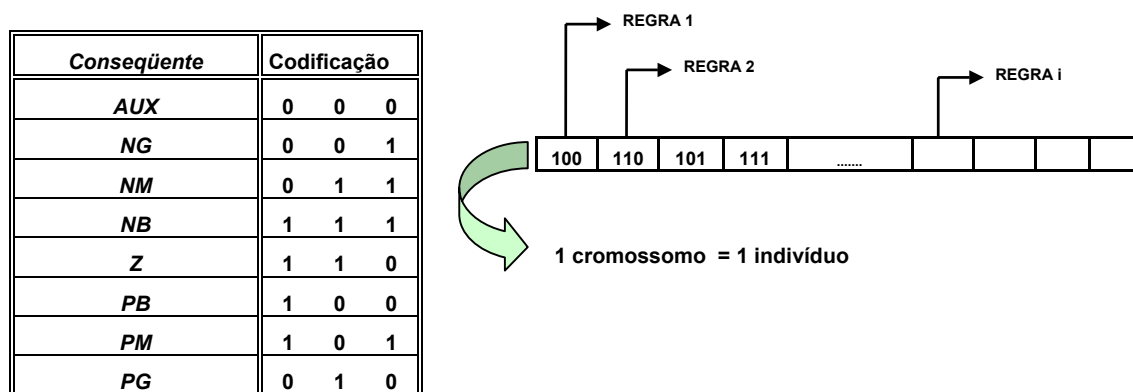


Figura 5.6 Representação binária das variáveis lingüísticas com 3 bits para configurar um cromossomo com todos os conseqüentes da base de regras do controlador fuzzy.

5.7.4 Otimização das funções de pertinência das entradas e saídas

Karr (1991) foi um dos pioneiros na aplicação dos algoritmos genéticos no objetivo de otimizar as funções de pertinência de um controlador fuzzy mantendo o conjunto de regras fixo.

O procedimento de otimizar os parâmetros de funções de pertinência pode ser assim descrito; suponha que um controlador possua um conjunto de partições fuzzy das entradas e das saídas representadas por funções de pertinência triangulares, definidas segundo a equação 5.23, com seus parâmetros indicados na Figuras 5.7.

$$\mu_{\text{triang}}(u) = \begin{cases} 0, & u \leq a \\ \frac{u-a}{c-a}, & a \leq u \leq c \\ \frac{c-u}{b-c}, & c \leq u \leq b \\ 0, & u \geq b \end{cases} \quad (5.23)$$

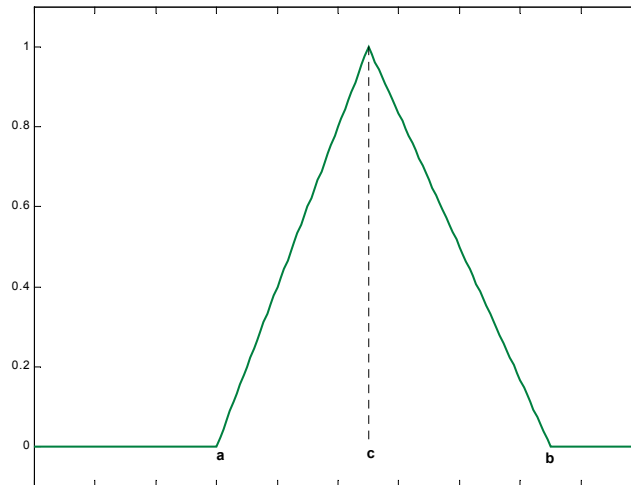


Figura 5.7 Parâmetros da função de pertinência triangular

Suponha que um controlador possua duas partições de entrada e uma partição de saída, cada qual com 7 funções de pertinência triangulares. Como cada função triangular envolve 3 parâmetros, têm-se 63 variáveis de projeto para serem codificadas. Então, um indivíduo é caracterizado por 63 parâmetros das 21 funções de pertinência triangular presentes no controlador.

5.7.5 Otimização dos universos do discurso das entradas e saídas

Outra estratégia é a otimização dos universos do discurso das variáveis do controlador fuzzy. Para esta otimização, basta codificar um indivíduo com os universos do discurso das variáveis de entrada e de saída do controlador.

Suponha um controlador com duas entradas e uma saída definidos, respectivamente, nos universos do discurso $[-e_1, e_1]$, $[-e_2, e_2]$ e $[-s_1, s_1]$. Neste caso, um indivíduo é um vetor com seis números reais. A estratégia, neste caso, é começar a busca sempre com amplos domínios, de tal forma que após a otimização, os domínios resultantes, para as variáveis de entrada e saída do controlador, sejam mais restritos.

5.7.6 Otimização simultânea de parâmetros de controladores fuzzy

Muitos pesquisadores preferem otimizar vários parâmetros dos controladores fuzzy simultaneamente. Lee e Takagi (1993) aplicaram algoritmos genéticos para otimizar tanto o conjunto de regras como as funções de pertinência de controladores fuzzy com regras do tipo Takagi-Sugeno e codificação genética tipo binária. Homaifar e McCormick (1995) otimizam, também simultaneamente, tanto o conjunto de regras como as funções de pertinência, só que de sistemas fuzzy com regras do tipo Mamdani e codificação genética do tipo ponto flutuante.

Park *et al.* (1993) utilizam codificação binária para representar o indivíduo através dos parâmetros da base de regra do controlador fuzzy com duas entradas e uma saída. O indivíduo é formado por uma parte binária correspondendo ao conjunto de regras e outra correspondendo aos parâmetros das funções de pertinências da saída, conforme ilustrado na Figura 5.8.

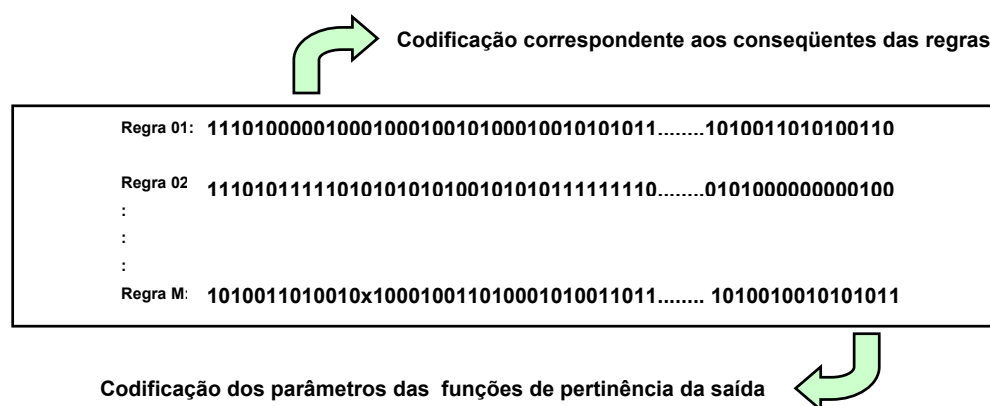


Figura 5.8 Estratégia de codificação de um indivíduo inserindo dois parâmetros do controlador fuzzy para otimização simultânea.

Mesmo diante das possibilidades em otimizar simultaneamente várias variáveis de um sistema fuzzy, Tsoukalas e Uhrig (1997) salientam que é muito mais simples e prático otimizar uma variável por vez, principalmente nos casos em que se conhece os universos do discurso de cada variável do controlador.

Geralmente, a otimização de controladores fuzzy por algoritmos genéticos envolve a codificação do indivíduo com os parâmetros do controlador fuzzy que se deseja otimizar. Definida a maneira de codificação do indivíduo, geram-se aleatoriamente indivíduos para formar a população. Aplica-se nesta população os operadores genéticos, a cada geração, até que se atinja o término da evolução. A população evoluída será formada por indivíduos ou controladores que apresentem o comportamento desejado, isto é, um controlador fuzzy ótimo.

Capítulo VI

Controle e identificação de um sistema mecânico com um grau de liberdade

Este capítulo apresenta os ensaios experimentais de identificação e controle ativo de vibrações de um sistema com um grau de liberdade. O sistema é composto por uma plataforma de alumínio montada sobre quatro lâminas metálicas flexíveis, configurando uma mesa vibratória, conforme mostrado na Figura 6.1.

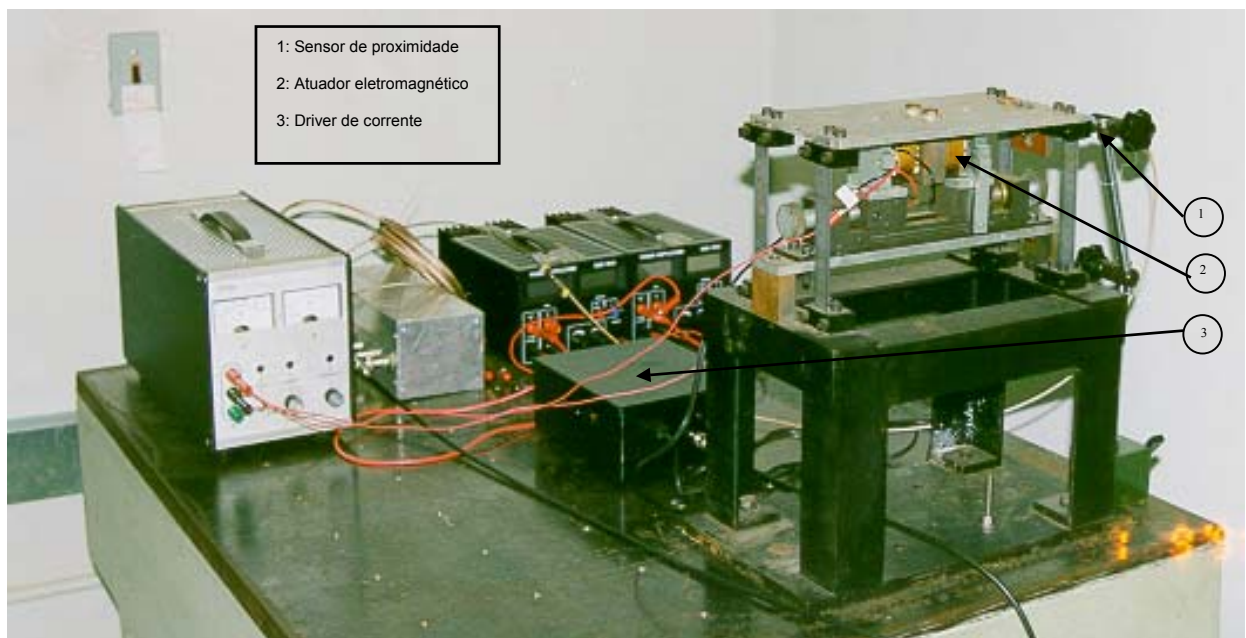


Figura 6.1 Bancada experimental do sistema mecânico vibratório de um grau de liberdade.

A mesa vibratória é instrumentada com um sensor de proximidade que gera as informações dos deslocamentos laterais. Estas informações são utilizadas por um controlador *fuzzy*, implementado num microcomputador PC. Os sinais de comando são enviados para um driver de corrente que alimenta um transdutor eletromagnético, transformando o sinal de entrada em forças eletromagnéticas de controle. Os componentes da bancada¹ são mostrados na Figura 6.2.

¹ No anexo I é apresentada a descrição dos equipamentos desta bancada experimental

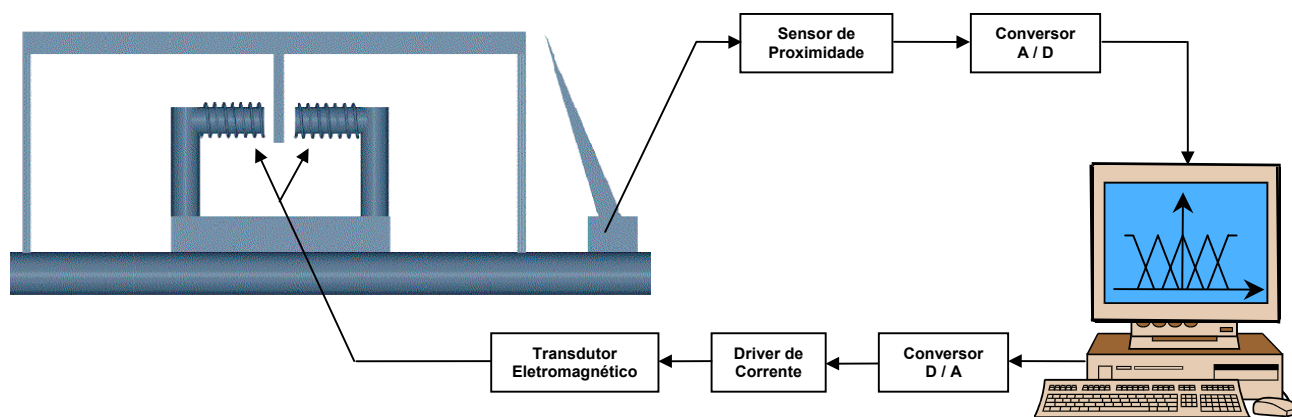


Figura 6.2 Esquema da montagem da instrumentação da bancada da mesa vibratória.

Esses ensaios experimentais estão divididos em 3 etapas. Na primeira etapa faz-se a identificação do comportamento dinâmico da mesa utilizando uma rede neural artificial, que procura modelar o comportamento do sistema em termos dos sinais de entrada e de saída. Na segunda etapa, utilizando o modelo neural identificado, procede-se à otimização numérica de alguns controladores *fuzzy* propostos. Uma vez otimizados, os controladores são implementados e avaliados experimentalmente na terceira etapa.

Como referência, na Figura 6.3 é mostrado o comportamento do sistema sem controle quando excitado por uma força de impacto. Destaca-se na Figura 6.3 o comportamento muito pouco amortecido do sistema e o elevado tempo de acomodação.

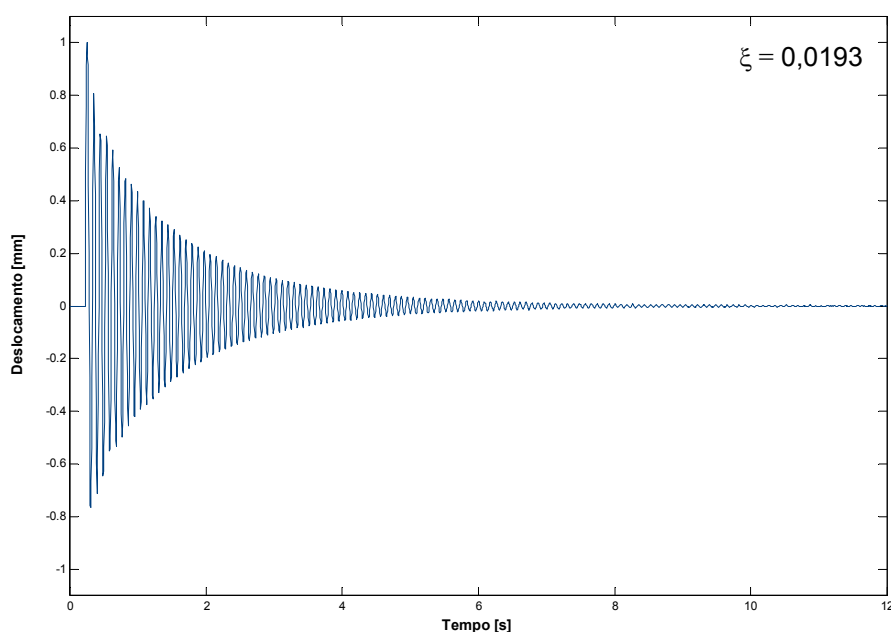


Figura 6.3 Resposta experimental ao impulso do sistema sem controle.

6.1 Identificação experimental do sistema de um grau de liberdade

Para identificação da mesa vibratória adotou-se a seguinte metodologia. Enviou-se para o driver de corrente sinais de comandos a partir do microcomputador PC, provocando vibrações no sistema. As informações dos deslocamentos da mesa foram capturadas pelo sensor que teve os seus sinais adquiridos e armazenados na memória do PC, numa taxa de amostragem de 200 Hz. Os dados de comando (entrada) e adquiridos (saída) formaram a base de informações para treinamento e validação da rede neural.

Na identificação do sistema utilizaram-se apenas informações sobre o sinal de comando [$u(k)$ $u(k-1)$ $u(k-2)$... $u(k-5)$] aplicado ao sistema, e dos deslocamentos [$x(k)$ $x(k-1)$ $x(k-2)$... $x(k-6)$], conforme diagrama da Figura 6.4.

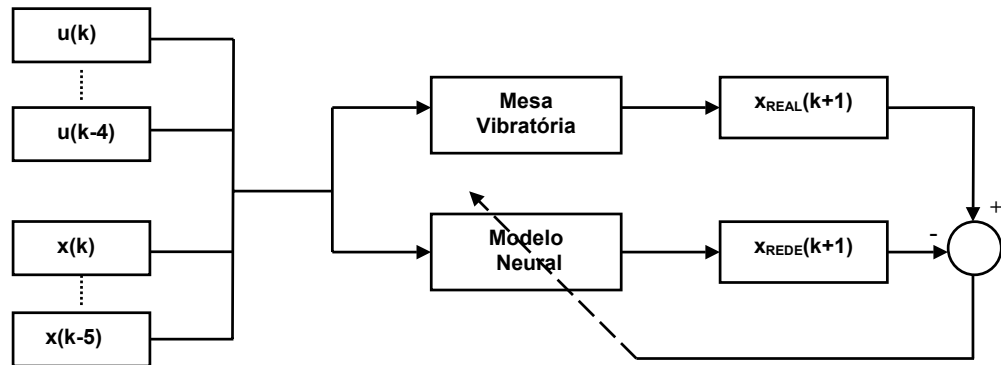


Figura 6.4 Identificação do modelo direto da planta

Como pode ser observado, a arquitetura proposta para a rede neural é semelhante às descrições clássicas baseadas em equações de diferenças, ou seja:

$$x(k) = \sum_{i=0}^N a_i \cdot x(k-i) + \sum_{j=0}^M b_j \cdot u(k-j-\Delta) \quad (6.1)$$

onde N e M são número inteiros que definem a ordem da equação de diferença, Δ é o atraso no controle e, a_i e b_i são coeficientes da equação. No caso da figura 6.4, $N=5$, $M=3$, $\Delta=0$ e os coeficientes a_i e b_i são representados pelos pesos ajustados pela rede treinada.

6.1.1 Treinamento

A entrada da rede neural, como se vê na Figura 6.4, contou com 11 neurônios na camada de entrada, 7 neurônios na camada oculta e 1 neurônio na camada de saída. A função de

ativação adotada para a camada oculta foi a tangente hiperbólica e para a saída a função linear. O algoritmo utilizado no treinamento foi o *backpropagation* e empregou-se o "toolbox" de redes neurais do software MATLAB®, versão 5.3, no treinamento da rede.

Para o treinamento, aplicou-se na mesa uma entrada de controle $u(k)$ com três amplitudes diferentes e frequência variando de 0 a 30 Hz no intervalo de 2 a 32 s. De 0 a 2 s o sinal é do sistema em repouso e de 32 a 44 s, o sinal é do sistema sem controle, conforme mostram os gráficos da Figura 6.5.

O resultado do treinamento, no que se refere à evolução do erro médio quadrático, é apresentado na Figura 6.6.

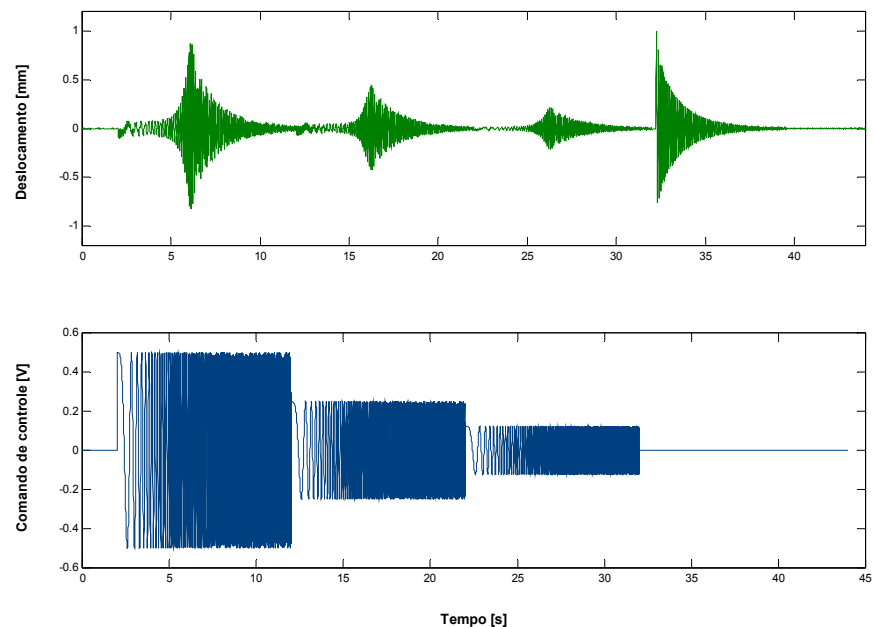


Figura 6.5 Sinais do deslocamento e da força eletromagnética variando de 0 a 30 Hz, utilizados no treinamento

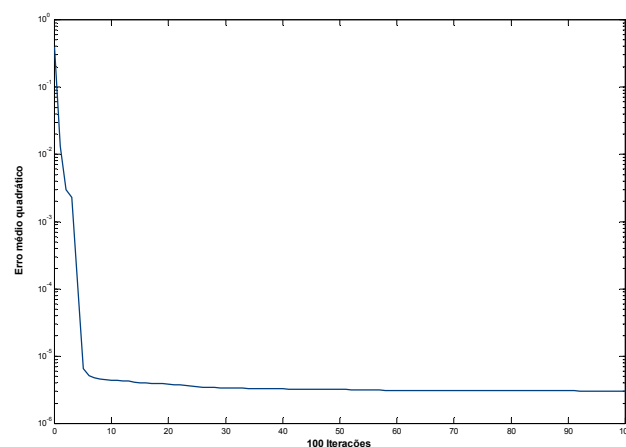


Figura 6.6 Evolução do erro médio quadrático do treinamento realizado.

6.1.2 Validação com o sistema em malha aberta

Para validação do sistema em malha aberta, gerou-se um sinal de controle $u(k)$ com três amplitudes diferentes e frequência variando de 0 a 20 Hz no intervalo de 0 a 32s. O resultado desta validação é apresentado na Figura 6.7, com uma ampliação no intervalo de tempo de 26 a 31 segundos. Não é mostrado todo o intervalo, pois os sinais não podem ser visualmente distinguidos, tal a precisão do modelo neural.

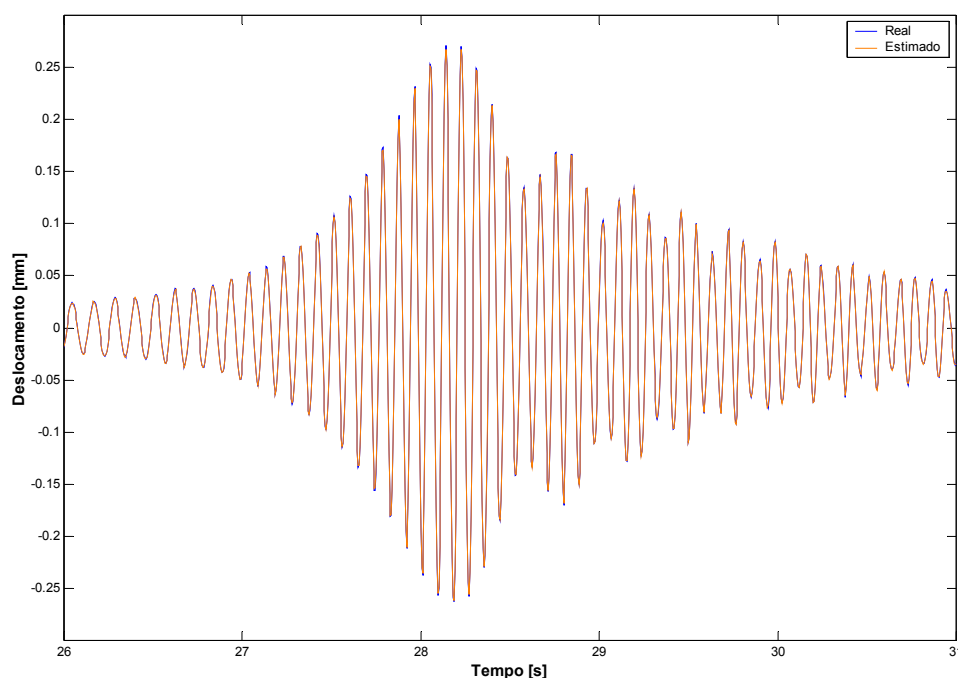


Figura 6.7 Sinais do deslocamento variando de 0 a 20 Hz utilizados na validação

6.1.3 Validação com o sistema em malha fechada

A validação em malha fechada consiste em apresentar à rede um conjunto de entradas iniciais que, uma vez propagado, gera a saída da rede. A saída gerada é incorporada a um novo vetor de entrada da rede, que propaga uma nova saída, que realimenta uma nova entrada e assim sucessivamente. Portanto, a rede simula o comportamento dinâmico do sistema de 1 g.d.l., desempenhando o papel de um integrador.

Não foi necessário, no caso deste experimento, derivar modelos neurais altamente precisos para a mesa para se obter bons controladores. Infere-se que a característica nebulosa do "*controlador fuzzy*" permite que se trabalhe com modelos razoavelmente incertos.

Para a validação do modelo neural em malha fechada, aplicou-se um sinal de comando do tipo degrau unitário à rede e comparou-se a resposta da mesma com a resposta do sistema real, quando sujeito à mesma excitação. Nessa comparação observou-se, preponderantemente, o comportamento em regime permanente, as freqüências presentes no transitório e o amortecimento do sistema. A Figura 6.8 mostra o resultado obtido.

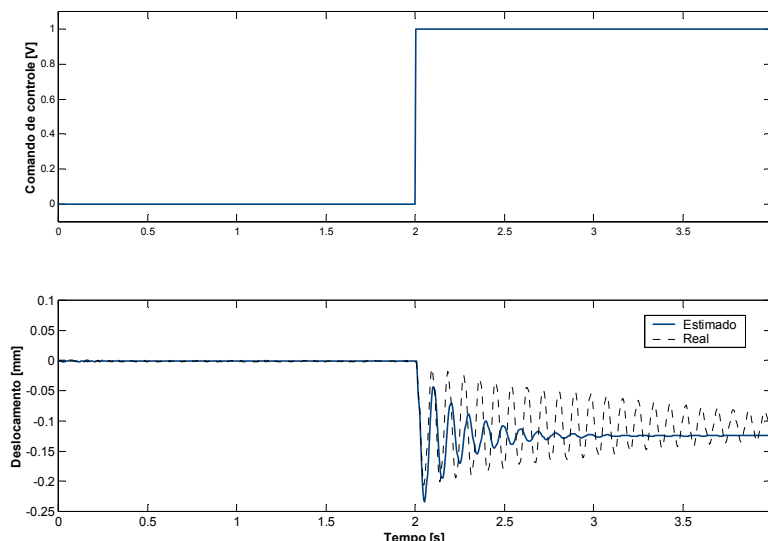


Figura 6.8 Validação do sistema neural a uma entrada degrau.

Como pode ser verificado há uma razoável concordância entre a resposta da mesa e a saída estimada pela rede neural. As duas saídas convergem aproximadamente para os mesmos valores em regime permanente, apresentam freqüências de oscilação semelhantes e diferem no tocante ao amortecimento. A rede é bem mais amortecida que o sistema real.

Outros tipos de excitações poderiam ter sido empregadas na avaliação do modelo neural da mesa, como por exemplo: entradas impulsivas, entradas do tipo rampa, etc. Uma comparação no domínio da freqüência também caberia. Estes testes não foram feitos, uma vez que a comparação, usando a resposta ao degrau unitário foi suficiente para eleger o modelo, que se mostrou adequado, como revelam os resultados que se seguem.

6.2 Otimização numérica dos controladores *fuzzy* utilizando algoritmos genéticos

Uma vez elaborado o modelo neural da mesa vibratória deve-se definir as estratégias de otimização dos controladores *fuzzy*. O diagrama de blocos com o controlador, sujeito à otimização, é mostrado na Figura 6.9.

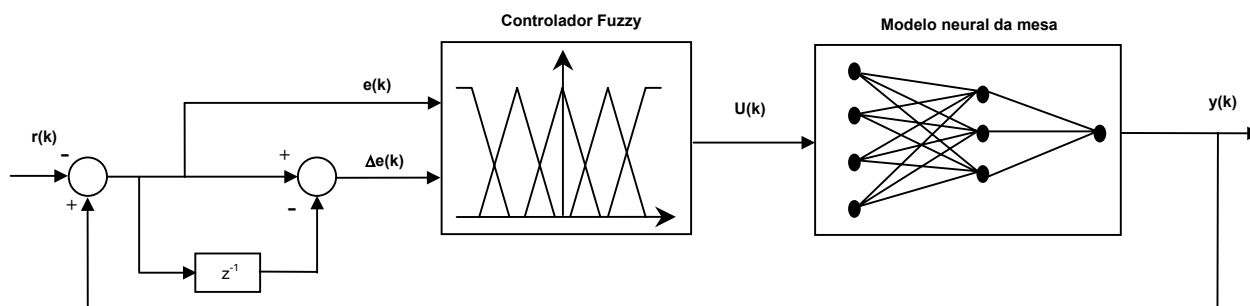


Figura 6.9 Diagrama de blocos com sistema de controle implementado.

O sinal de erro, $e(k)$, é definido como o sinal de deslocamento da mesa $y(k)$, obtido pelo sensor de proximidade, subtraído de um sinal de referência $r(k)$. Este sinal de erro, juntamente com a variação do erro, $\Delta e(k)$, são as entradas do controlador *fuzzy*. A saída do controlador *fuzzy* é uma "tensão" que alimenta o modelo neural da mesa.

6.2.1 As estratégias de otimização

A estratégia de otimização parte da escolha de um controlador fuzzy inicial. Para este controlador deve-se definir: o intervalo de variação dos sinais de entrada e do sinal de saída, o número e os parâmetros iniciais das funções de pertinência e uma base de regras completa. Com exceção do intervalo de variação dos sinais, as demais variáveis de inicialização serão objeto de otimização, não sendo portanto críticas. O procedimento de otimização começa pela busca do conseqüente da base de regras iniciais. Evolui para a otimização dos pesos das regras. Segue otimizando os parâmetros das curvas de pertinência da entrada. A partir do controlador obtido no passo anterior excluem-se as pesos inferior a um determinado valor (no caso 0,5) e otimiza-se novamente os pesos. Esta seqüência de otimizações é mostrada na Figura 6.10.

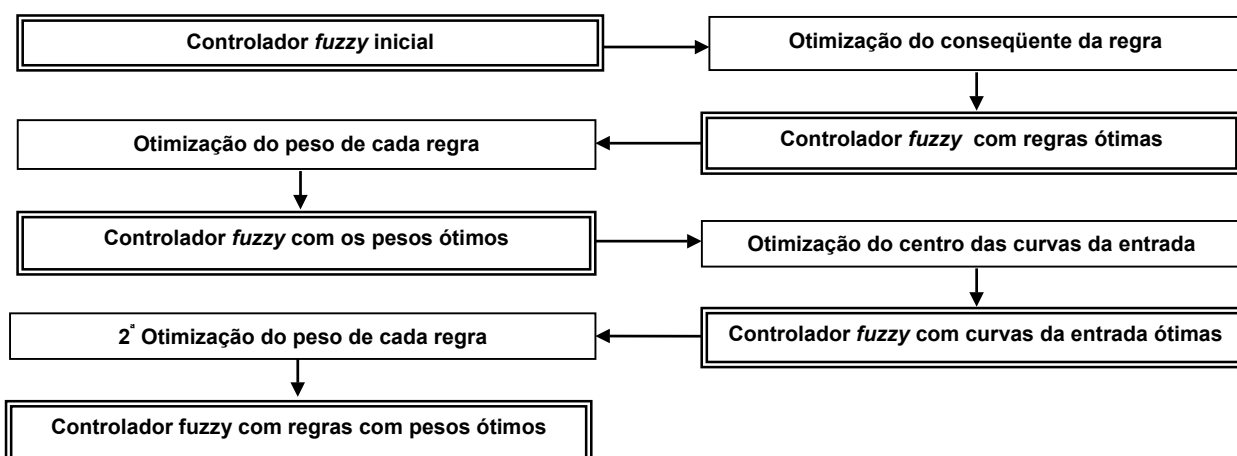


Figura 6.10 Estratégia de otimização de controladores a partir de um controlador *fuzzy* inicial.

Uma vez estabelecida a estratégia de otimização, definiu-se a seguinte função objetivo:

$$IP_{MESA} = -\int_0^t \frac{1}{2} \sqrt{[\alpha x(t).F(t)]^2 + [\beta u(t).G(t)]^2} \quad (6.2)$$

onde t é o tempo de simulação adotado (1 segundo), $X(t)$ é a resposta do modelo neural, $U(t)$ é o valor de comando de controle obtido para um determinado controlador ou indivíduo do algoritmo, $F(t)$ e $G(t)$ são funções do tempo que ponderam a resposta e o controle, α e β são parâmetros constantes. No caso deste experimento $\alpha=\beta=1$ e $F(t)=G(t)$ é ilustrada na Figura 6.11.

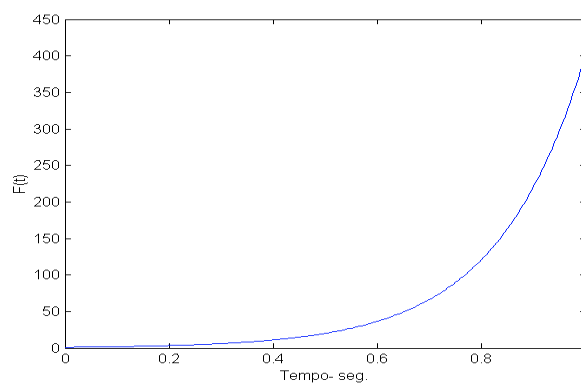


Figura 6.11 Função auxiliar $F(t)$ para cálculo da função objetivo.

No processo de otimização por algoritmo genético optou-se por uma formulação onde os indivíduos são codificados em ponto flutuante, a seleção natural se fez por torneio, não houve cruzamentos, apenas mutações entre o indivíduos. Nos casos estudados, a reprodução dos indivíduos não trouxe benefícios importantes ao processo de otimização, tornando-o inclusive mais lento.

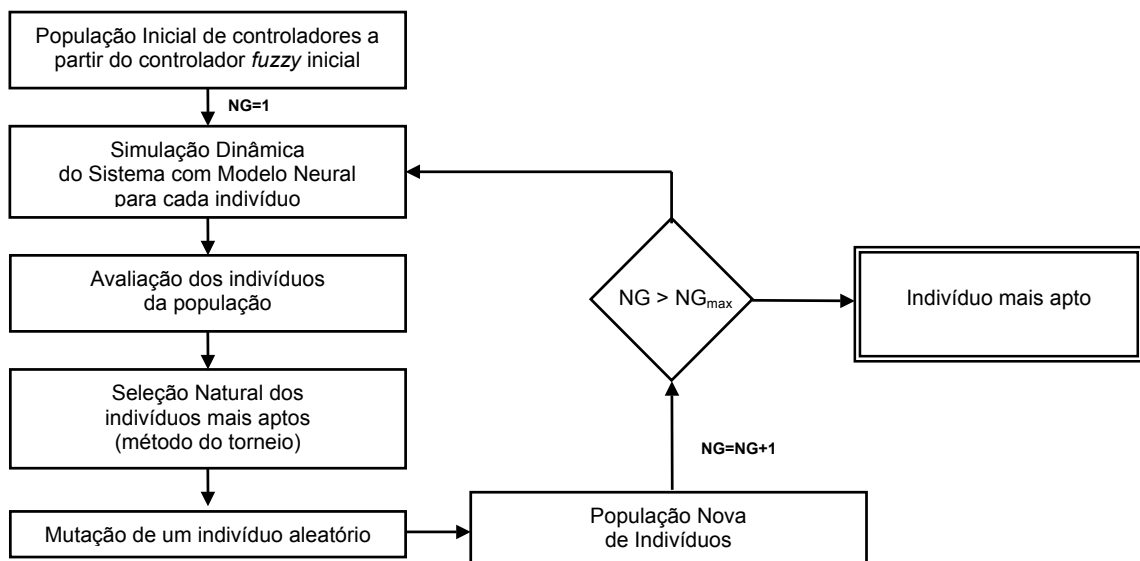


Figura 6.12 Fluxograma do algoritmo genético utilizado na otimização de controladores *fuzzy*.

6.2.2 Controlador Inicial

Escolheu-se um controlador *fuzzy* para o início da otimização com funções gaussianas igualmente espaçadas nos universos do discurso. As regras do controlador *fuzzy* inicial são mostradas na Tabela 6.1. O valor entre parênteses se refere ao peso da regra. Escolheu-se tanto para variáveis lingüísticas da entrada $e(k)$ e $\Delta e(k)$, como para a saída de controle $V(k)$ três curvas de pertinência: negativo (N), zero (Z), positivo (P), conforme a Figura 6.13.

Tabela 6.1 Conjunto de regras do controlador *fuzzy* inicial.

$\Delta e(k)$ \ Erro	Erro		
	N	Z	P
N	Z(1)	Z(1)	Z(1)
Z	Z(1)	Z(1)	Z(1)
P	Z(1)	Z(1)	Z(1)

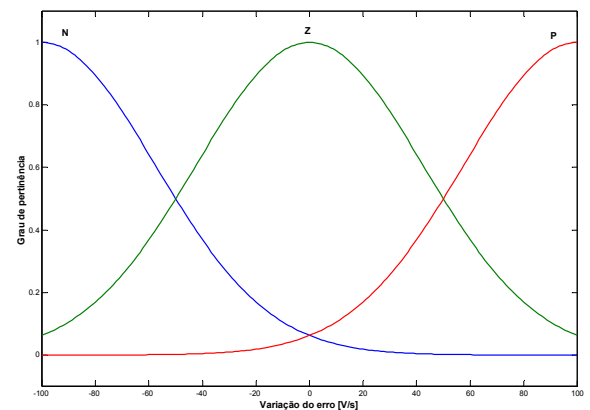
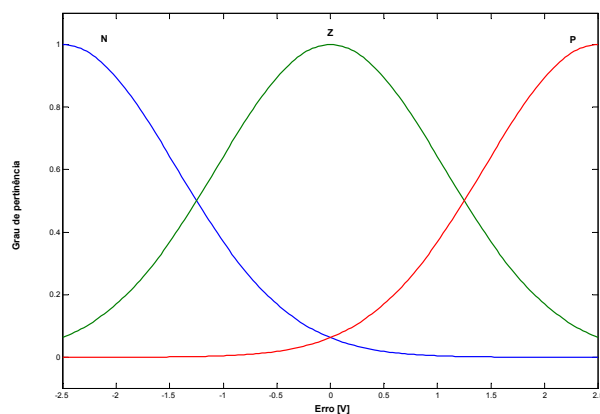


Figura 6.13 Funções de pertinência no universo do discurso do erro e da variação do erro, variáveis de entrada do controlador *fuzzy* inicial.

6.2.3 Resultados numéricos de otimização

A Tabela 6.2 resume os controladores obtidos ao longo do processo de otimização e mostra também o cálculo do coeficiente de amortecimento, obtidos a partir dos dados experimentais.

Os coeficientes de amortecimento de cada componente harmônico do sinal são obtidos através da decomposição em Wavelets. As funções Wavelets utilizadas possuem como parâmetros a frequência, o coeficiente de amortecimento e a fase. Através de um procedimento de otimização, que maximiza o índice de correlação entre a função Wavelets e o sinal, decompõe-se o sinal em uma série de funções Wavelets e os coeficientes de amortecimento são obtidos diretamente do parâmetro correspondente da função Wavelets (Santos,1999).

Tabela 6.2 Controladores *fuzzy* obtido

Controlador	Estratégia de otimização	ξ
M-livre	-	0,0193
M1	Conseqüente da regras	0,1375
M2	Pesos (base de regras completa)	0,2510
M3	Funções de pertinência das entradas (centro)	0,2895
M4	Pesos (base de regras reduzida a 6)	0,2506

Dada a semelhança dos resultados apresenta-se a seguir apenas o resultado numérico do controlador M1.

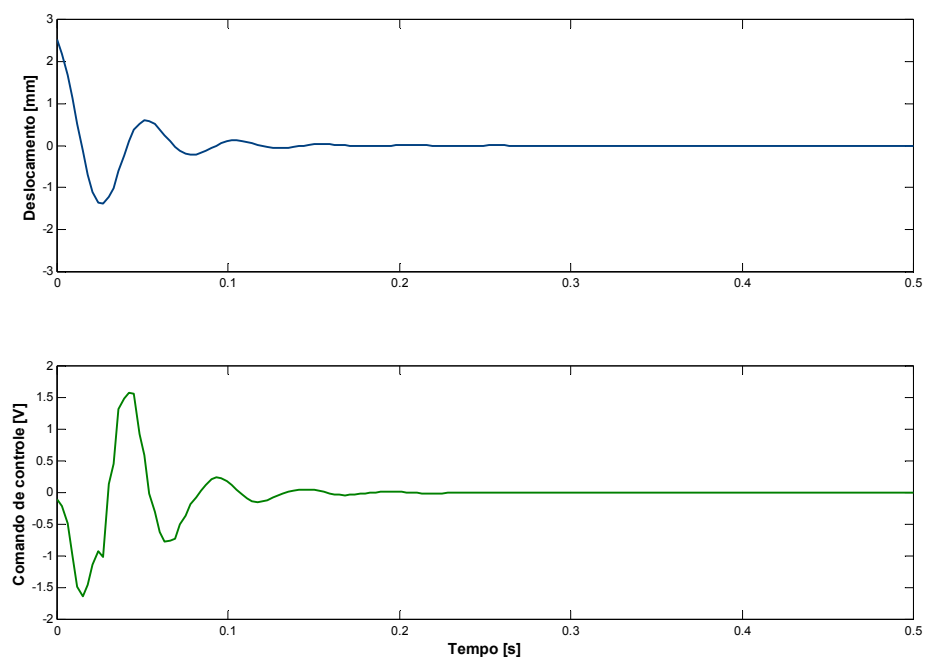


Figura 6.14 Deslocamento lateral e comando de controle para o sistema simulado com a rede neural treinada e controlado pelo controlador **M1**.

Como pode ser observado na Figura 6.14, o controlador diminui acentuadamente a duração do transitório, aumentando de forma significativa o amortecimento do sistema.

6.3 Resultados dos testes experimentais no domínio do tempo

Todos os controladores foram avaliados, e propiciaram resultados bem semelhantes. Vamos nos ater, neste momento, a apresentar apenas o controlador **M3**.

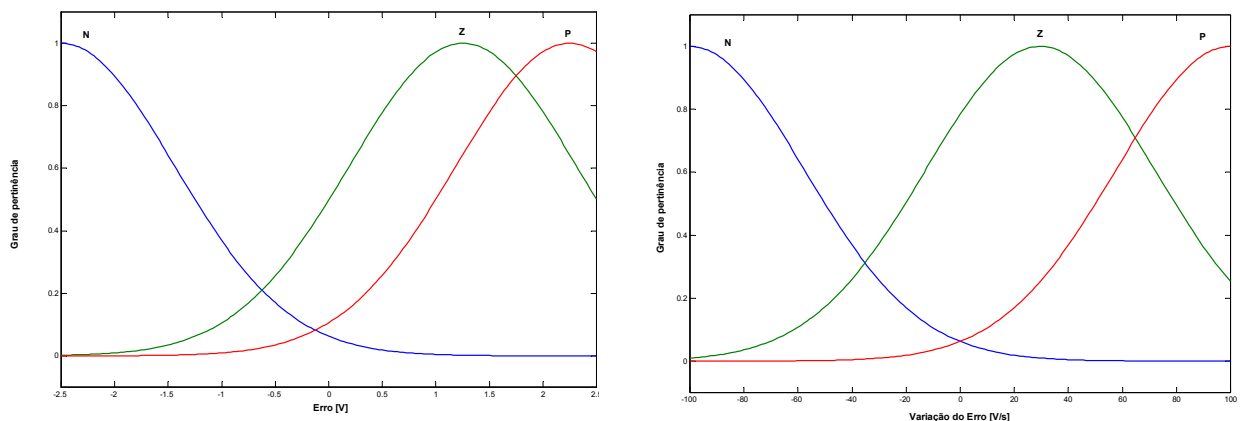
Para avaliação, o sistema foi excitado em dois instantes ($t=0s$ e $t=2,5s$) por um comando senoidal, na frequência natural do sistema durante meio segundo.

O controlador **M3**, otimizou as funções de pertinência das entradas. A base de regras de **M3** foi derivada a partir das otimizações **M1** e **M2**, respectivamente. A Tabela 6.3 mostra a base de regras do controlador.

Tabela 6.3 Conjunto de regras do controlador *fuzzy* **M3** com as regra e pesos otimizados.

Δ erro \ Erro	N	Z	P
N	Z(0,61)	N(0,94)	Z(0,01)
Z	N(0,52)	Z(0,95)	Z(0,85)
P	Z(0,05)	P(1,00)	P(0,46)

Os centros das curvas de pertinência das entradas foram modificados após a otimização, conforme mostra a Figuras 6.15.



O comportamento no tempo do controlador **M3** é apresentado nas Figuras 6.16-17.

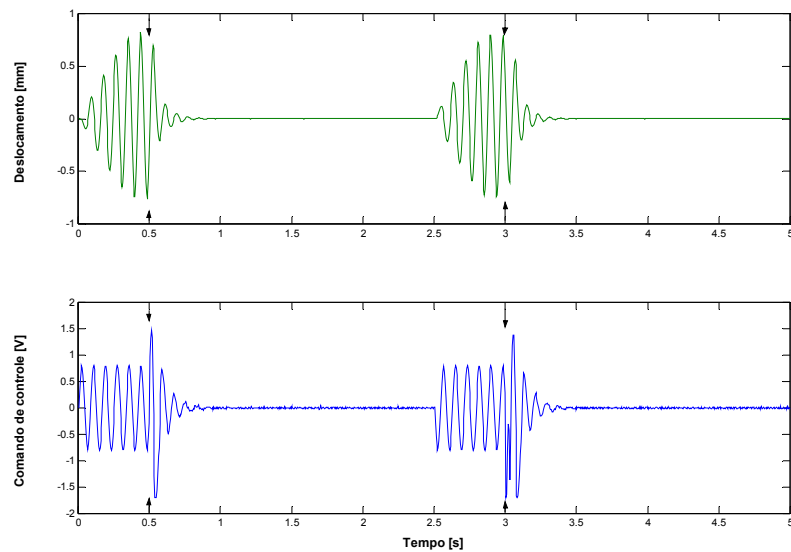


Figura 6.16 Deslocamento lateral e comando de controle para o sistema em malha fechada, com o controlador **M3** ativado em $t=0,5$ segundos.

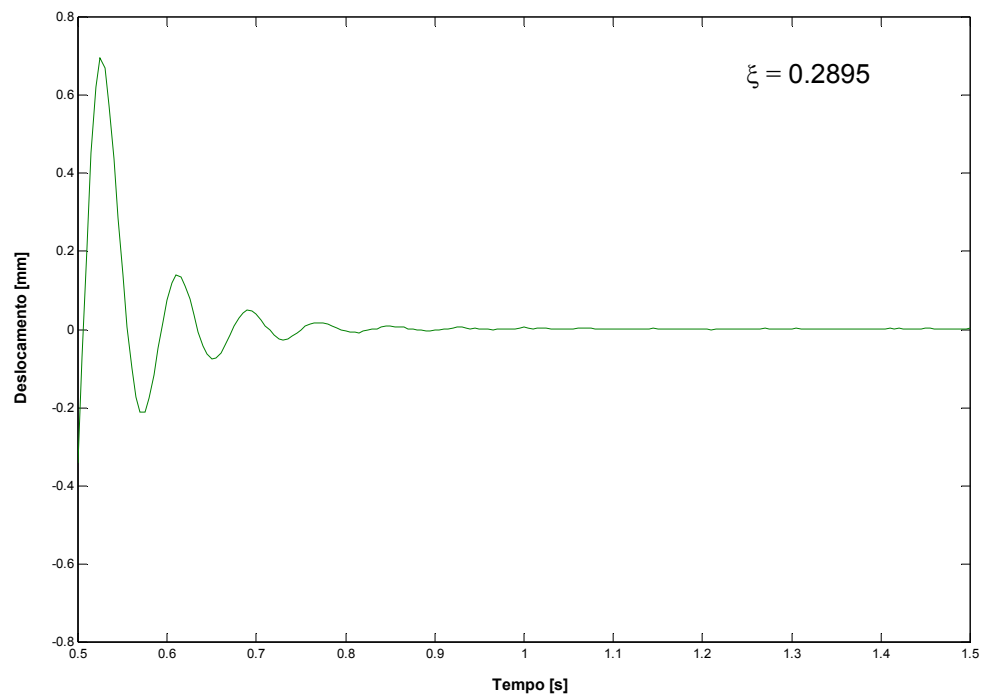


Figura 6.17 Deslocamento lateral com o controlador **M3** no intervalo de tempo de 0,5 a 1,5 segundos.

Como se pode observar o controlador projetado funcionou muito bem, introduzindo um amortecimento no sistema e reduzindo o tempo de acomodação. Verificou-se também uma modificação importante na forma das funções de pertinência, como mostrado na Figura 6.15.

6.4 Caracterização do controle no domínio da frequência

Com a finalidade de investigar o comportamento do controlador no domínio da frequência foram realizados um conjunto de ensaios baseados na instrumentação mostrada esquematicamente na Figura 6.18. A descrição completa das condições de realização dos ensaios é apresentada no Apêndice I.

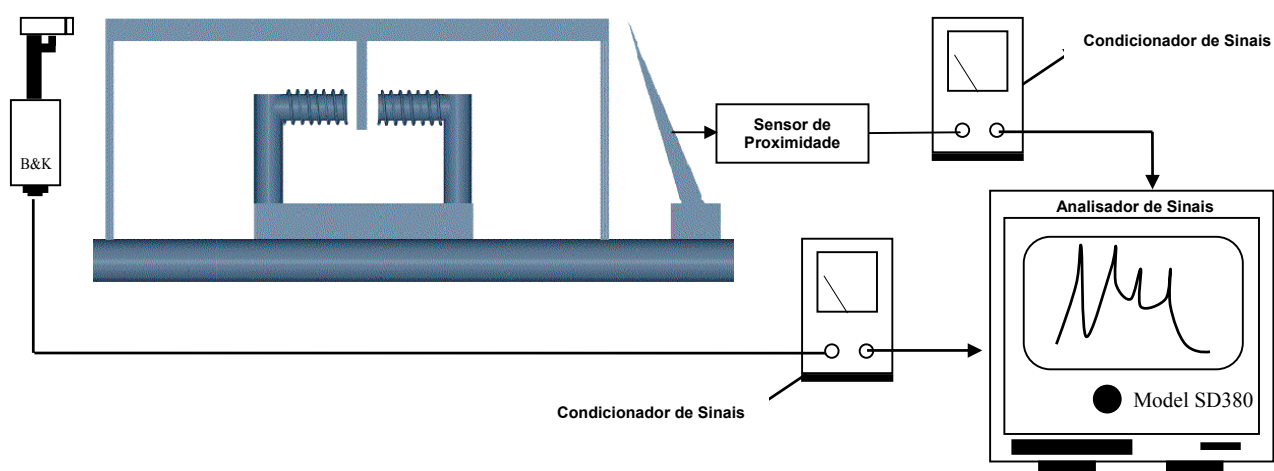


Figura 6.18 Esquema da montagem da instrumentação da bancada da mesa vibratória para caracterização dos controladores no domínio da frequência.

Foram feitas avaliações de desempenho de todos os controladores. Porém, serão mostrados apenas os resultados referentes ao controlador **M4**. A mesa, nos testes, foi sempre excitada com uma força impulsiva produzida por um martelo de impacto. A resposta em frequência para a mesa livre e sob controle é mostrada nas Figuras 6.19-21.

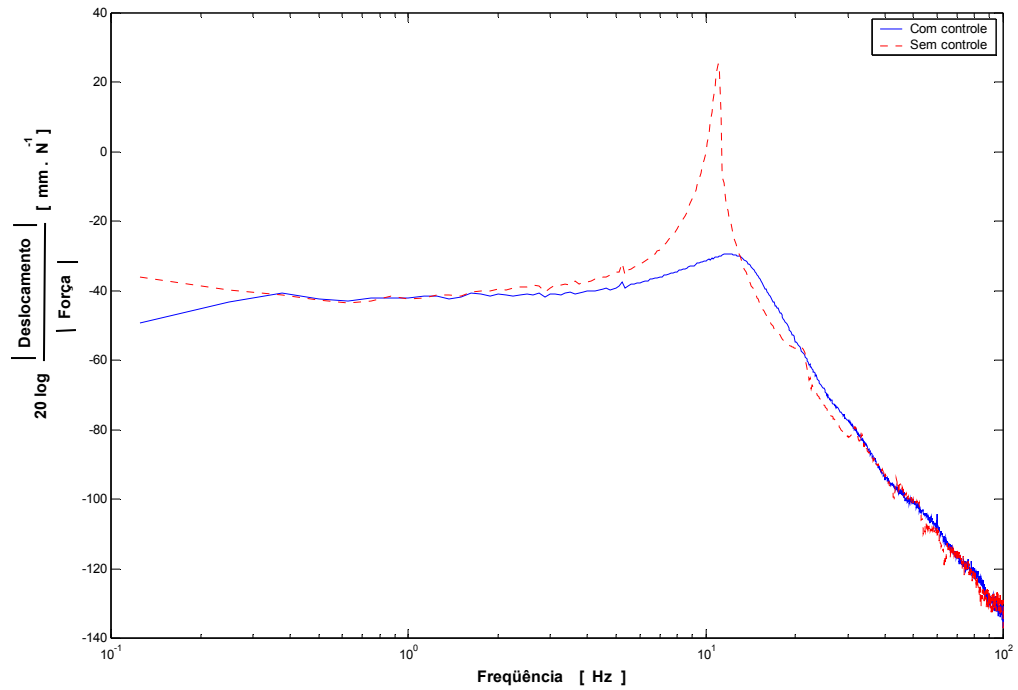


Figura 6.19 Comparação da resposta em frequência do sistema livre com o sistema com o controlador **M4**.

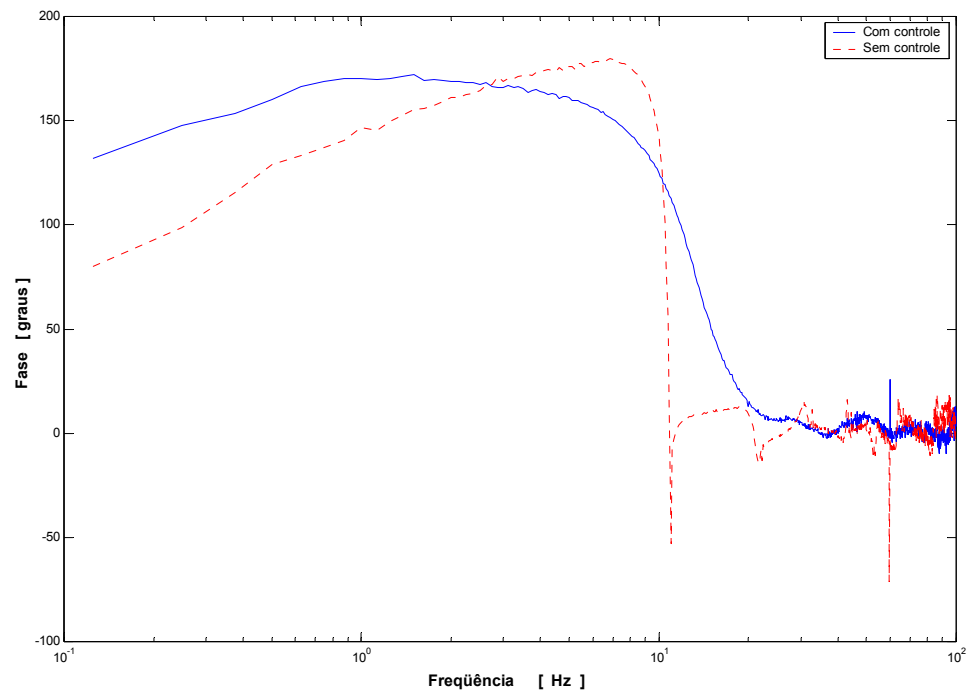


Figura 6.20 Comparação entre as fases do sistema livre com o sistema com o controlador **M4**.

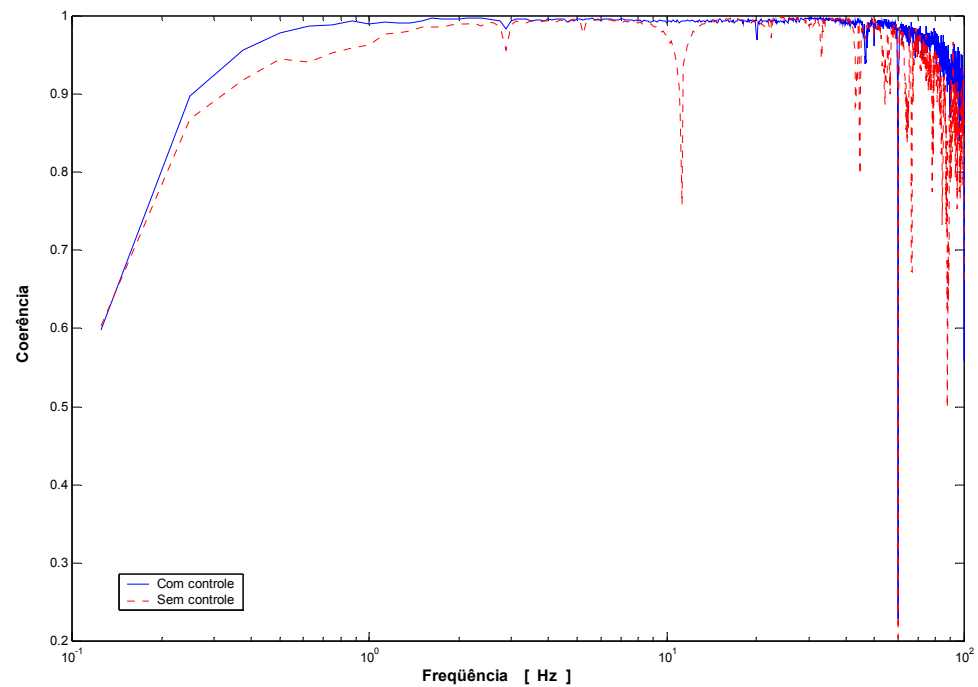


Figura 6.21 Coerência entre os sinais obtidos do sistema livre e com o sistema com o controlador **M4**.

Conforme pode ser observado nas Figuras 6.19-21, a característica fundamental do controlador é a introdução do amortecimento no sistema, tornando-o mais robusto às perturbações, especialmente àquelas perturbações com frequência próxima à de ressonância do sistema, no caso da mesa, 12 Hz. Observa-se ainda que a banda passante do sistema não se modifica, e o controlador introduz um pequeno atraso de fase em baixas frequências.

Capítulo VII

Controle e identificação de um sistema mecânico com infinitos graus de liberdade

A metodologia proposta é avaliada experimentalmente neste capítulo no controle ativo de vibrações de um sistema dinâmico que apresenta vários graus de liberdade.

O sistema é composto por uma viga flexível de aço inoxidável rigidamente fixada em uma das suas extremidades e livre na outra, conforme mostrado na Figura 7.1.

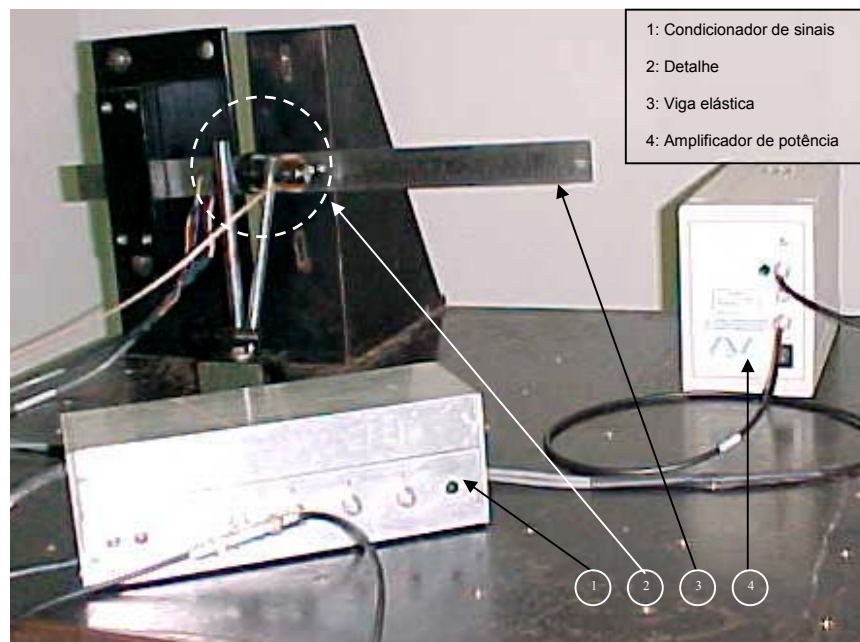


Figura 7.1 Bancada experimental do sistema mecânico vibratório.

A viga flexível é instrumentada com um sensor de proximidade que gera as informações dos deslocamentos da laterais da viga. Estas informações são utilizadas por um controlador *fuzzy*, implementado num PC. Os sinais de comando, calculados no PC, são enviados para um amplificador de fonte inversora que alimenta dois transdutores piezelétricos, que transformam, por sua vez, o sinal de entrada em momentos de controle. Os componentes¹ do ensaio são mostrados na Figura 7.3 .

¹ No anexo II é apresentada a descrição dos equipamentos desta bancada experimental

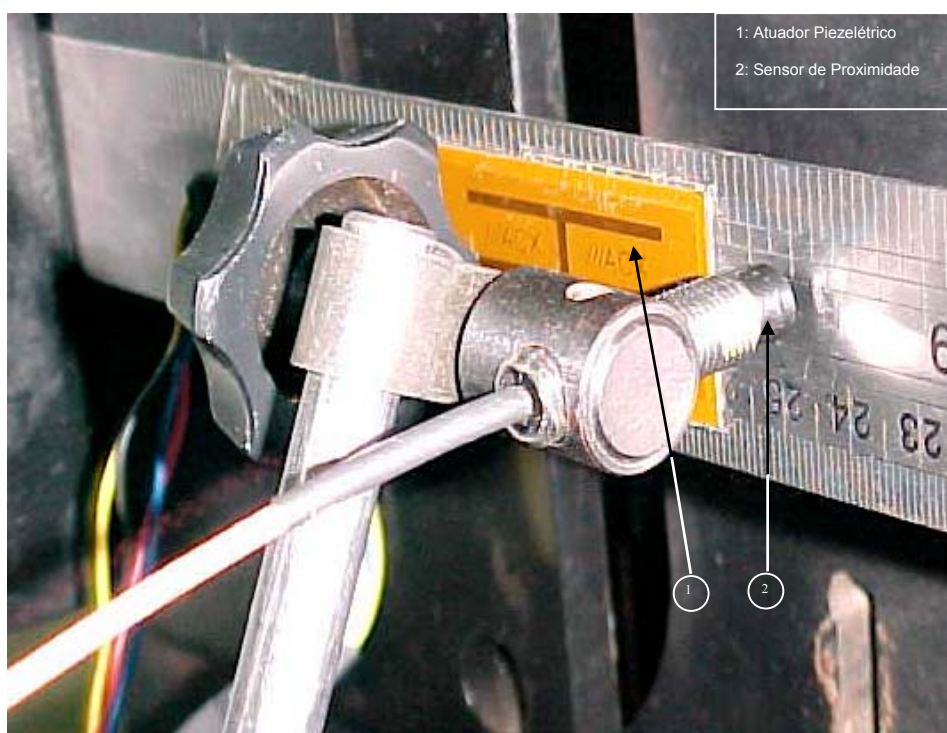


Figura 7.2 Detalhe da instrumentação da bancada experimental

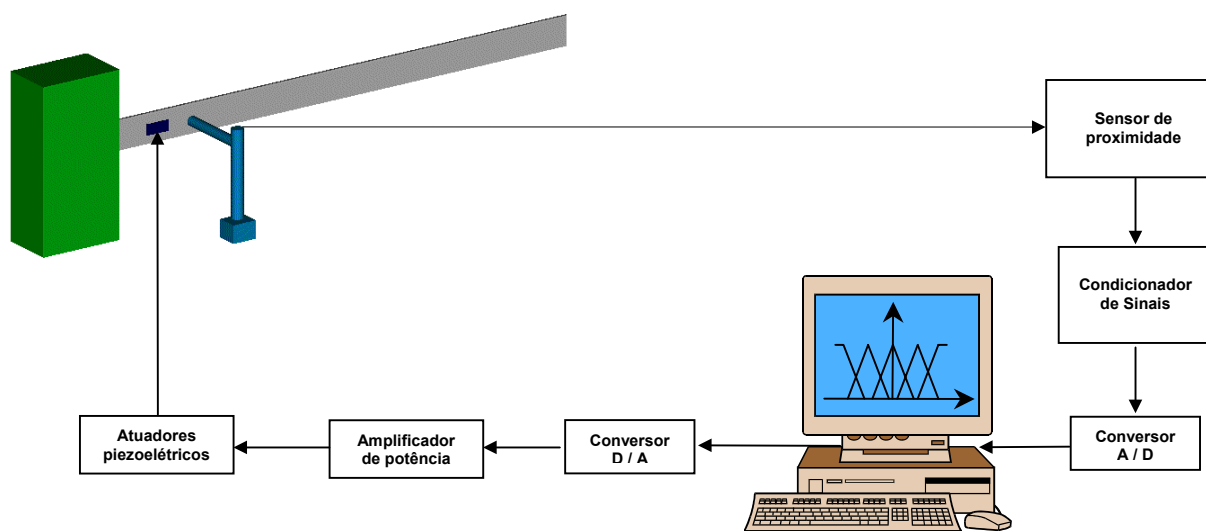


Figura 7.3 Esquema da montagem da instrumentação da bancada da viga vibratória.

O posicionamento do atuador ao longo da viga, não foi objeto de análise neste trabalho, pois aproveitou-se uma estrutura experimental já disponível no laboratório. A posição do sensor, por sua vez, foi definida considerando-se a excursão do deslocamento lateral da viga e a sensibilidade do sensor.

A Figura 7.4a mostra a resposta no tempo do comportamento da viga, observado pelo sensor, quando excitada por uma força de impacto. Destaca-se o comportamento muito pouco amortecido do sistema e, por consequência, o elevado tempo de acomodação, maior que 80 segundos.

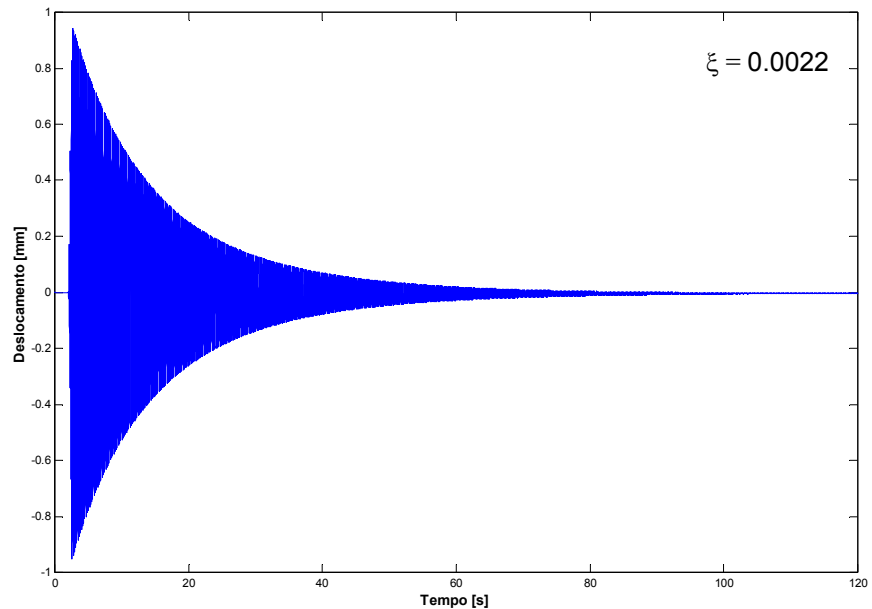


Figura 7.4a Resposta experimental ao impulso do sistema sem controle.

A Figura 7.4b mostra a resposta em frequência da viga obtida experimentalmente.

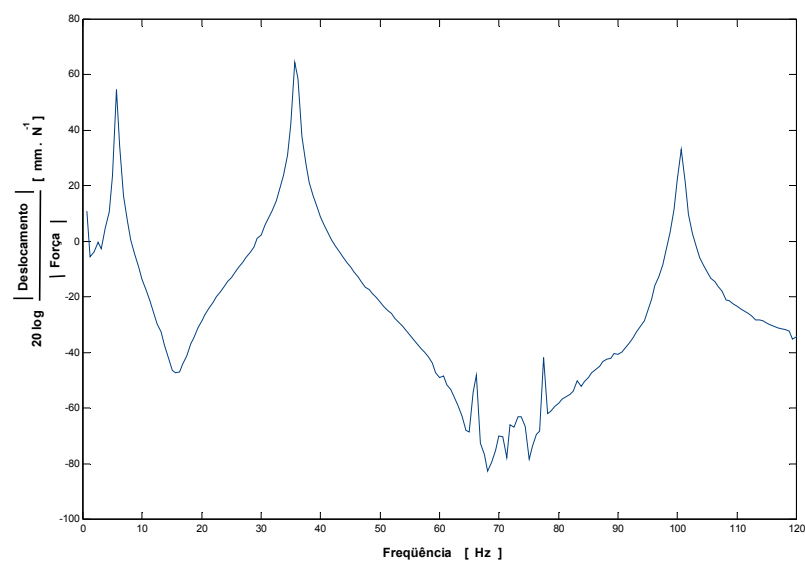


Figura 7.4b Resposta em frequência da viga

Como pode ser observado na Figura 7.4b, a viga apresenta três frequências naturais abaixo de 120 Hz. Pretende-se, neste experimento, construir um controlador fuzzy que atenua as vibrações em baixa frequência e controle o primeiro modo de vibrar da viga. Para atender esta condição de projeto o sinal do sensor é filtrado por um filtro digital passa-baixa com banda passante de 10 Hz.

7.1 Identificação experimental do sistema de infinitos graus de liberdade

Para identificação da viga adotou-se a mesma metodologia usada no experimento da mesa. Gerou-se sinais de comandos para o sistema e capturou-se, via o sensor, o deslocamento da viga. Os sinais, adquiridos e comandados foram armazenados em arquivo, a uma taxa de 300 Hz, e utilizados no treinamento e validação do modelo neural do sistema.

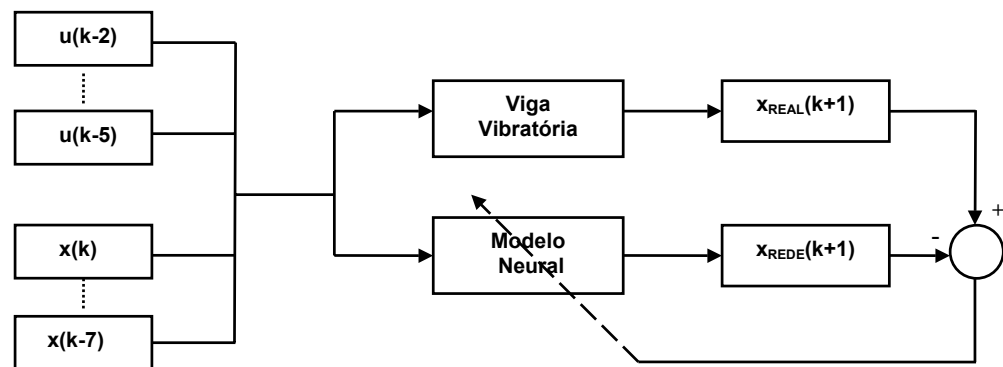


Figura 7.5 Identificação do modelo direto da planta

O vetor de entrada para o treinamento da rede foi:

$$X = [x(k) \quad x(k-1) \quad x(k-2) \dots x(k-8) \quad u(k-2) \quad u(k-3) \dots u(k-5)] \quad (7.1)$$

enquanto que a saída é o vetor $x(k+1)$. Este arranjo dos sinais de entrada foi obtido após alguns ensaios numéricos que avaliaram diferentes alternativas comparando a capacidade de propagação da rede em malha fechada. No caso deste experimento, diferentemente do caso anterior, a definição por esta arquitetura foi mais trabalhosa, fato esperado dada a maior complexidade do sistema a ser mapeado.

7.1.1 Treinamento

A arquitetura da rede neural adotada teve 12 neurônios na camada de entrada (4 sinais de comando, 8 sinais de saída), 10 neurônios na camada oculta e um na camada de saída. As funções de ativação adotadas foram: tangente hiperbólica na camada oculta e linear na saída. O algoritmo utilizado no treinamento foi o *backpropagation*.

Como já mencionado, a arquitetura de rede foi definida a partir de testes computacionais que avaliaram a capacidade de mapeamento da rede e o comportamento desta em malha fechada.

A Figura 7.6 mostra a resposta da viga a um sinal de excitação com frequência e amplitude variantes no tempo. A frequência variou, por três vezes, de 0 a 30 Hz nos primeiros 120 segundos de teste. No tempo restante, como mostram as figuras, o sinal de controle foi do tipo degrau.

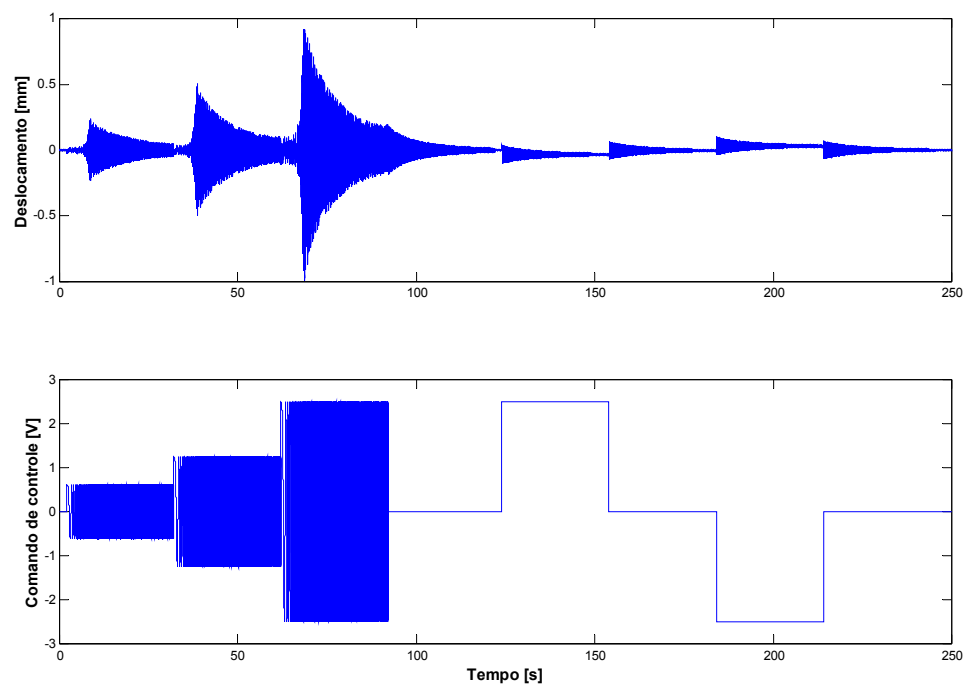


Figura 7.6 Sinais de leitura do sensor de proximidade e de comando de controle utilizados no treinamento

A Figura 7.7 mostra a evolução do erro RMS em função das iterações do algoritmo *backpropagation*.

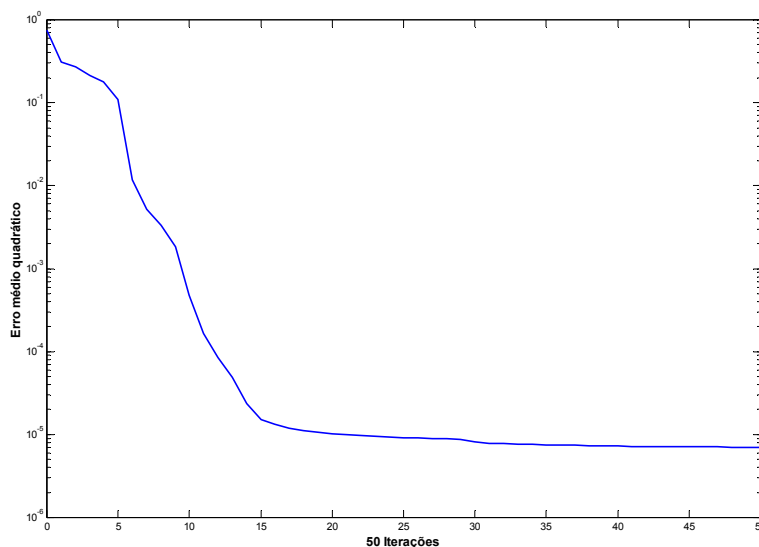


Figura 7.7 Evolução do erro RMS durante o treinamento realizado.

7.1.2 Validação com o sistema em malha aberta

Para validação, a rede foi excitada com um sinal senoidal de 3,0 Hz e amplitude de um volt. Os sinais de excitação e controle foram armazenados e submetidos à rede. A Figura 7.8 mostrar os resultados obtidos.

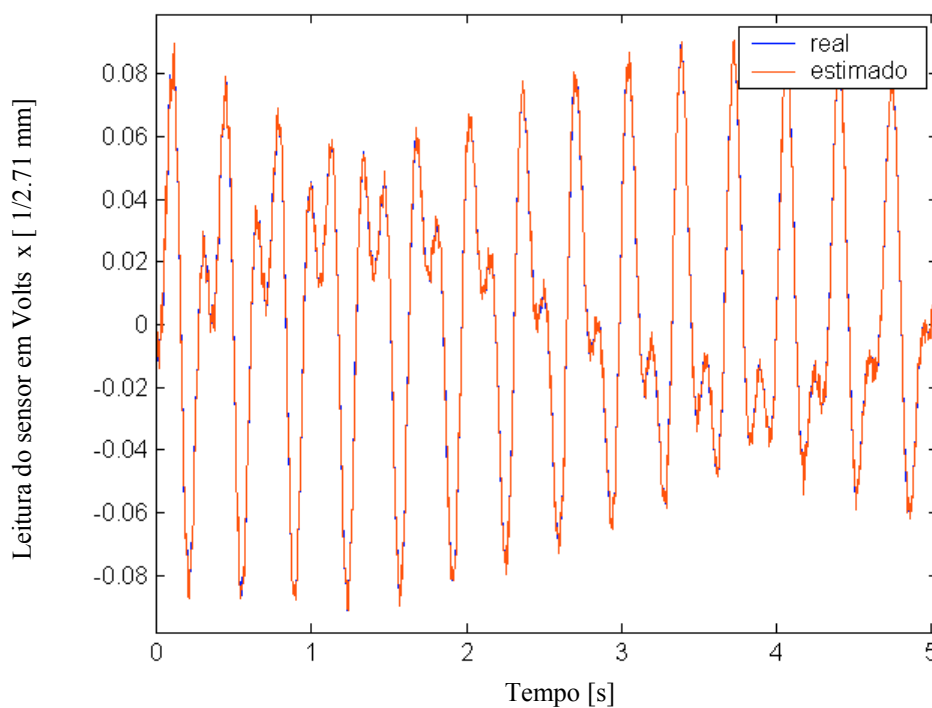


Figura 7.8 Validação com um sinal de controle senoidal.

A Figura 7.8 mostra a excelente capacidade de mapeamento da rede, que foi treinada com um conjunto de dados bem diferentes dos de validação.

7.1.3 Validação com o sistema em malha fechada

A validação em malha fechada procura avaliar a capacidade da rede em propagar o estado futuro a partir de estados anteriores calculados pela própria rede. Esta funciona como um "integrador" da dinâmica do sistema. A validação começa apresentando à rede um determinado estado inicial (vetor X , dado pela equação 7.1) e uma ação de controle conhecida. A partir daí ela evolui no tempo, realimentando-se com seus próprios dados.

Na validação observou-se: o comportamento em regime permanente, as frequências presentes no transitório e o amortecimento do sistema. Quando estes parâmetros se aproximavam dos valores esperados então o modelo era aceito.

No caso da viga, usou-se um sinal degrau de 2,5 volts para validar o sistema. Os resultados obtidos são apresentados nas Figura 7.9a,b-c.

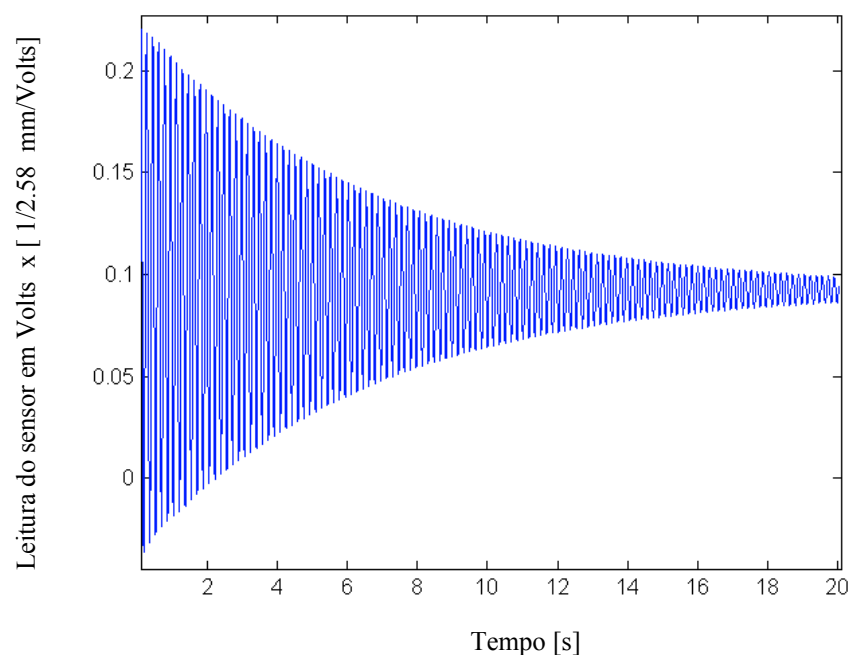


Figura 7.9a Resposta da rede em malha fechada com o controle constante de 2,5 volts.

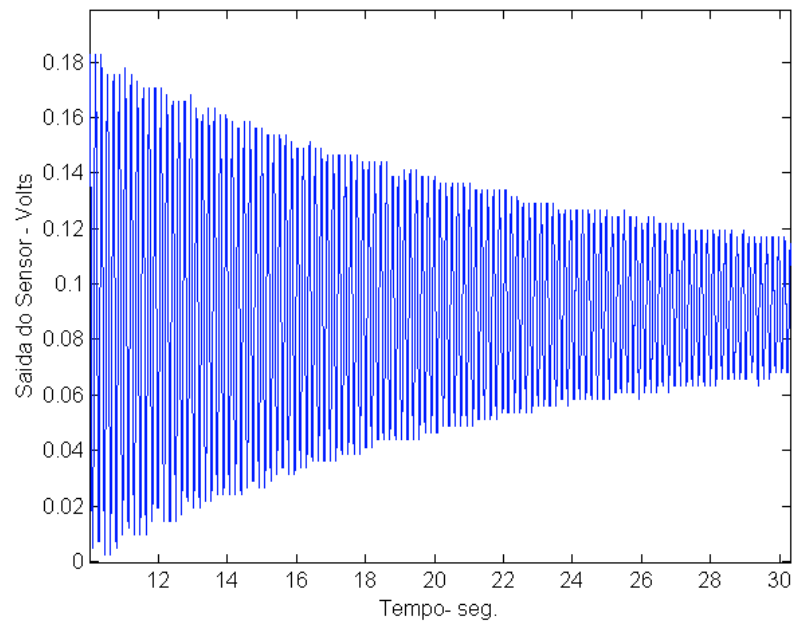


Figura 7.9b Resposta da viga a uma excitação constante de 2,5 volts.

Observa-se nas Figuras 7.9a-b que a rede retém a característica pouco amortecida do sistema e sua saída converge para o valor real.

A Figura 7.9c mostra a densidade espectral de potência presente no sinal da viga e no sinal gerado pela rede.

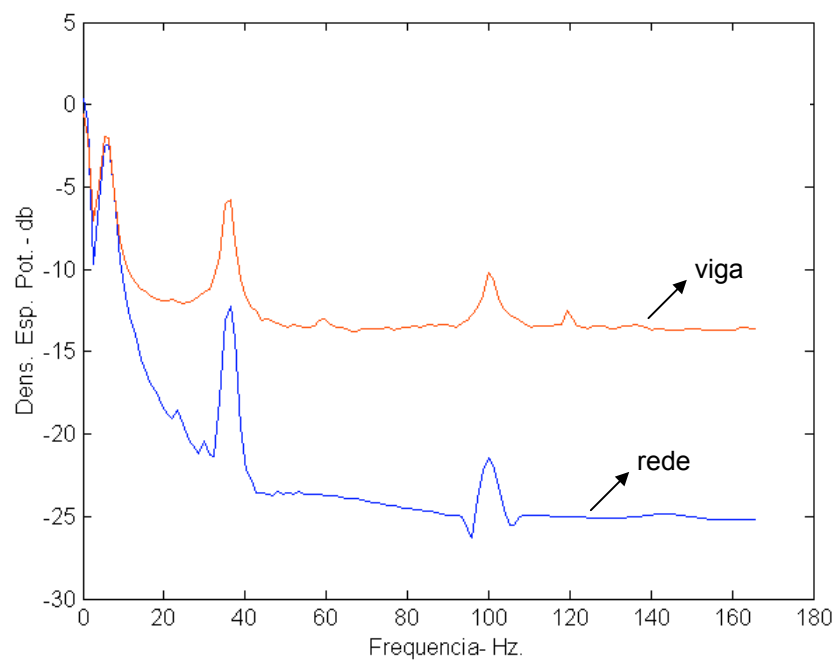


Figura 7.9c – Densidade espectral de potência do modelo neural e da viga real.

Como pode ser observado na Figura 7.9c, há uma boa concordância em baixa frequência (região de interesse do controlador) entre o modelo neural e a viga real.

Uma vez definido o modelo neural, o próximo passo é a otimização do controlador.

7.2 Otimização numérica dos controladores *fuzzy* utilizando algoritmos genéticos

Uma vez elaborado o modelo neural da viga, definiram-se as estratégias de otimização dos controladores *fuzzy* de forma semelhante ao efetuado com o experimento da mesa. Neste caso, o sinal de erro, $e(k)$, foi definido como o sinal de deslocamento da viga $y(k)$, obtido pelo sensor de proximidade. O sinal de erro e a sua variação no tempo (derivada) foram as entradas do controlador *fuzzy* e saída o sinal de controle.

7.2.1 Definição da função objetivo

A estratégia de otimização foi a mesma utilizada para o ensaio anterior com o sistema de um grau de liberdade, tendo como função objetivo:

$$IP_{VIGA} = -1/2 \int_0^t \sqrt{\alpha [x(t).F(t)]^2 + \beta [\dot{x}(t).H(t)]^2 + \gamma [u(t).G(t)]^2} dt \quad (7.2)$$

onde t é o tempo de simulação adotado em 3 segundos, $x(t)$ é a resposta do modelo neural, $u(t)$ é o valor de comando de controle obtido para um determinado controlador ou indivíduo do algoritmo. $F(t)$, $H(t)$ e $G(t)$ são funções que ponderam no tempo os sinais de deslocamento, de velocidade e de controle enquanto os parâmetros α , β e γ são pesos constantes.

No caso estudado estas funções $F(t)$, $G(t)$ e $H(t)$ foram iguais e são mostradas na Figura 7.10. Com estas funções procurou-se valorizar o comportamento em regime do sistema controlado e garantir a independência da otimização em relação às condições iniciais de simulação. Os pesos constantes foram escolhidos de forma que equilibrassem a contribuição das três parcelas na função objetivo. Os valores destas variáveis foram: $\alpha=100$, $\beta=1$ e $\gamma=100$.

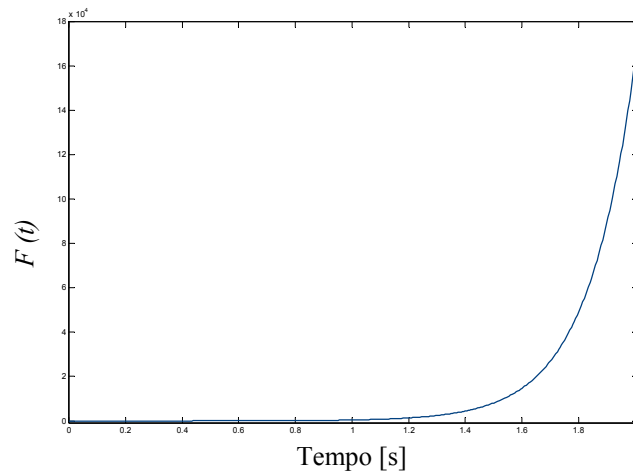


Figura 7.10 Função auxiliar $F(t)$ para cálculo da função objetivo.

Semelhantemente ao caso da mesa, o algoritmo genético usado neste experimento foi construído a partir da codificação dos indivíduos em ponto flutuante. Utilizou-se o torneio como mecanismo de seleção natural, promoveu-se a mutação e não foram realizados cruzamentos entre os indivíduos.

7.2.2 Controlador Inicial e as estratégias de otimização

O controlador *fuzzy* inicial adotado possui uma base regras completa, onde todas as regras ativam a função de pertinência nula na saída, o que significa dizer, que para quaisquer entradas, o controlador inicial gera uma saída zero. Os pesos adotados para as regras foram todos unitários. Escolheu-se tanto para as variáveis linguísticas da entrada $e(k)$ e $\Delta e(k)$ cinco curvas de pertinência tipo gaussianas: negativo grande (NG), negativo pequeno (NP), zero (Z), positivo pequeno (PP) e positivo grande (PG), totalizando 25 regras. Para as saídas adotou-se cinco valores *crisp*, configurando o controlador *fuzzy* como do tipo Takagi-Sugeno.

O procedimento percorreu o seguinte o caminho:

- a) otimizou-se primeiramente o conjunto de regras, tendo como variável dependente a função de pertinência da saída;
- b) a partir do controlador obtido no passo anterior, otimizaram-se os pesos associados a cada uma das regras;
- c) em seguida, reduziu-se o conjunto de regras, eliminando aquelas com peso inferior a 0,5 e fez-se novamente a otimização dos pesos, agora numa base de 13 regras (para este experimento);

- d) o controlador anterior foi mais uma vez otimizado, observando agora como variáveis de otimização, os centros das funções de pertinência das duas entradas.

A Tabela 7.1 , mostra os testes efetuados e os respectivos valores de amortecimento experimentais obtidos.

Tabela 7.1 Controladores *fuzzy* otimizados

Controlador	Estratégia de otimização	Amortecimento (ξ)
V-livre	-	0,0022
V1	Conseqüente da regra	0,0667
V2	Pesos (base de regras completa)	0,0919
V3	Funções de pertinência das entradas (centro)	0,0967
V4	Pesos (base de regras reduzida a 13)	0,0927
V5	Funções de pertinência das entradas (centro)	0,0997

A Figura 7.11 mostra o comportamento numérico do controlador V1, obtido com a otimização da base de regras, tendo como variáveis o conseqüente de cada regra.

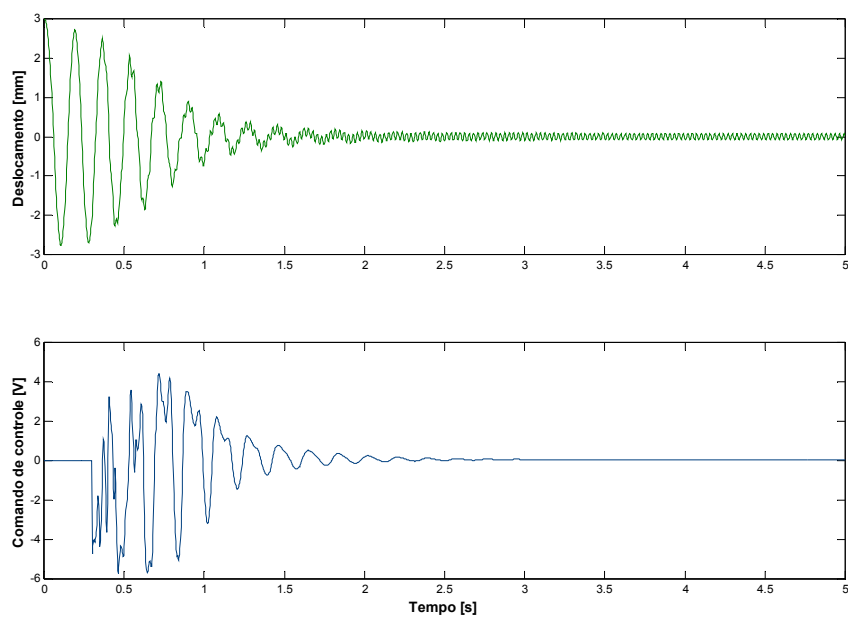


Figura 7.11 Sinais do deslocamento e do controle para o sistema simulado com a rede neural treinada e controlado pelo controlador **V1**.

7.3 Resultados dos testes experimentais no domínio do tempo

Todos os controladores obtidos no item anterior foram avaliados experimentalmente. Para isso o sistema foi excitado durante 5 segundos por um comando senoidal. A frequência da excitação (5,6 Hz) foi escolhida de forma que provocasse uma perturbação considerável na viga e a amplitude do sinal foi de 4,5 volts.

Dada a semelhança do comportamento dos diferentes controladores, apresentaremos apenas alguns deles.

Controlador V3

O controlador **V3**, como já mencionado, otimiza os centros das funções de pertinência da entrada. A sua base de regras, mostrada na Tabela 7.2, é a mesma do controlador **V2**.

Tabela 7.2 Conjunto de regras do controlador **V3**.

Erro $\Delta \text{ erro}$	NG	NP	Z	PP	PG
NG	PP(0,01)	NP(0,49)	NP(0,90)	NG(0,30)	PP(0,73)
NP	NG(0,99)	NG(0,95)	NG(0,63)	NG(0,45)	PP(0,84)
Z	NG(0,37)	PP(0,19)	Z(0,77)	PG(1,00)	NG(0,81)
PP	NG(0,37)	PG(0,49)	PP(0,70)	Z(0,02)	PP(0,03)
PG	NP(1,00)	Z(0,48)	NG(0,01)	PG(0,68)	PG(0,53)

O controlador **V3** teve alguns centros das funções de pertinência otimizados, conforme mostra a Figura 7.12.

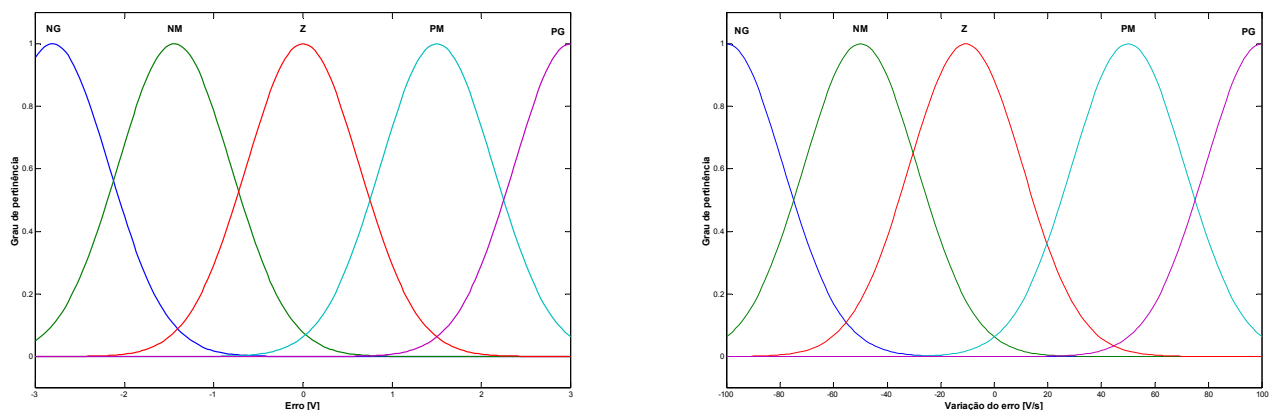


Figura 7.12 Funções de pertinência no universo do discurso do erro e da variação do erro, variáveis de entrada do controlador **V3**.

A Figura 7.13 apresenta o deslocamento da viga durante 3 segundos e o esforço de controle aplicado nos atuadores piezelétricos com o controlador **V3**.

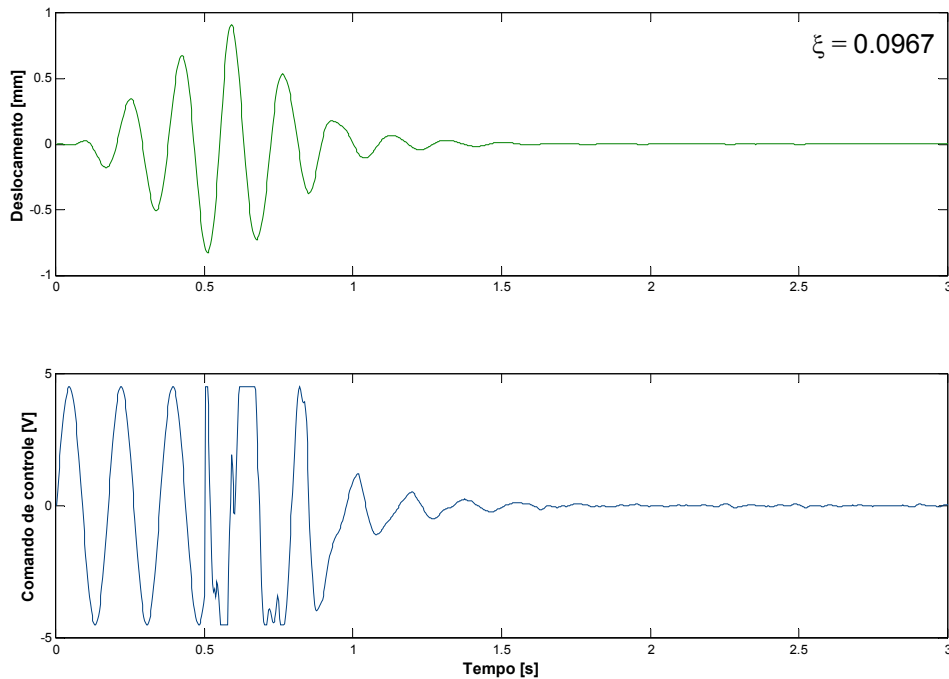


Figura 7.13 Deslocamento lateral e comando de controle para o sistema a malha fechada, com o controlador **V3** ativado 0,5 segundos após o início das excitações.

Este ensaio revela o bom comportamento do controlador, que atenua de forma significativa as vibrações num tempo bastante pequeno. Observa-se ainda que os deslocamentos nos centros das funções de pertinência das entradas não foram grandes, relativamente às condições iniciais, onde as funções de pertinência eram igualmente espaçadas. Por esta razão, o controlador obtido não melhorou de forma significativa o controlador fuzzy derivado no passo anterior, como revela o amortecimento mostrado na Tabela 7.1.

Controlador V5

O controlador **V5**, que possui os pesos modificados pelo controlador **V4**, tem a base de regras conforme mostra a Tabela 7.3.

Tabela 7.3 Conjunto de regras do controlador **V5**.

Erro Δ erro	NG	NP	Z	PP	PG
NG			NP(0,02)		PP(0,63)
NP	NG(1,00)	NG(0,77)	NG(0,31)		PP(0,52)
Z			Z(0,38)	PG(0,93)	NG(0,01)
PP			PP(0,24)		
PG	PG(0,49)			PG(0,63)	PG(0,45)

O controlador **V5** teve alguns centros das funções de pertinência otimizados, conforme mostra a Figura 7.14.

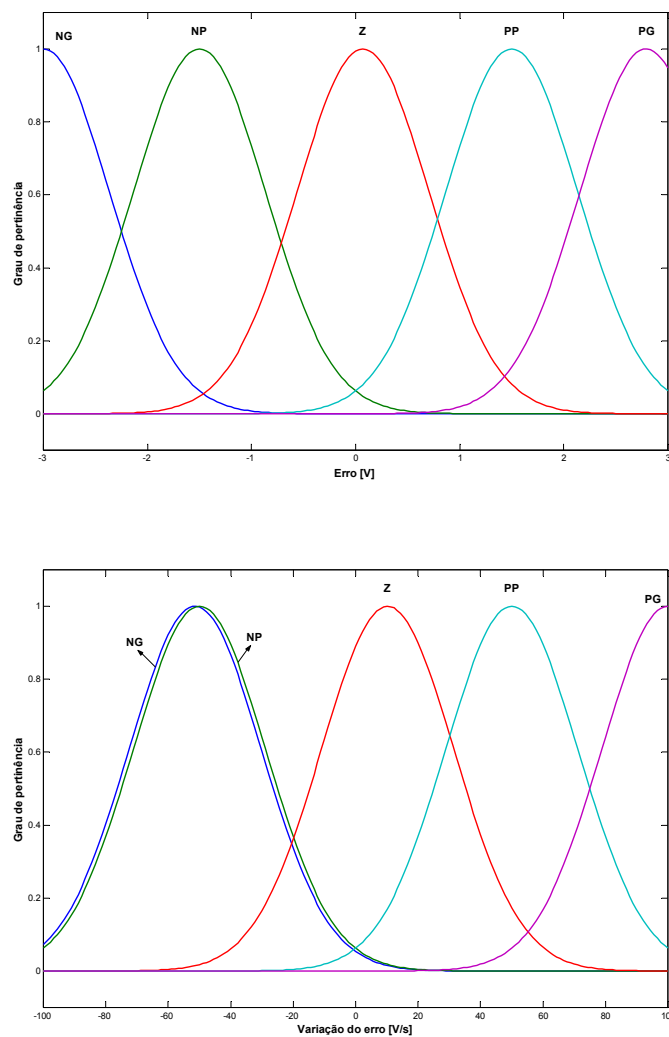


Figura 7.14 Funções de pertinência no universo do discurso do erro e da variação do erro, do controlador **V5**.

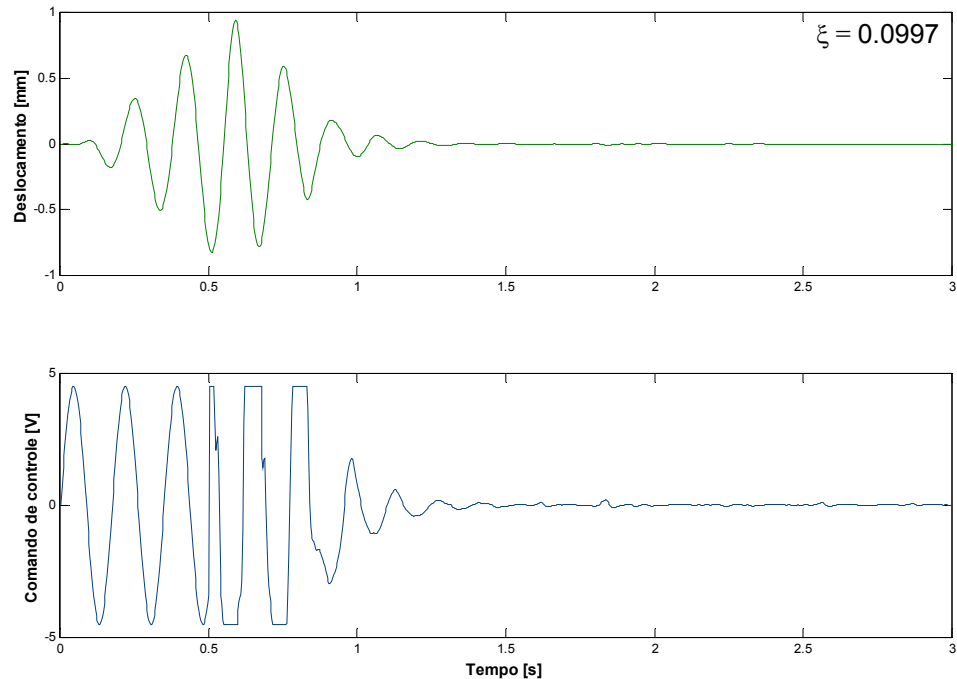


Figura 7.15 Deslocamento lateral e comando de controle para o sistema em malha fechada, com o controlador **V5** ativado em $t = 0,5$ segundo.

Observa-se, com este ensaio, o bom comportamento do controlador, a despeito da base de regras ser incompleta. Verifica-se ainda neste caso uma alteração importante nos centros da funções de pertinência, especialmente da variação do erro. A alteração observada no centro da função de saída NP sugere uma baixa sensibilidade do sistema controlador-viga à variação do centro desta função de pertinência. A Tabela 7.3 mostra que apenas uma das treze regras envolvem esta função de pertinência.

7.4 Caracterização do controle no domínio da frequência

Uma série de testes experimentais foram realizados com o propósito de avaliar o comportamento do controle no domínio da frequência.

As Figuras 7.16a-b mostra os equipamentos utilizados nos testes.

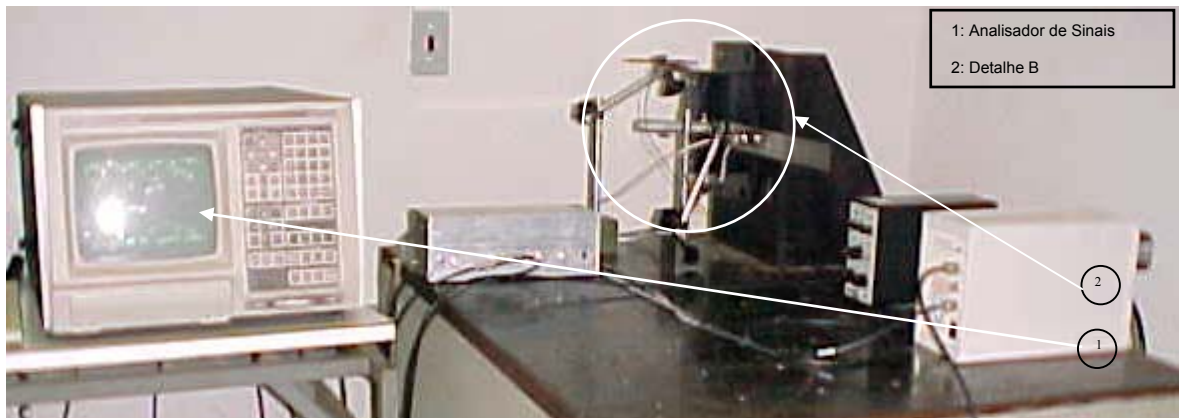


Figura 7.16a Esquema da montagem da instrumentação da bancada da viga vibratória para caracterização dos controladores no domínio da frequência.

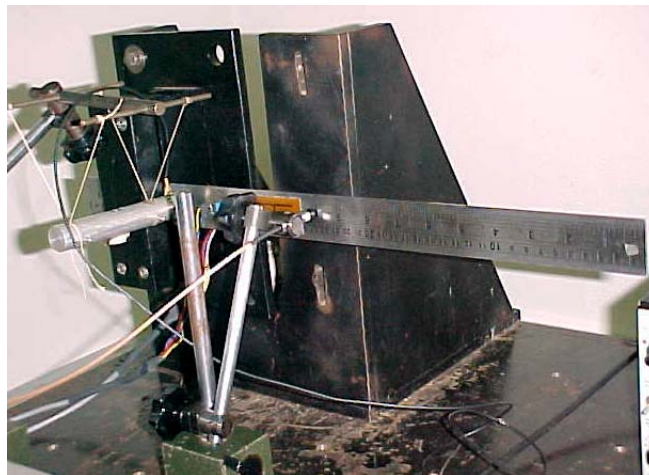


Figura 7.16b Detalhe B do esquema da montagem da instrumentação da bancada da viga vibratória para caracterização dos controladores no domínio da frequência, onde aparecem o sensor, o atuador e o pêndulo de impacto.

O sistema foi amostrado com um passo de 0,625 Hz, em uma banda de análise de 0- 500 Hz e utilizando janela retangular. Demais configurações do analisador, ver no Apêndice II.

Foram feitas avaliações da performance de todos os controladores no domínio da frequência, porém só serão mostrados os resultados obtidos com o controlador **V2**. No ensaio, a viga foi excitada com uma força impulsiva produzida pelo pêndulo de impacto. Esta mesma força é aplicada ao sistema livre e ao sistema controlado.

Na Figura 7.17 a resposta em frequência do controlador **V1** é comparado com a resposta do sistema livre.

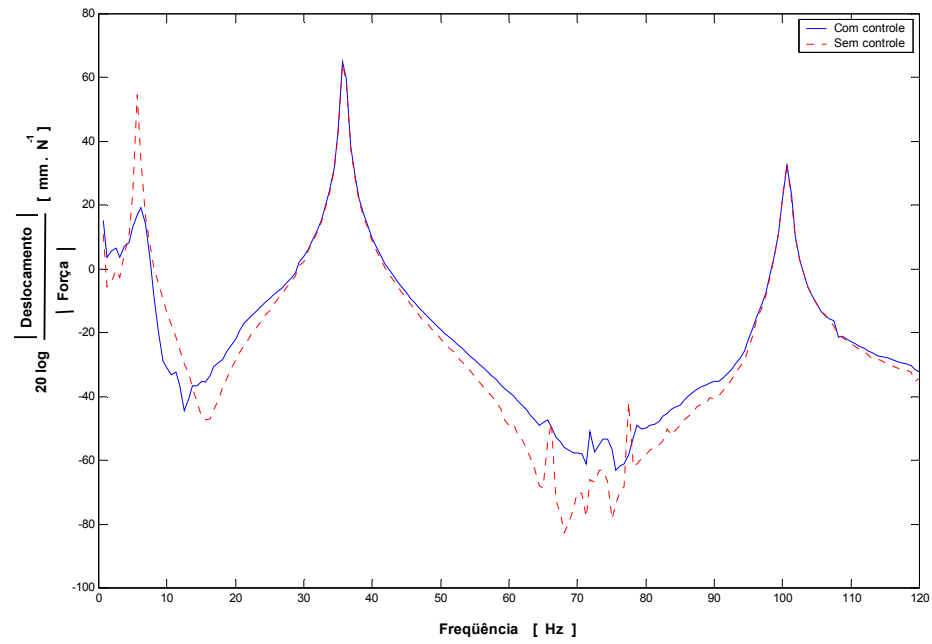


Figura 7.17 Resposta em frequência da viga livre e da viga controlada.

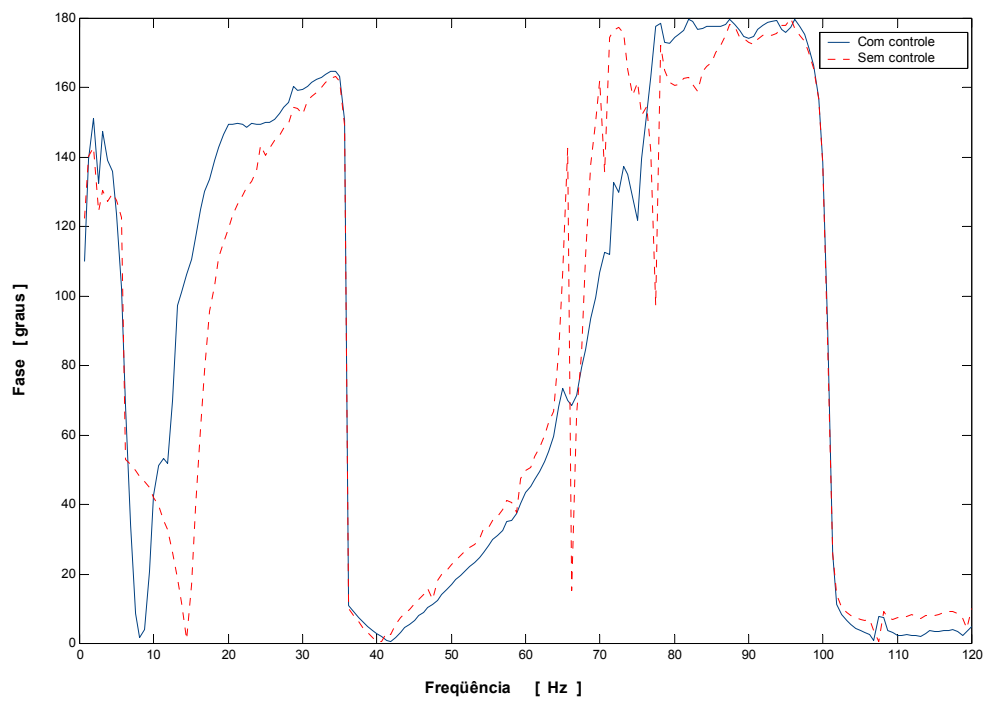


Figura 7.18 Variação da fase com a frequência da viga livre e da viga controlada.

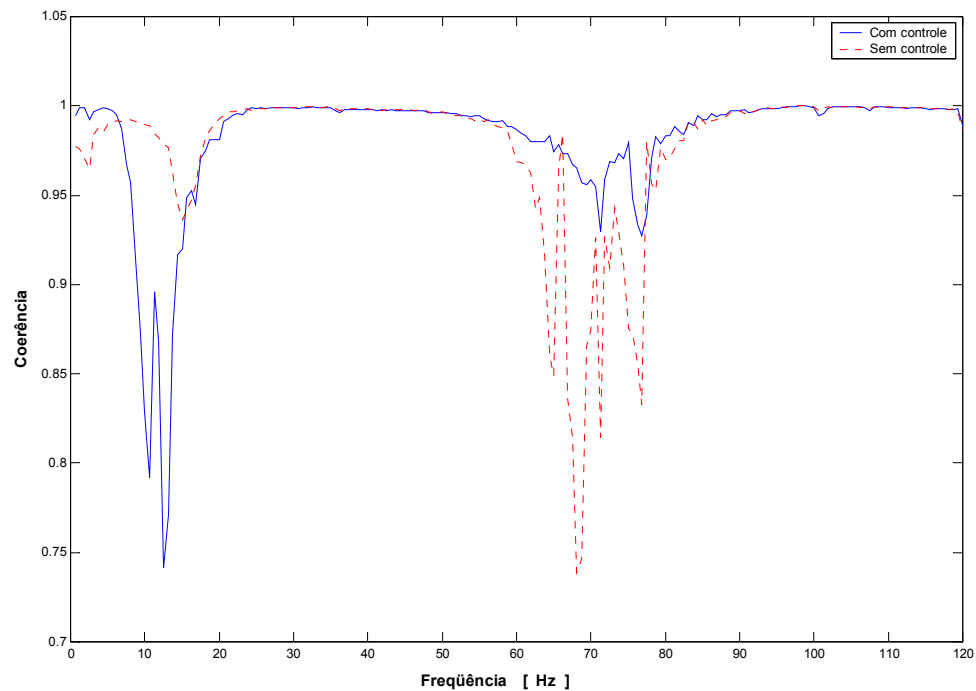


Figura 7.19 Coerência entre os sinais obtidos do sistema livre e do sistema controlado.

Os gráficos revelam o forte amortecimento introduzido pelo controlador na região de baixa frequência e a uma influência muito pequena em alta, fato que está em concordância com os resultados obtidos no domínio do tempo e mostrados nos testes anteriores. A pouca influência do controlador em altas frequências se deve ao fato do controlador simplesmente não "enxergar" estas frequências, pois o sinal do sensor é filtrado em 10 Hz.

Observa-se ainda que o controlador introduziu um certo atraso de fase no sistema, deve ter contribuído para isso o filtro passa baixas utilizado que, por natureza, provoca um atraso de fase.

O diagrama da coerência mostrado revela, por sua vez, uma boa confiança no diagrama de Bode em altas frequências e uma certa imprecisão em baixa frequência. Esta imprecisão não prejudica no entanto a avaliação sobre o comportamento do controlador feita anteriormente.

Capítulo VIII

Conclusão

Este trabalho apresentou uma metodologia de projeto de controladores híbridos inteligentes que envolveu, na sua formulação, conceitos de redes neurais artificiais, algoritmos genéticos(AG) e controladores fuzzy.

A metodologia proposta admite a disponibilidade de um conjunto de dados de entrada e saída do sistema que se quer controlar. Esses dados servem para projetar e validar uma rede neural artificial, que modela o comportamento do sistema dinâmico. O modelo neural da planta, uma vez definido, é utilizado no processo de otimização de controladores fuzzy. Este processo de otimização fundamenta-se nos resultados da teoria dos algoritmos genéticos.

A metodologia proposta foi avaliada numérica e experimentalmente no controle de vibrações de uma viga flexível engastada-livre e de um sistema massa-mola com um grau de liberdade. A metodologia não requer o modelo matemático da planta e pode ser aplicada em situações onde o conhecimento intuitivo do processo é muito vago.

Nos casos estudados utilizou-se o algoritmo *backpropagation* na construção dos modelos neurais. A otimização dos controladores fuzzy envolveu otimização genética, onde as variáveis dependentes foram modeladas por ponto flutuante, e os indivíduos experimentaram ao longo das gerações torneios seletivos e mutações.

Um aspecto importante da metodologia proposta refere-se à obtenção e validação do modelo neural em malha fechada do sistema. O modelo neural, obtido a partir de dados de entrada e saída da planta, deve apresentar - admitindo uma certa imprecisão - algumas características do sistema real, como por exemplo o ganho em regime permanente, o amortecimento e a frequência de oscilação.

O algoritmo genético revelou-se robusto, eficiente e, nos casos estudados, apresentou uma boa velocidade de convergência. A velocidade do AG foi definida, preponderantemente, pelo tempo envolvido com o cálculo da função objetivo. Verificou-se a sensibilidade do algoritmo em relação ao tamanho da população e ao número de gerações consideradas. Os valores adotados neste estudo mostraram-se adequados e não se fez nenhuma análise, em maior profundidade, sobre isso no trabalho.

Quanto ao controlador fuzzy, as informações demandadas pela metodologia proposta ao projetista são mínimas. Na estratégia utilizada definiu-se apenas o número de funções de pertinência das entradas e da saída e o intervalo de variações desses sinais. Admitiu-se ainda que as funções de pertinência eram gaussianas na entrada e valores *crisp* na saída.

O algoritmo foi capaz de otimizar o conjunto de regras, os pesos de cada uma e ajustar os centros das funções de pertinência das entradas e os valores *crisp* da saída.

De uma maneira geral, os objetivos deste trabalho foram alcançados, uma vez que :

- *foi apresentada uma metodologia de projeto de controladores híbridos inteligentes, centrados na lógica fuzzy, nos algoritmos genéticos e em redes neurais artificiais, que dispensa o modelo matemático e o conhecimento intuitivo do comportamento dinâmico do sistema a ser controlado, e*
- *a metodologia proposta foi avaliada, numérica e experimentalmente, no controle ativo de vibrações de sistemas dinâmicos, e os resultados obtidos demonstraram a eficiência da estratégia apresentada.*

Considerando os resultados obtidos no presente trabalho, alguns estudos futuros devem ser desenvolvidos, no sentido de:

- a) testar a metodologia para sistemas mais complexos, com mais de uma entrada e uma saída, não lineares, etc.;
- b) aprofundar os estudos na área de modelagem de sistemas dinâmicos por meio de redes neurais, observando especialmente a capacidade de operação das mesmas em malha fechada;
- c) investigar outras alternativas para modelar o sistema a ser controlado, como por exemplo: modelos ARMA, ARX, modelos neuro-fuzzy etc;
- d) avaliar outras estruturas para os algoritmos genéticos na otimização;
- e) analisar a sensibilidade do procedimento proposto em relação aos parâmetros do controlador fuzzy (pesos, regras, funções de pertinência etc.) e, desse modo, determinar quais são os parâmetros que mais contribuem para a otimização da função objetivo;
- f) Investigar alternativas de aplicação “*on line*” da metodologia proposta, onde a otimização utilize o sistema real.

Finalmente, é importante destacar que, apesar da metodologia proposta dispensar o modelo e até o conhecimento intuitivo do sistema dinâmico, quando se tem alguma informação sobre o comportamento desse sistema, tal informação pode e deve ser usada na obtenção do controlador.

Embora, nos casos apresentados, não se tenha usado formalmente o conhecimento intuitivo sobre os processos, este conhecimento esteve presente em todos os instantes da pesquisa, simplesmente por ser impossível ao pesquisador se desfazer do saber adquirido. De onde se conclui que ao se projetar os controladores deve-se usar todo o conhecimento que se tenha sobre o processo e, se esse conhecimento for insuficiente para se obter os resultados desejados, a metodologia proposta neste trabalho pode ser útil.

Referências Bibliográficas

Abreu, G. L. C M., Teixeira, R. L. and Ribeiro, J. F., 2000, "An neural network-based direct inverse control for active control of vibrations of mechanical systems, Anais do VI SBRN - Simpósio Brasileiro de Redes Neurais – CD Volume 1, Rio de Janeiro, RJ.

Assilian, S., 1974, "Artificial Intelligence Techniques in the Control of Real Dynamic Systems", Ph.D. thesis. Queen Mary College, University of London, London, UK.

Baitinger, U.G. , K. Kropp, 1993, "Optimization of fuzzy logic controller inference rules using a genetic algorithm, Proceedings of 1st. European Congress on Fuzzy and Intelligent Technologies, v.2, p. 1090-1096, 1993.

Bandler, W and Kohout, L.J. , 1980, "Fuzzy power sets and fuzzy implication operators", Fuzzy sets and systems 14, 155-174.

Banks, H.T., Smith, R.C., Wang, Y., 1996, "Smart Material Structures: Modeling, Estimation and Control", Wiley – Masson Research in Applied Mathematics Series.

Batur, C. and Kasparian, V. , 1991, "Predictive fuzzy expert controllers", Computers in Industrial Engineering 20(2).

Bailey, D. and Thompson, D. , 1990, "How to develop a Neural Network", AI Expert, Vol. 5, No. 6, pp. 38-47, 1990.

Braitenberg, V., 1990. "Reading the structure of brains" Network 1, 1-12.

Carrara, V. e Neto, A. R., 1999, "Redes Neurais para Controle de Atitude de Satélites", IV Simpósio Brasileiro de Automação Inteligente – SBAI, pp. 77-82.

Carse, B., Fogarty, T.C. and Munro, A.A., 1996, "Evolving fuzzy rule based controllers using genetic algorithms", Elsevier Science B.V., Fuzzy Sets and System 80, pp 273-293.

Cunha Jr, S.S., 1999, "Estudo Teórico e Numérico de Absorvedores Dinâmicos de Vibrações, Dissertação de Mestrado, Universidade Federal de Uberlândia, Uberlândia –MG.

Delben, A., Bretas, N. and Carvalho, A. , 1996, “ Restabelecimento ótimo de energia em sistemas de distribuição: Uma comparação entre o desempenho de busca fuzzy e algoritmos genéticos. Technical Report 48, Instituto de Ciências Matemáticas de São Carlos –SP.

Den Hartog, J.P.,1956, Mechanical Vibrations, Fourth Edition, McGraw-Hill Book Co.

Dombi, J. , 1982, “A general class of fuzzy operators, the De Morgan class of fuzzy operators and fuzziness induced by fuzzy operators”. Fuzzy Sets and Systems 8, 149-163.

Dubois e Prade , 1986, “New results about properties and semantics of fuzzy set-theoretic operators”. In P.P. Wang and S.K.Chang (Eds), Fuzzy Sets,pp. 59-75. New York,USA:Plenum Press.

Dubois, D. and H. Prade , 1988, “Possibility theory - an approach to computerized processing of uncertainty”. New York, USA:Plenum Press. ISBN/ISSN 0-306-42520-3.

Dubois, D. and Prade,H. , 1991, “Fuzzy sets in approximate reasoning, part 1: Inference with possibility distributions”. Fuzzy Sets and Systems 40,143-202.

Elman, J.L. , 1990, “Finding structure in time.” Cognitive Science 14,179-211.

Gaines, B.R. , 1976, “Foundations of fuzzy reasoning”, International Journal of Man-Machine Studies 8, 313-327.

Giles, R. , 1976, “Lukasiewicz logic and fuzzy set theory”, International Journal of Man-Machine Studies 8,313-327.

Goldenberg, D.E., 1989, Genetic Algorithms in search, optimization, and machine learning”, Addison-Wesley Publishing Company, Inc. USA.

Hadaegh, F.Y. , Chiang, R.Y. , Lin, C.F. e Politopoulos , 1994, “Structural Vibration Suppression using Fuzzy Logic Control, IEEE, 0-7803-1872-2/94, pp 953-958.

Hamacher, H. , 1978, “Über logische verknüpfungen unscharfer aussagen und deren zugehörige bewertungsfunktionen”, In G.J. Klir T, Trappl an L. Ricciardi (Eds.), Progress in Cybernetcs and System Research, Volume 3, Washington (DC), USA: Hemisphere.

Hanes, M.D., Ahalt, S.C., and Orin, D.E., 1994, "Intelligent Control of Object Acquisition for Power Grasp", IEEE International Symposium on Intelligent Control-Proceedings 1994, pp. 303-308.

Harris, C.J. and Moore, C.G. , 1989, "Intelligent identification and control for autonomous guided vehicles using adaptive fuzzy-based algorithms", Engineering Applications of Artificial Intelligence 2, 267-285.

Harp, S.; Samad, T. and Guha, A. , 1989, "Towards the Genetic Synthesis of Neural Networks", in Proceedings of Third International Conference on Genetic Algorithms, Morgan Kaufmann, San Mateo, CA, 1989.

Hebb, D. , 1949, "Organization of behavior", John Wiley, New York, NY, 1949.

Heider, H. and Drabe, T., 1997, " Fuzzy System Design with a Cascaded Genetic Algorithm", proceeding in IEEE, 0-7803-3949-5/97, pp 585-588.

Hellendoorn, J. , 1990, "Reasoning with Fuzzy Logic", Ph.D. thesis, Delft University of Technology, Delft, The Netherlands.

Hines, J.W. , 1997, "MATLAB Supplement to Fuzzy and Neural Approaches in Engineering", copyright by John Wiley & Sons, USA.

Holland, J.H. , 1975, "Adaptation in Natural and Artificial System", University of Michigan Press, Ann Arbor, MI, 1975.

Holland, J.H. , 1992, "Genetic Algorithms", Scientific America, Vol 267, pp. 66-72, 1992.

Homaifar, A, E. McCormick , 1995, "Simultaneous design of membership functions and rule sets for fuzzy controllers using genetic algorithms, IEEE Transactions on Fuzzy System, v.3, p. 129-139, 1995.

Hopfield, J.J., 1982, "Neural networks and physical systems with emergent collective computational abilities", Proceedings of the National Academy of Sciences , 2554-2558.

Houck, C.R. , Joines, J.A. and Kay, M.G. , 1995, A genetic algorithm for function optimization: A Matlab Implementation”, NCSU-IE: North Carolina State University- Industrial Engineering, Technical report 95–09.

Huang, S.J. and Lian, R.J., 1996, “ A combination of fuzzy logic and neural network algorithms for active vibration control”. Proc. Instn Mech Engrs –Part I – Journal of Intelligent Material Systems and Control, vol. 210, No. 3, pp. 153-329.

Hwang e Zein-Sabatto ,1997, “Fuzzy Controller Design Using Genetic Algorithms”, proceeding in IEEE, 0-7803-3844-8/97, pp 6-10.

Jager, R. , H.B. Verbruggen and Buijn, P.M. , 1991, “Adaptive fuzzy control”. In proceedings 1991 European simulation, Ghent, Belgium, pp 140-144, SCS.

Karr, C. , 1991, “Design of an adaptive fuzzy logic controller using a genetic algorithm”, In R. Belew and L. Booke, Eds., Proceeding on Fourth International Conference on Genetic Algorithms, pp. 450-457, Morgan Kaufmann, Los Altos, CA, 1991.

Kinzel, J. Klaawonn,F. and Kruse, R., 1994, “Modification of genetic algorithms for designing and optimising fuzzy controllers, in Proc. First IEEE Internationala Conference on Evolutionary computation, pp 28-33.

Kohonen, T., 1984, “ Self-organization and associative memory”. Springer-Verlag, Berlin.

Korenev, B.G. and Reznikov, L.M., 1993, “Dynamic Vibration Absorbers, Theory and Technical Applications”, John Wiley & Sons Ltd., England.

Lakoff, G. , 1973, “Hedges: a study in meaning criteria and the logic of fuzzy concepts”, Journal of Philosophical Logic 2, 458-508.

Larsen, P.M. , 1980, “Industrial applications of fuzzy logic control”, International Journal of Man-Machine Studies 12, (1),3-10.

Lee, C.C. , 1990a, “Fuzzy logic in control system: fuzzy logic controler – part i & ii”, IEEE Transactions on Systems, Man and Cybernetics, Vo. 20, No.02, pp. 404-418.

Lee, C.C. , 1990b, "Fuzzy logic in control system: fuzzy logic controller – part ii", IEEE Transactions on Systems, Man and Cybernetics, Vo. 20, No.02, pp. 419-435.

Lee, M. and Takagi, H. , 1993, "Integrating design stages of fuzzy systems using genetic algorithms", in Proceeding of Second IEEE International Conference on Fuzzy Systems, pp, 612-617, IEEE San Francisco, 1993.

Ling, P. and Xu, H. , 1996, " Fuzzy evaluation and control for the comfort sense of an elevator". In IEEE International conference on Industrial Technology, pp 607-611.

Linkens, D.A. and Abbod, M.F. , 1998, "Fuzzy logic monitoring and control of depth of anesthesia", Proceeding in IEE, Savoy Place, London Wc2ROBL,UK.

López, A.L., 1999, "Controle Nebuloso e Algoritmos Genéticos: Uma aplicação ao robô móvel Khepera, dissertação de mestrado, Universidade Federal de Santa Catarina, Florianópolis, SC.

Mamdani, E.H. , 1974, "Applications of fuzzy algorithm for simple dynamic plant", Proceedings IEE 121(12), 1585-1588.

Mamdani, E.H. and Assilian,S , 1975, "An experiment in linguistic synthesis with a fuzzy logic controller", International Journal of Man-Machine Studies 7,1-13

Mamdani, E.H. , 1977a, "Applications of fuzzy logic to approximate reasoning using linguistic synthesis", IEEE Transactions Computer, vol. C-26, no. 12, pp. 1182-1191.

Mamdani, E.H. , 1977b, "Applications of fuzzy set theory to control systems: a survey", In Fuzzy Automata and Decision Processes, M.M. Gupta, G.N. Sardis and B.R. Gaines. Eds. Amsterdam: North-Holland, pp. 77-88.

Marques, R.F.A., 2000, "Estudo Teórico e Numérico de Absorvedores Dinâmicos de Vibrações Ativos e Adaptativos", Dissertação de Mestrado, Universidade Federal de Uberlândia, Uberlândia-MG.

Matsuoka, H. , 1991, "A simple fuzzy simulation model for nuclear reactor system dynamics", Nuclear Technology 94,228-241.

McCulloch, W. and Pitts, W. , 1943, "A Logical Calculus of the Ideas Immanent in Nervous Activity", Bulletin of Mathematical Biophysics, 1943.

Mendel, J.M., 1995, "Fuzzy Logic Systems for Engineering: A Tutorial", Proceedings of the IEEE, vol. 83, No. 3, pp. 345-377.

Michalewicz, Z. , 1994, Genetic Algorithms + Data Structures = Evolution Programs, Second, Extended edition, Springer-Verlag. USA.

Miller, R., 1987, "Representation of brief temporal patterns, Hebbian synapses, and the left-hemisphere dominance for phoneme recognition", Psychobiology 15, pp 241-247.

Minsky, M.L. and Papert, S.A. , 1969, "Perceptrons", MIT Press, Cambridge, Ma., USA, 1969.

Moricic, B. and Nikolov, Z , 1990, "GENNET- System for Computer Aided Neural Network Design Using Genetic Algorithms", Proceedings of IJCNN-90, Washington, DC, January 1990.

Moudgal, V.G. , Passino, K.M. and Yurkovich, S. , 1994, " Expert Supervisory Control for a Two-Link Flexible Robot, IEEE, 1050-4729/94, pp3296-3301.

Olgac, N. and Holsek, M., 1997, "Active Vibration Absorption using the Delayed Resonator with Relative Position Measurement ", Journal of Dynamic System, Measures and Control, Vol. 119, pp 131-136.

Parker, D. , 1982, "Optimal Algorithms for adaptive networks: second order back propagation, second order direct propagation and second order hebbian learning, Proc. Of the IEEE International Conference on Neural Networks, Vol. II, San Diego, CA, pp 593-600.

Nashif, A.D. Jones, D.I.G., Henderson, J.P., 1985, Vibration Damping, John Wiley & Sons.

Park, S.H. , Kim, Y.H. , Choi, Y.K., Cho, J.C. and Jeon, H.T. , 1993, "Self-organization of Fuzzy Rule Base Using Genetic Algorithm", Proceeding in a Firth IFSA World Congress, pp 881-886.

Pétie, A.M. , Moreau, A. and Willaëys, D. , 1990, "Tool for approximate reasoning in expert systems", In Proceeding TIMS/ORSA Conference, Las Vegas, USA.

Pedrycz, W. , 1993, “ Fuzzy Control and Fuzzy Systems, second (extended) edition, John Wiley & Sons, New York.

Rade, D.A. and Steffen Jr.,V. , 1999, “Optimization of Dynamic Absorbers Over a Frequency Range”, proceeding of 17th International Modal Analysis Conference, Orlando, Florida, US, pp. 188-193.

Raza,H., Ioannou, P., and e Yossef, H.M., 1994, “Surface Failure Detection for an F/A-18 Aircraft Using Neural Networks and Fuzzy Logic”, in IEEE Interntional Conference on Neural Networks – Conference Proceedings, Vol. 5, IEEE, Piscataway, NJ, 94CH3429-8, pp.3363-3368.

Ribeiro, J. F., 1995, “Controle Robusto aplicado ao Problema de Vibrações em Rotores Flexíveis”, Tese de Doutorado, São José dos Campos.

Rogers, C. A., 1992, "Intelligent Material Systems – The Dawn of New Materials Age", Center for Intelligent Material Systems and Structures, Virginia Polytechnique Institute and State Univesity.

Roseblatt, F., 1958, “The Perceptron: a probabilistic model for information storage and organization in the human brain”, Psychological review, vol.65, 386-408.

Rumelhart, D.E. ; Hinton, G.E. and Williams R.J. , 1986, “Learning Representations by Back-propagating Errors”. Nature, 323:533-536, 1986.

Santos, M.B., 1999, "Uma contribuição à análise de sinais utilizando Wavelets", Dissertação de mestrado, Universidade Federal de Uberlândia- Minas Gerais, Brasil.

Schweizer, B. and Sklar, A. , 1961, “Associative functions and statistical triangle inequalities”. Publicationes Mathematicae Debrecen 8, 169-186.

Steffen Jr., V., Rade, D.A., 2000, Optimisation of Dynamic Vibration Absorbers over a Frequency band, Mechanical Systems and Signal Processing, Vol. 14, No. 5, pp 679-690.

Sugeno, M. , 1977, "Fuzzy measrues and fuzzy integrals: a survey". In M.M.Gupta, G.N. Sardis and B.R. Gaines (Eds, Fuzzy Automata and Decision Processes, pp. 89-102. Amsterdam, The Netherlands:North-Holland Publishers.

Sun, J. Q., Jolly, M.R. and Norris, M.A., 1995, " Passive, adaptive and active tuned vibration absorber – A survey", Transactions of the ASME, vol.117, pp. 234-242.

Takagi, T. and Sugeno,M. , 1983, "Derivation of fuzzy control rules from human operator's control actions", in proceeding of IFAC Symposium on Fuzzy Information, Knowledge, Representation and Decision Analysis, pages 55-60, Marseilles, France.

Teixeira, R.L. and Ribeiro, J.F., 2001, "Semi-active control of Vibrations of Mechanical using Fuzzy Logic, Genetic Algorithms and Neural Networks, Proceedings of the IX DINAME-International Symposiun on Dynamic Problems of Mechanics, Florianópolis – SC , pp 39-44.

Teixeira, R.L. e Ribeiro, J.F., 2000, "Controle de vibrações usando lógica fuzzy com regras otimizadas por algoritmos genéticos" CONEM 2000- Congresso Nacional de Engenharia Mecânica, Natal, RN –Brasil.

Terano,T. , Asai, K. and Sugeno,M., 1992, "Fuzzy Systems Theory and Its Applications, Academic Press, Boston.]

Thrift, P, 1991, "Fuzzy logic synthesis with genetic algoritms," in proc. IV International Conference on Genetic Algorithms, pp 450-457.

Tsang, E. C.C. e Yeung,D.S.,1999, "Optimizing Fuzzy Knowledge Base by Genetic Algorithms and Neural Networks, proceeding in IEEE, Vol. III, pp 367-371.

Tsoukalas, L.H., Ikonomopoulos, A., Uhrig, R.E., 1992, Virtual Measurements Using Neural Networks and Fuzzy Sets, in Proceedings of American Power Conference", Vol. 54-II , pp. 1437-1442 ,54th Annual Meeting, Chicago, April 13-15.

Túpac, Y. J., Pacheco, M. A., Velasco M. e Tanscheit, R., 1999, "Geração do conjunto de regras de interferência para um controlador nebuloso usando algoritmos genéticos, 4o SBAI – Simpósio de Automação Inteligente, SP.

Uhrig, R.E., Tsoukalas, L.H., Ikonomopoulos, A., 1994, " Application of Neural Networks and Fuzzy Systems to Power Plants ", in Proceedings of the IEEE International Conference on Neural Networks, Vol.2, Part 6 , Orlando, FL, June 27-29,1994,IEEE Piscataway, NJ, 1994,pp. 510-512.

Uhrig, R.E. and Tsoukalas, L.H. , 1997, "Fuzzy and Neural Approaches in Engineering". John Wiley & Sons, New York, USA.

Wang, L. X. e Mendel, J, M., 1992, "Backpropagation Fuzzy Systems as Nonlinear Dynamic System Identifiers", in IEEE International Conference on Fuzzy Systems, San Diego, CA, pp. 1409-1418.

Weber, S. , 1983, A general concept of fuzzy connectives, negations and implications based on t-norm and t-co-norms. Fuzzy Sets and Systems 11,115-134.

Werbos, P. , 1974, " Beyond Regression: New Tool for Prediction and Analysis in the Behavioral Science". Ph.D. thesis, Harvard University, 1974.

Whitely, D. and Starkwerther , 1990, "Optimizing Small Neural Networks Using a Distributed Genetic Algorithm, Proceedings of IJCNN-90-Was.-DC, Washington, DC, January 1990.

Widrow, B. and Hoff, M.E. , 1960, "Adaptive Searching Circuits". In 1960 IRE WESCON Convention Records, 1960.

Willmott, R. , 1980, "Two fuzzier implication operators in the theory of fuzzy power sets". Fuzzy Sets and Systems 20,67-78.

Wu, W. , 1986, "Fuzzy reasoning and fuzzy relational equations". Fuzzy Sets and Systems 20,67-78

Wu, Y. and Xu,B. , 1999, "Study on the Damping Fuzzy Control of Semi-active Suspension". Proceeding in IEEE 0-7803-5296-3/99, pp 66-69.

Yager, R.R. , 1980a, "An approach to inference in approximate reasoning. International Journal of Man-Machine Studies 13,323-338.

Yager, R.R., 1980b, "On a general class of fuzzy connectives". Fuzzy Sets and Systems 4, 235-242.

Yager, R.R. and Filev, D.P., 1993, SLIDE: a simple adaptive defuzzification method. IEEE Transactions on Fuzzy Systems 1(1), 69-78.

Yagishita, O., Itoh, O. and Sugeno, M., 1985, "Application of fuzzy reasoning to the water process, in Industrial Application of Fuzzy Control, M. Sugeno, Ed. Amsterdam: North-Holland, pp 19-40.

Yasunobu, S., Miyamoto, S., 1985, "Automatic train operation by predictive fuzzy control", in Industrial Application of Fuzzy Control, M. Sugeno, Ed. Amsterdam: North-Holland, pp 1-18.

Yasunobu, S., Haswaga, T., 1986, "Evaluation of an Automatic Container Crane operation system based on predictive fuzzy control", Control Theory Advanced Technologies, vol. 2 no. 3, pp. 419-432. T. M. Sugeno, Ed. Amsterdam: North-Holland, pp 1-18.

Yoshimura, T., 1998, "A semi-active suspension of passenger cars using fuzzy reasoning and the field testing". International Journal of Vehicle Design, Vol. 29, No. 02, pp 150-166.

Zadeh, L. A., 1965, "Fuzzy Sets, Information and Control", Vol. 8, pp. 338-353.

Zadeh, L. A., 1968, "Fuzzy Algorithms, Information and Control", Vol. 12, pp. 94-102.

Zadeh, L. A., 1972, "A rationale for fuzzy control". Transactions ASME, Journal of Dynamic System, Measures and Control, vol. 34, 3-4.

Zadeh, L. A., 1973, "Outline of a new approach to the analysis of complex systems and decision processes". IEEE – Transactions on Systems, Man and Cybernetics SMC-3, 28-44.

Zadeh, L. A., 1975, "Calculus of fuzzy restrictions". In L.A. Zadeh, K. S. Fu, K. Tanaka and M. Shimura (Eds.), Fuzzy Sets and their applications to cognitive and decision processes, pp 1-39. New York USA: Academic Press.

Zimmermann, H.J., 1985, "Fuzzy Set Theory and its applications", Computer and People Series, pp. 1-66. Academic Press.

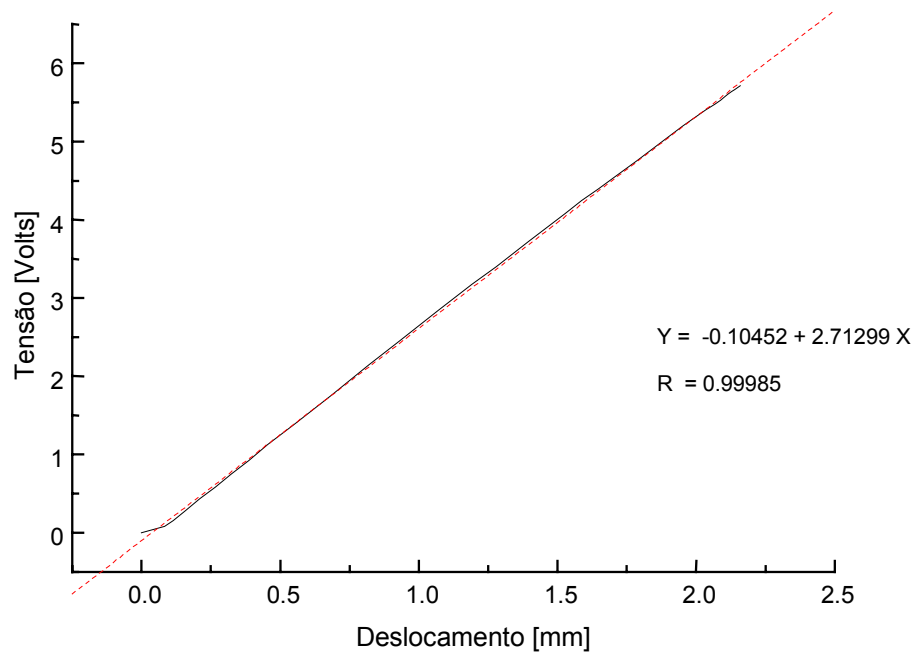
Apêndice I

Componentes do sistema mecânico de 1 grau de liberdade

Equipamento	Fabricante	Especificações
Sensor de deslocamento eletromagnético	DYMAC	
Martelo de Impacto B & K	Brüel & Kjaer	Impact Hammer Type 8202 Force Transducer Type 8200
Condicionador de Sinais do Martelo de Impacto	Brüel & Kjaer	Charge Amplifier Type 2635 [1,02 pC / N] e [31,6 mV / N] Duração da martelada de 1 a 5 ms Excita de 0 - 500 Hz
2 Fontes de Potência		EMG 18131 [30 Volts ; 2,5 A]
Fonte de Corrente	TECTROL	Modelo TC 50-02 [50 V ; 2 A]
Osciloscópio	HP - Hewlett Packard	Oscilloscope 54603B
Placa de aquisição	UEI - United Electronic Industries	12 Bits
Analizador de Sinais.	Scientific-Atlanta Spectral Dynamics	SD380 Signal analyser Banda de análise 0- 100 Hz Janela retangular - 800 linhas Trigger externo Passo em frequência = 0.125 Hz
Condicionador de Sinais Sensor de deslocamento eletromagnético	Laboratório de Sistemas Mecânicos	
Atuador Eletromagnético	Laboratório de Sistemas Mecânicos	
Driver de Corrente	Laboratório de Sistemas Mecânicos	

I.I Curva de calibração do sensor de deslocamento eletromagnético

O sensor de proximidade, do tipo DYMAC, detecta as posições da mesa. Ele não apresenta contato entre as partes. É um transdutor linear do tipo deslocamento-tensão, conforme mostra sua curva de calibração estática.



I.II Martelo de impacto utilizado nos ensaios para caracterização dos controladores no domínio da frequência.



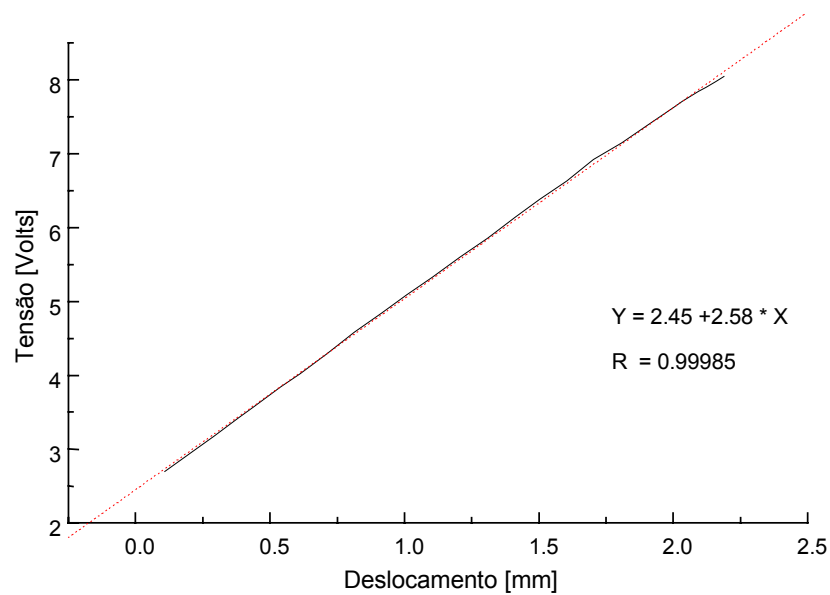
Apêndice II

Componentes do sistema mecânico com infinitos graus de liberdade

Equipamento	Fabricante	Especificações
Sensor de deslocamento eletromagnético	DYMAC	
Pêndulo de Impacto B & K	Brüel & Kjaer	Force Transducer Type 8200
Condicionador de Sinais Do Pêndulo de Impacto	Brüel & Kjaer	Charge Amplifier Type 2635 [3,82 pC / N] e [316 mV / N] Duração da martelada de 1 a 5 ms Excita de 0 - 500 Hz
Atuador Piezelétrico	ACX	QP10N Máxima Tensão [150 V]
Amplificador de sinal Do atuador piezelétrico	AVC Instrumentation	Power amplifier 790 series [30 V / V]
Osciloscópio	HP - Hewlett Packard	Oscilloscope 54603B
Placa de aquisição	UEI - United Eletronic Industries	12 Bits
Analizador de Sinais.	Scientific-Atlanta Spectral Dynamics	SD380 Signal analyser Banda de análise 0- 500 Hz Janela retangular - 800 linhas Trigger externo Passo em frequência = 0.625 Hz
Condicionador de Sinais Sensor De deslocamento eletromagnético	Laboratório de Sistemas Mecânicos	

II.I Curva de calibração do sensor de deslocamento eletromagnético

Uma nova calibração foi necessária pois neste experimento com a viga o sensor de proximidade detecta um outro tipo de material (aço inoxidável).

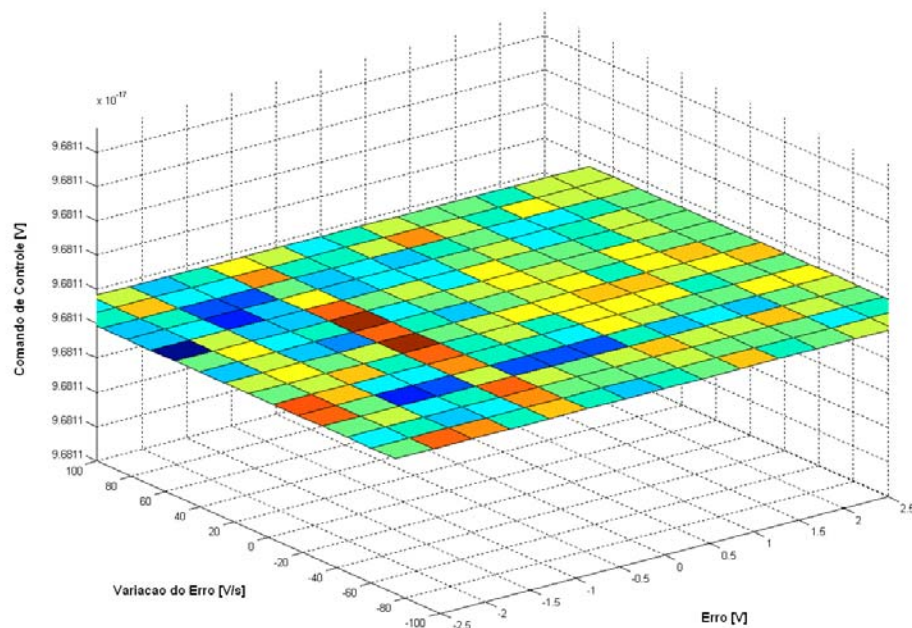


II.II Pêndulo de impacto utilizado nos ensaios para caracterização dos controladores no domínio da frequência.

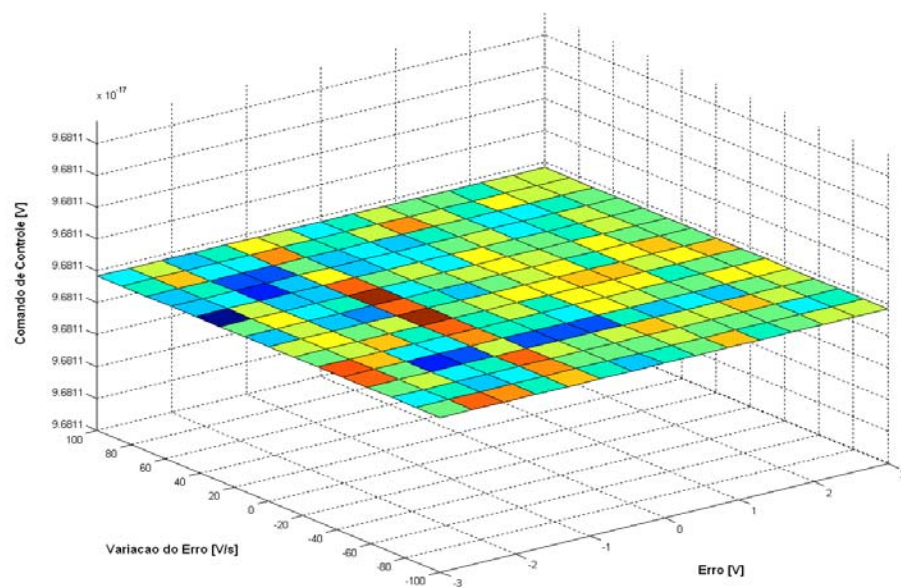


Apêndice III

Superfície dos Controladores Iniciais submetidos à Otimização



III.I Superfície de Controle do Controlador Inicial do Experimento do Sistema de 1 GDL.



III.II Superfície de Controle do Controlador Inicial do Experimento do Sistema de n GDL.

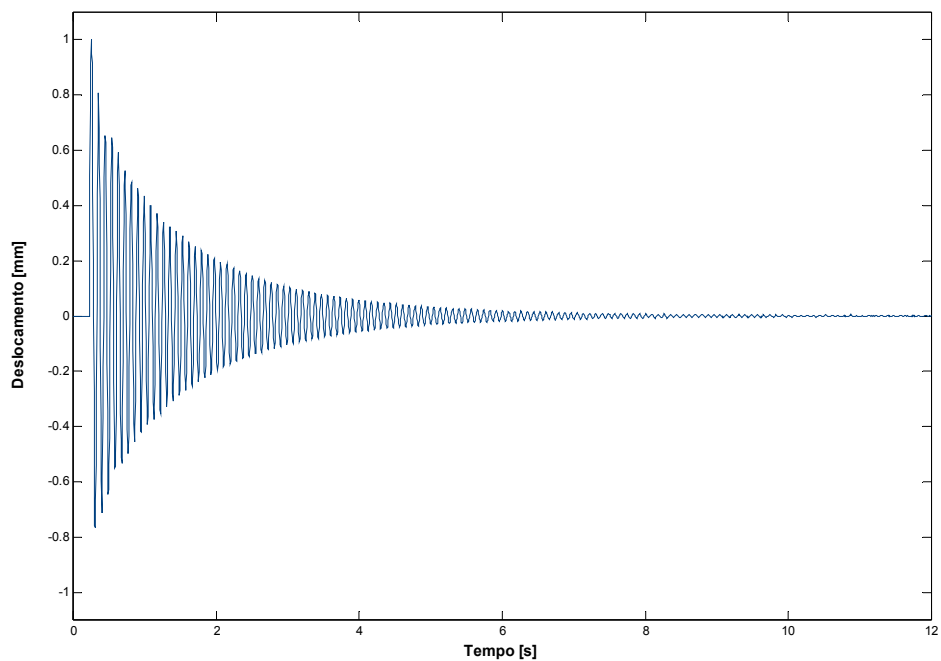
Apêndice IV

Respostas no Domínio do Tempo e Frequência dos Controladores do Capítulo 6

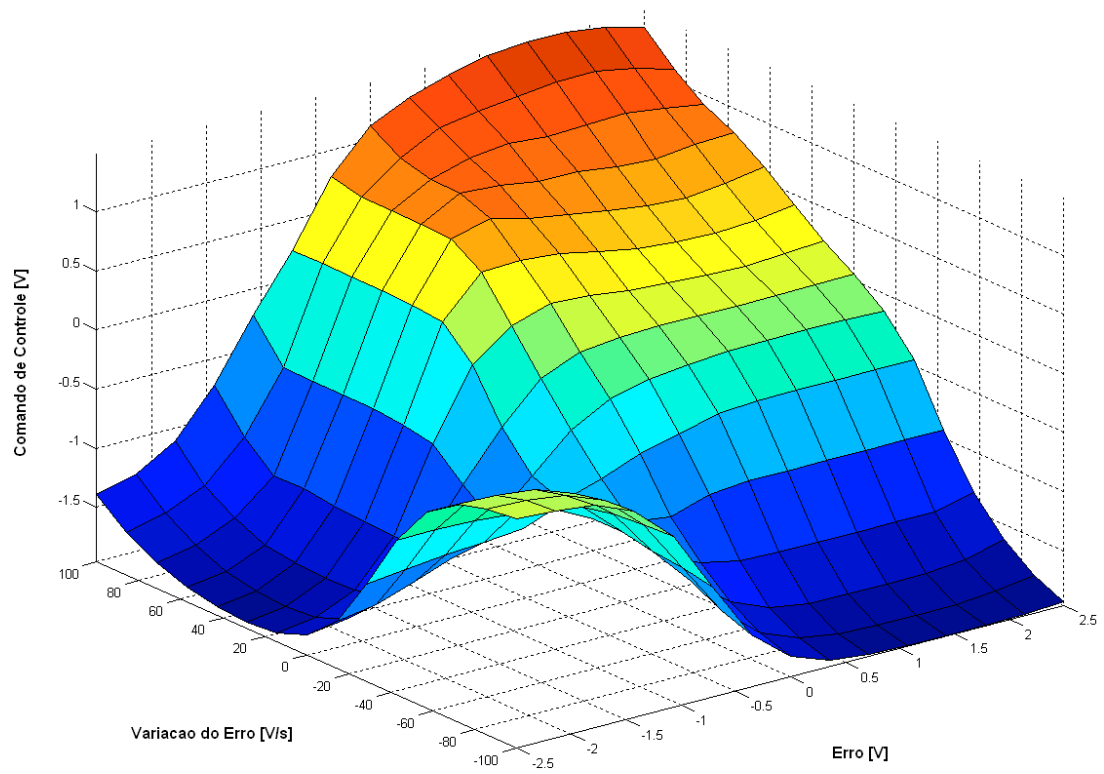
IV.I Controladores *fuzzy* otimizados para o sistema de 1 GDL

Controlador	Estratégia de otimização	ξ
M-livre	-	0,0193
M1	Conseqüente da regras	0,1375
M2	Pesos (base de regras completa)	0,2510
M3	Funções de pertinência das entradas (centro)	0,2895
M4	Pesos (base de regras reduzida a 6)	0,2506

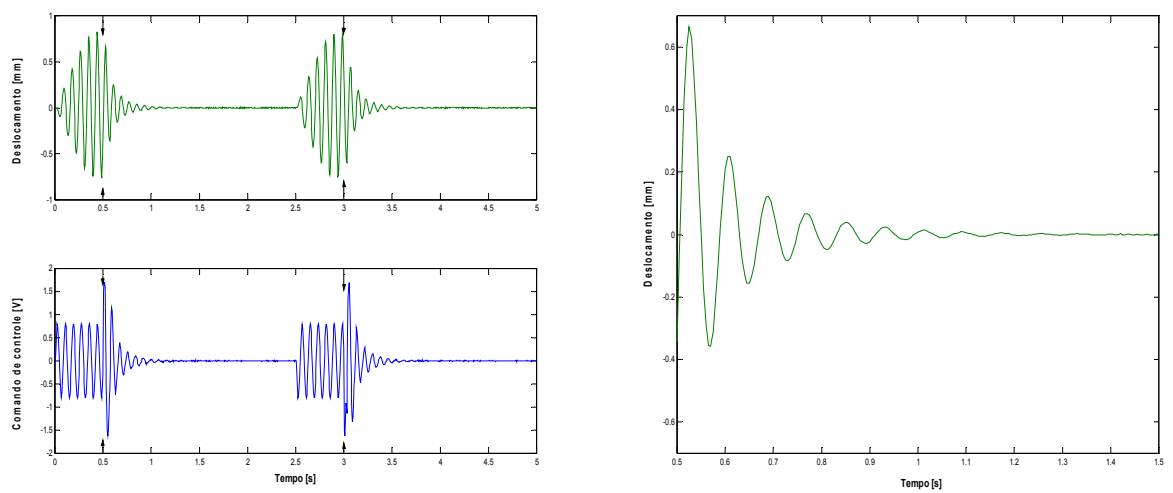
IV.II Resposta Livre do sistema de 1 GDL



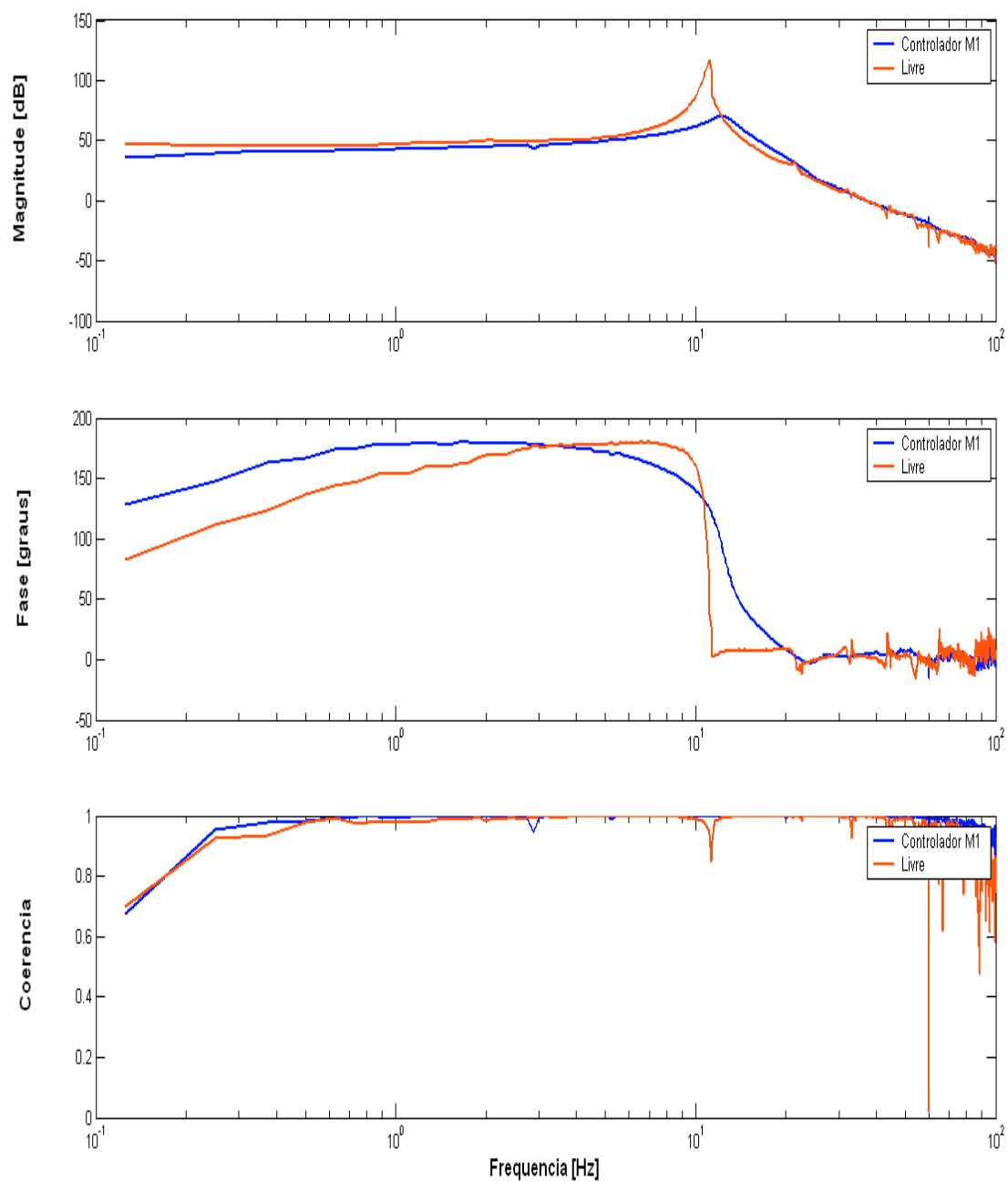
IV.III Superfície de Controle do Controlador M1



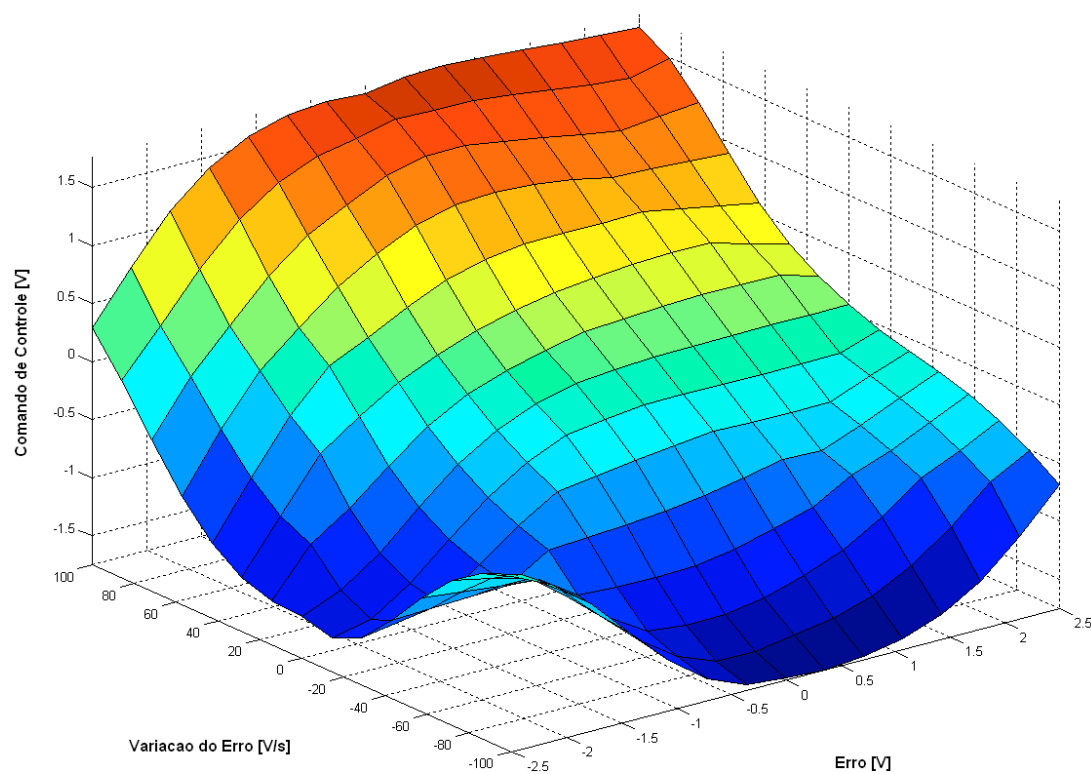
IV.IV Resposta temporal do Controlador M1



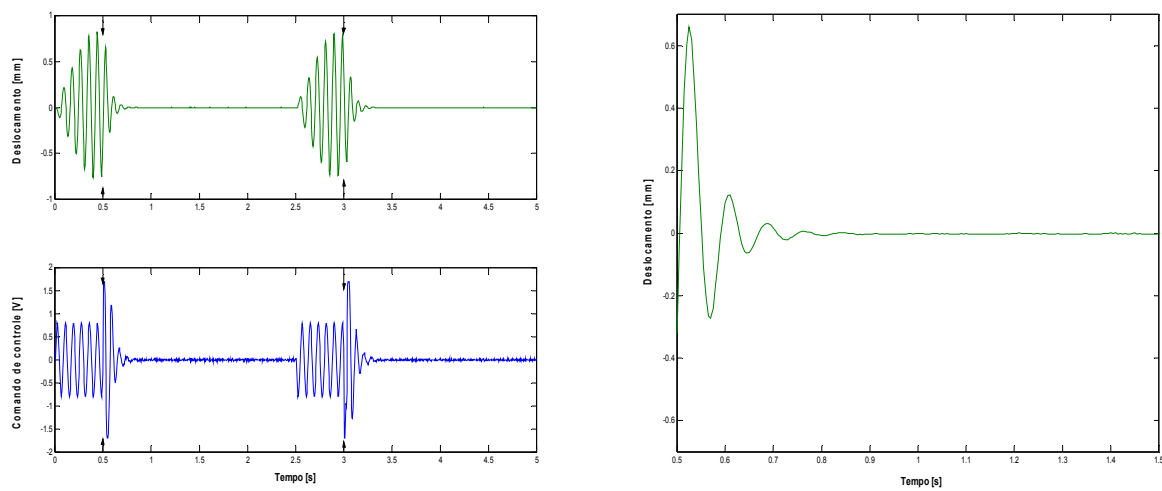
IV.V Resposta freqüencial do Controlador M1



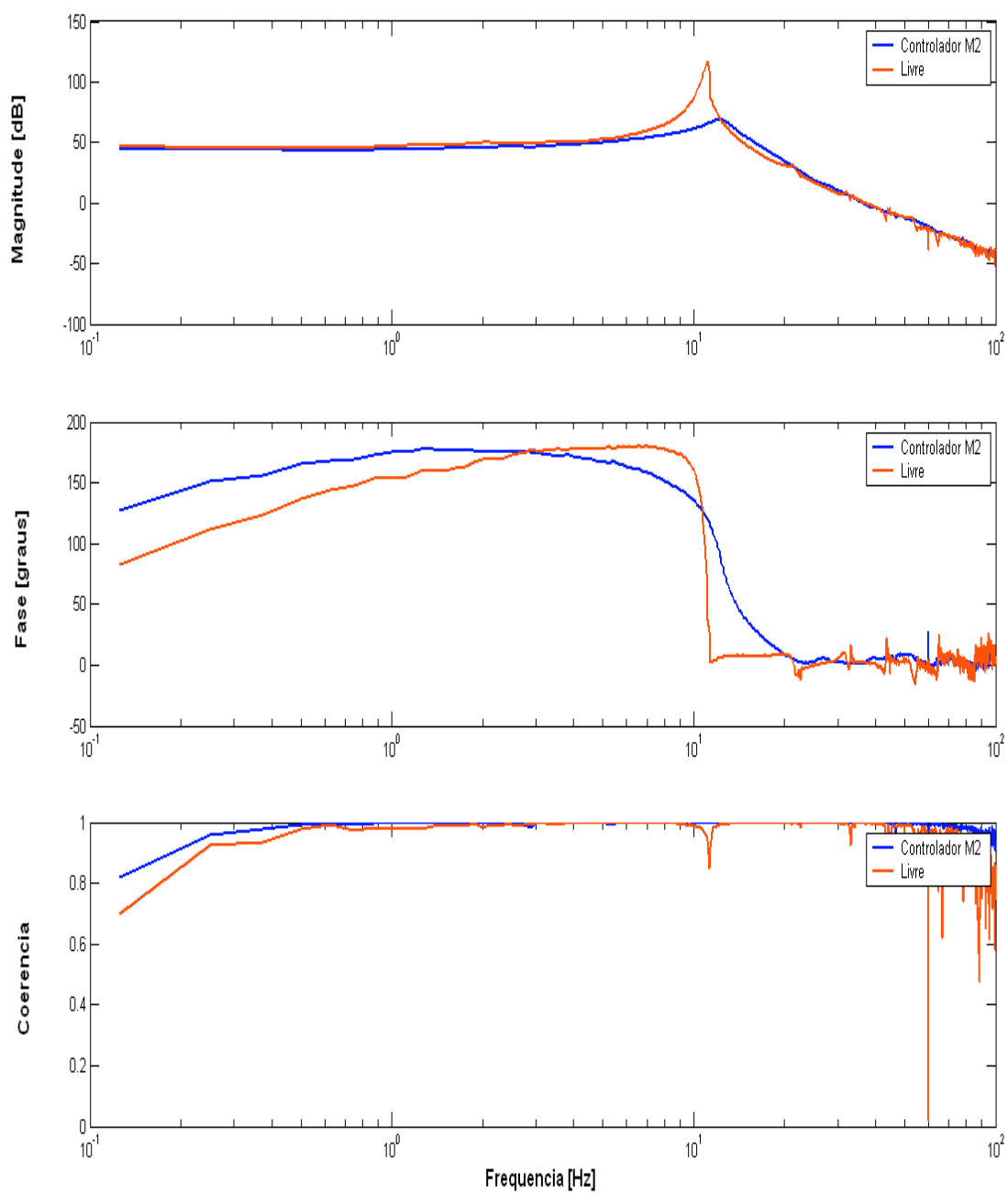
IV.VI Superfície de Controle do Controlador M2



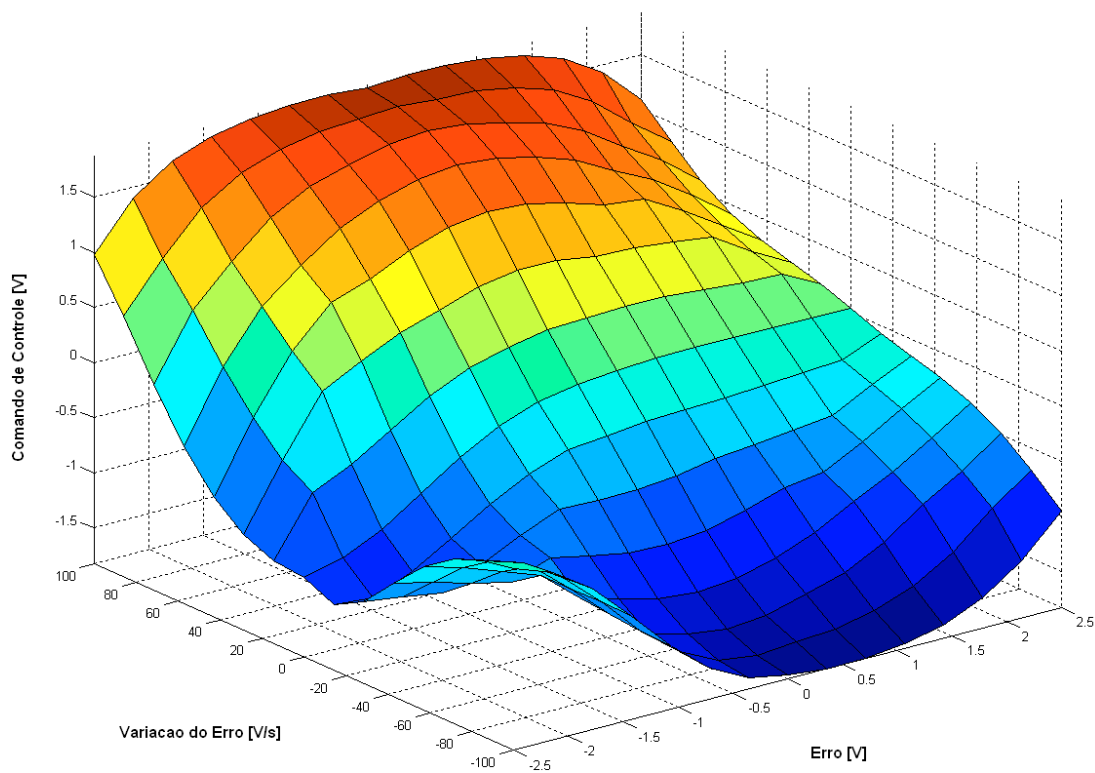
IV.VII Resposta temporal do Controlador M2



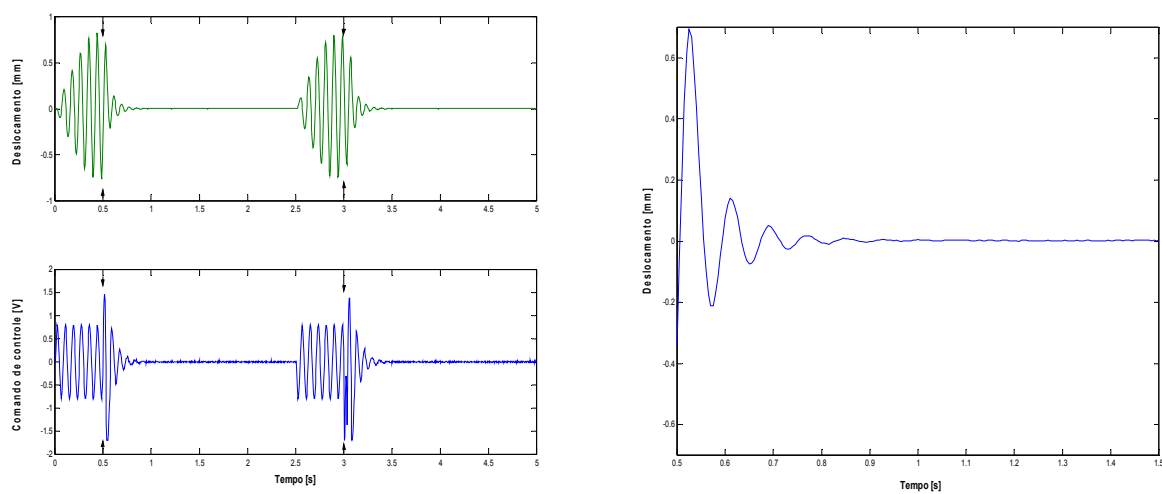
IV.VIII Resposta freqüencial do Controlador M2



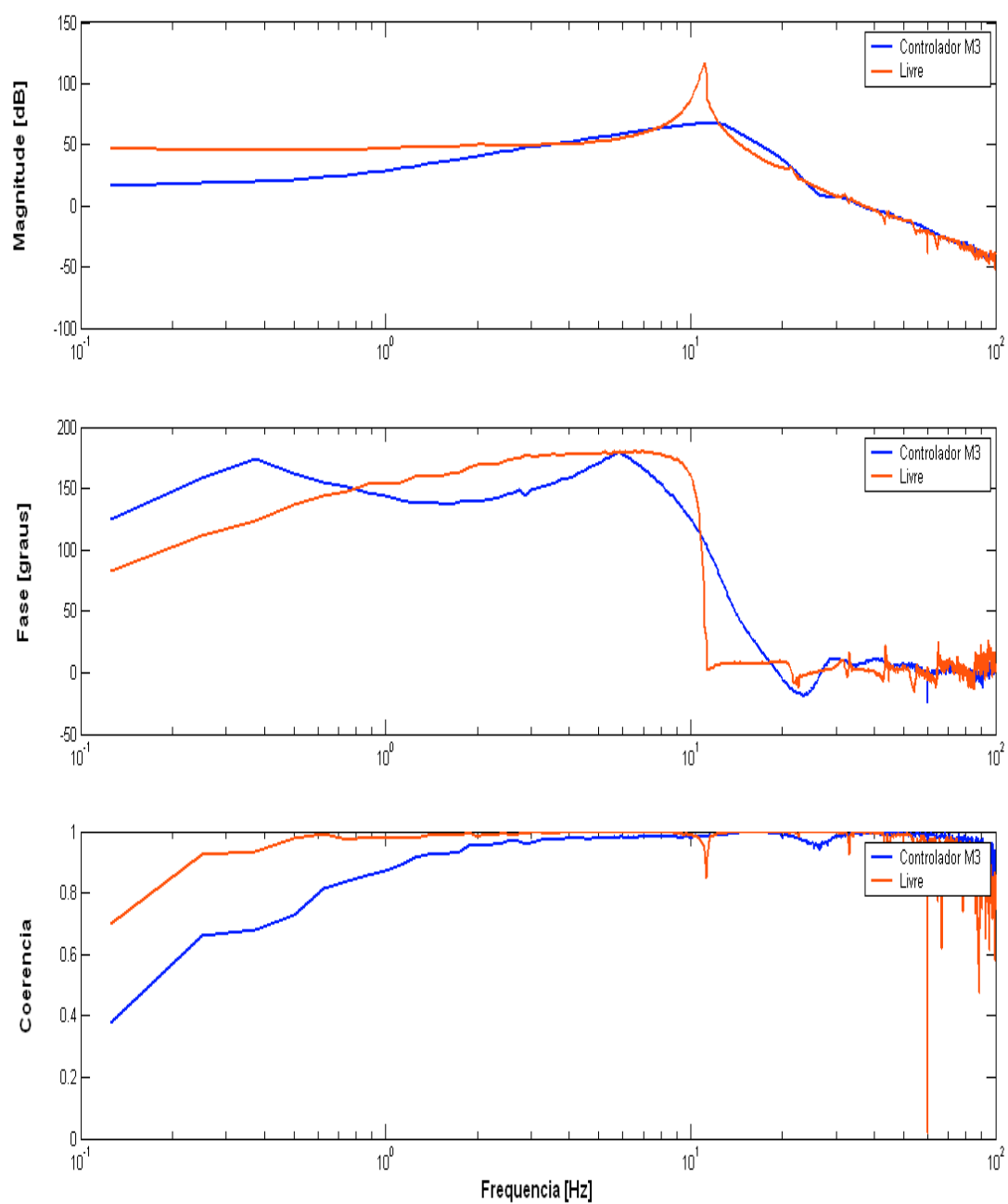
IV.IX Superfície de Controle do Controlador M3



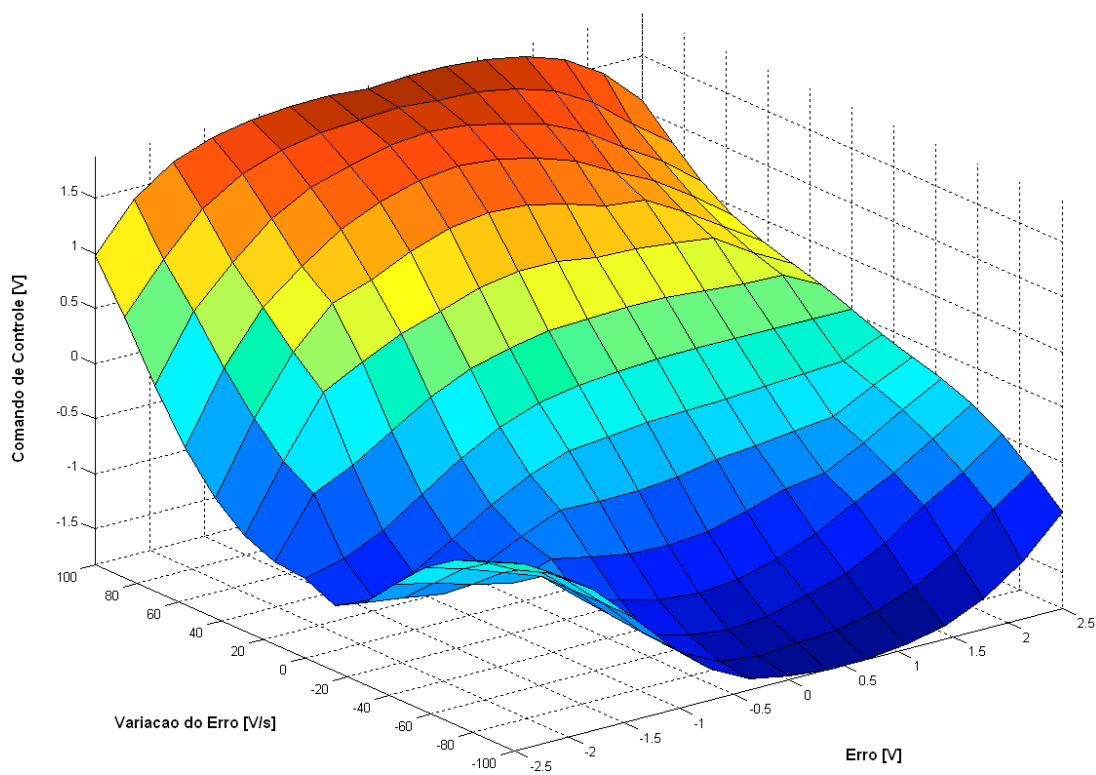
IV.X Resposta temporal do Controlador M3



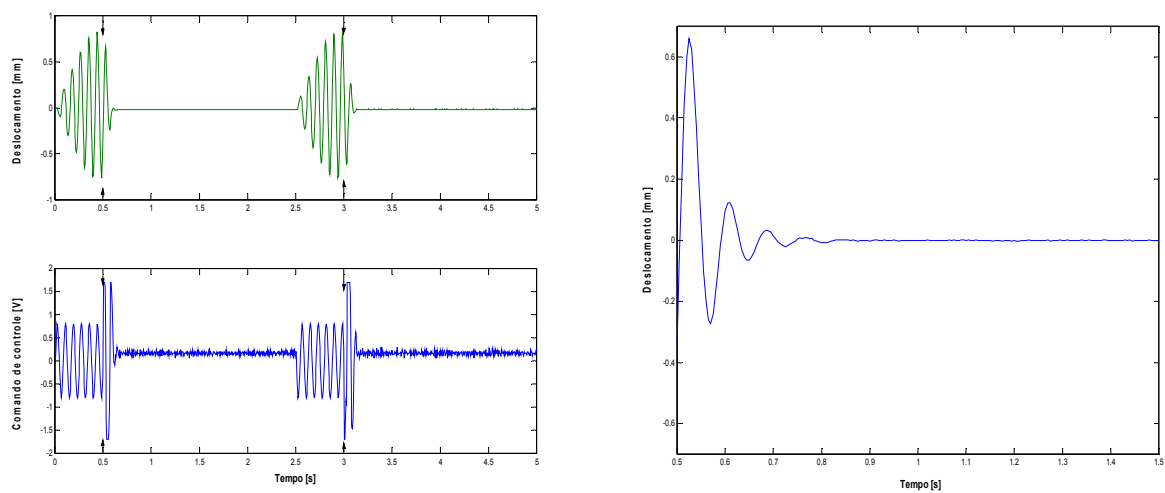
IV.XI Resposta freqüencial do Controlador M3



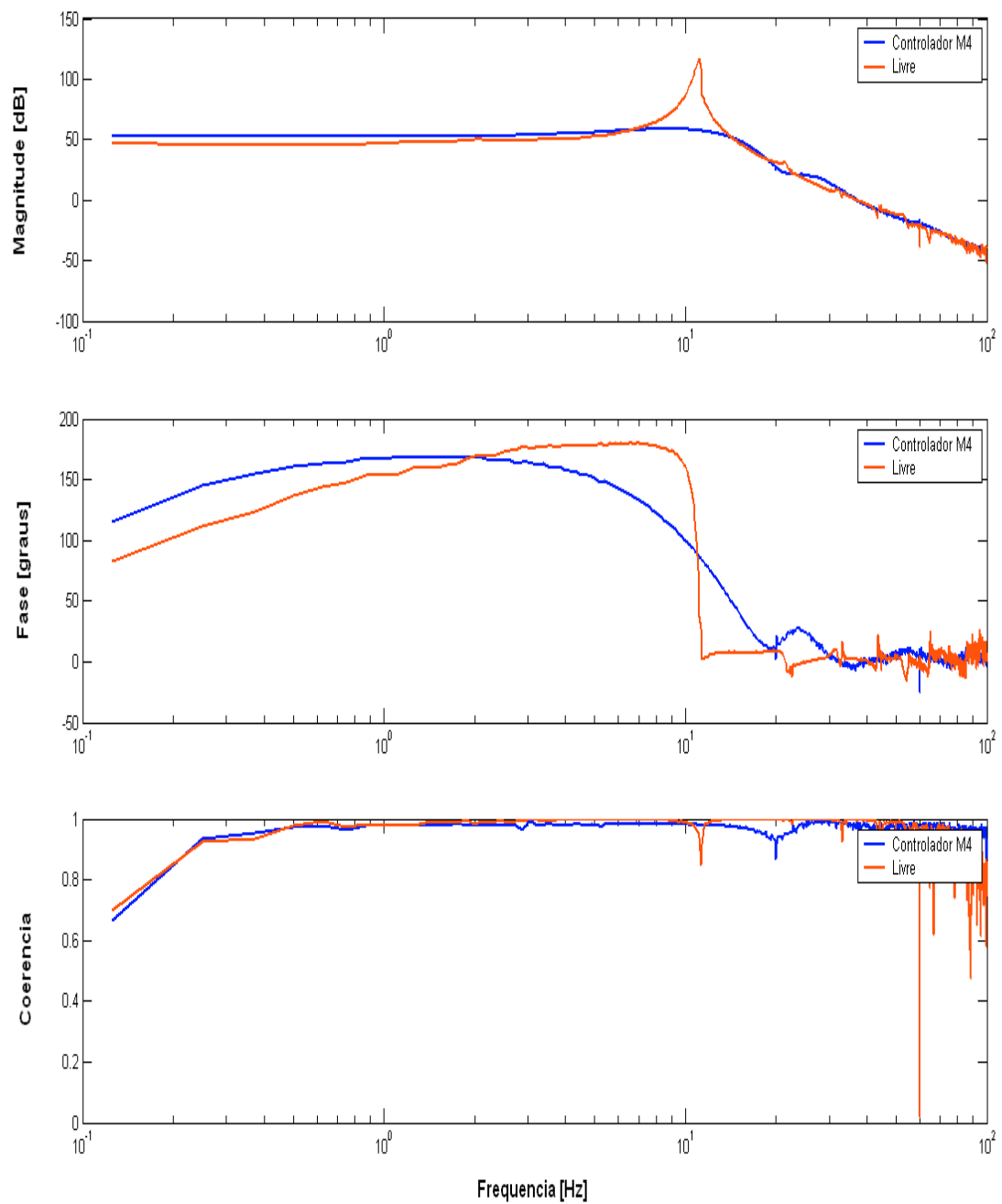
IV.XII Superfície de Controle do Controlador M4



IV.XIII Resposta temporal do Controlador M4



IV.XIV Resposta freqüencial do Controlador M4



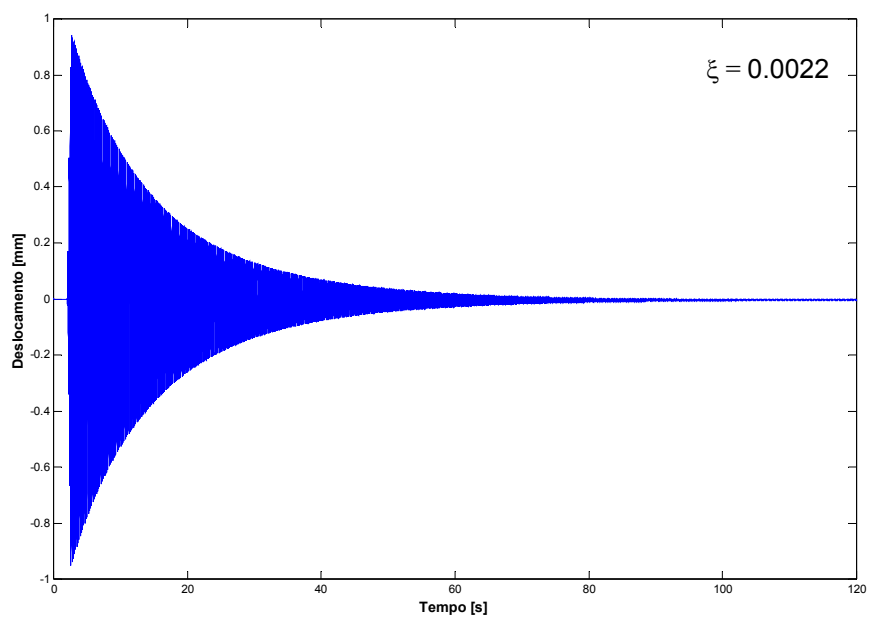
Apêndice V

Respostas no Domínio do Tempo e Frequência dos Controladores do Capítulo 7

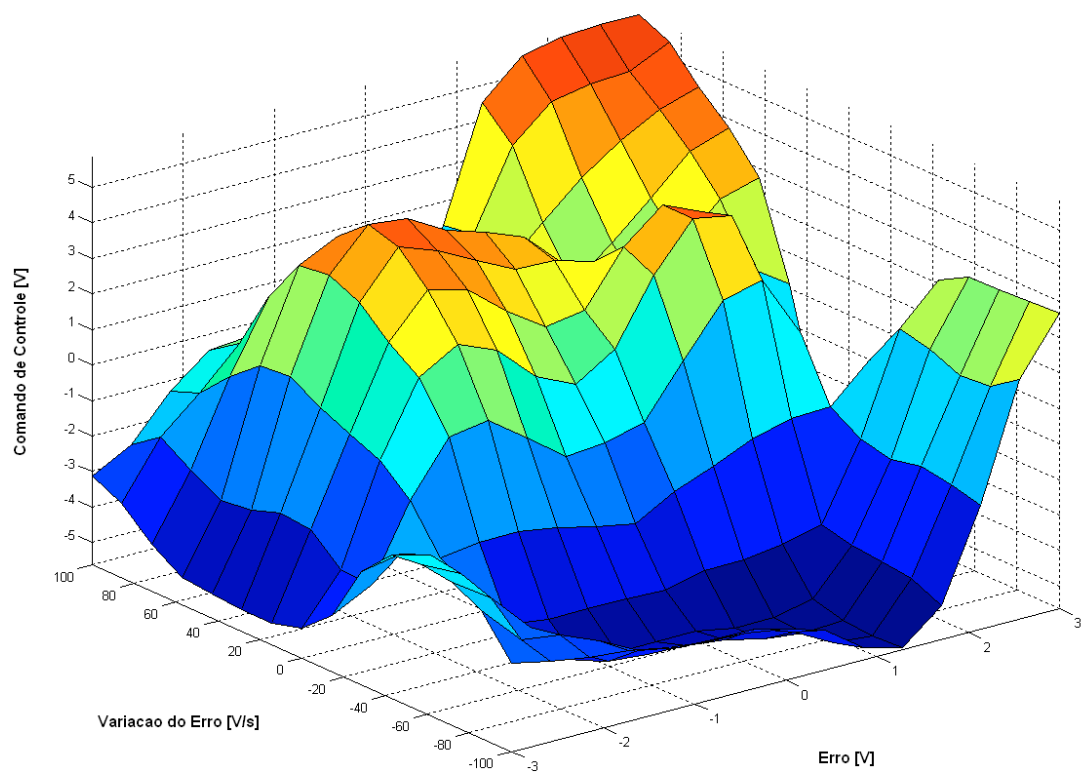
V.I Controladores *fuzzy* otimizados para o sistema de n GDL

Controlador	Estratégia de otimização	Amortecimento (ξ)
V-livre	-	0,0022
V1	Conseqüente da regra	0,0667
V2	Pesos (base de regras completa)	0,0919
V3	Funções de pertinência das entradas (centro)	0,0967
V4	Pesos (base de regras reduzida a 13)	0,0927
V5	Funções de pertinência das entradas (centro)	0,0997

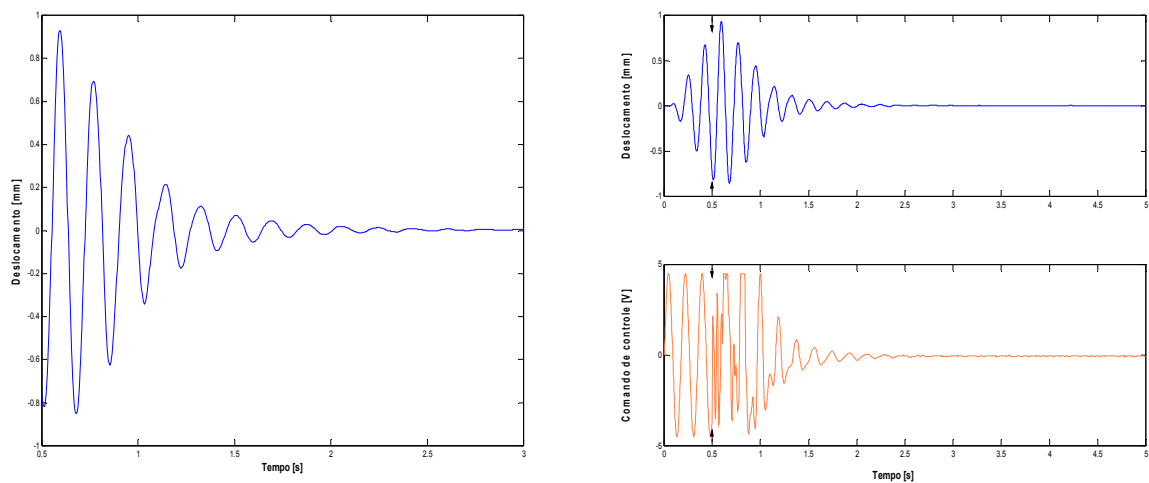
V.II Resposta Livre do sistema de n GDL



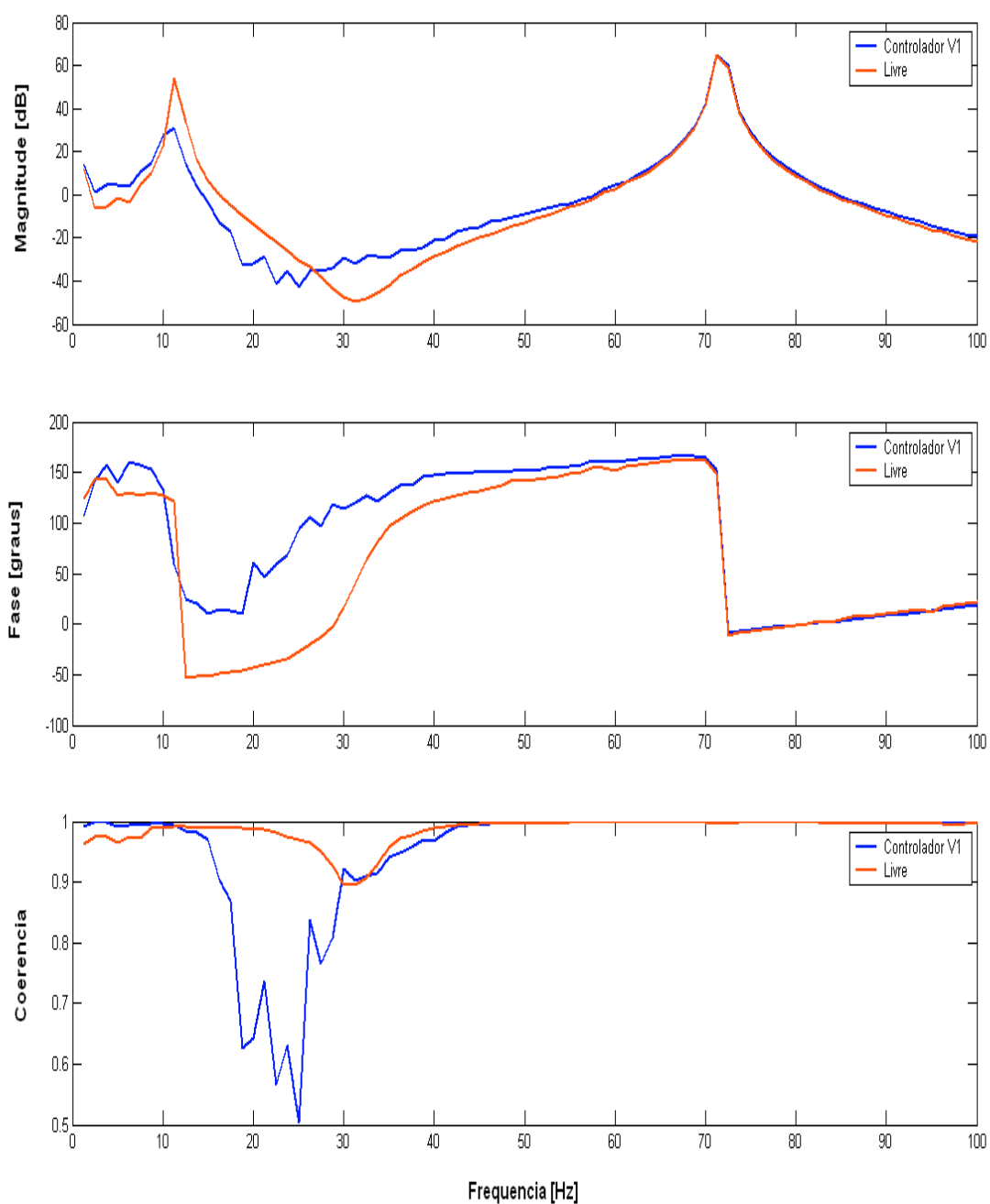
V.III Superfície de Controle do Controlador V1



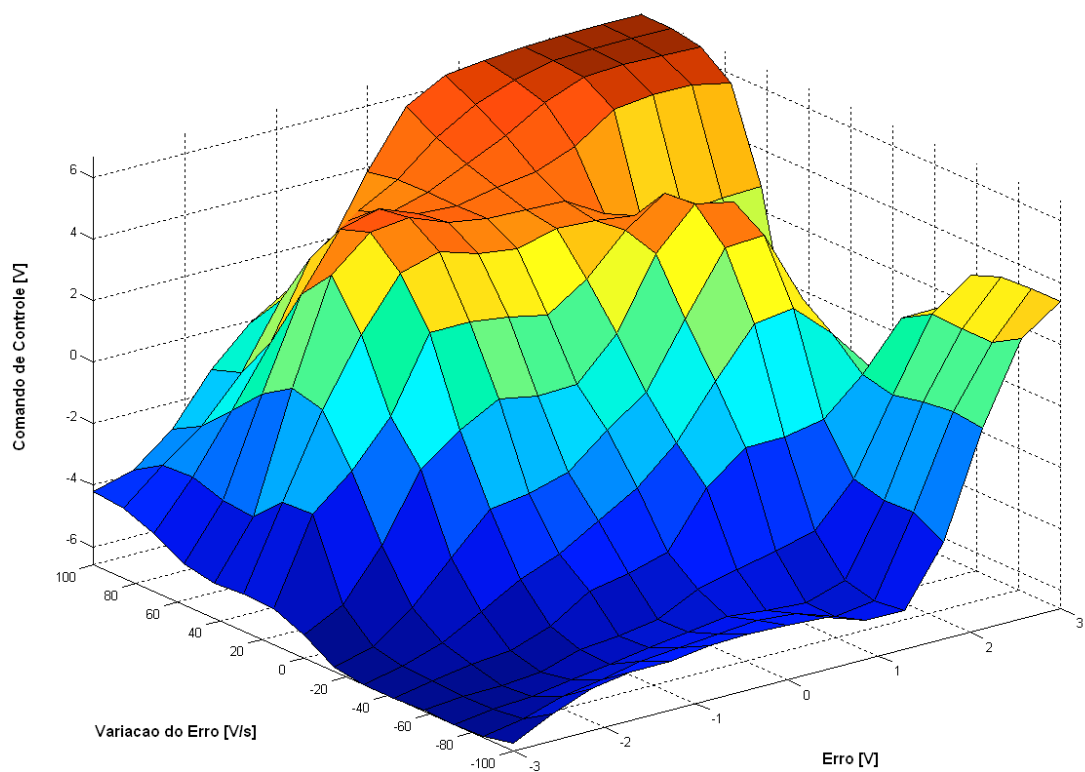
V.IV Resposta temporal do Controlador V1



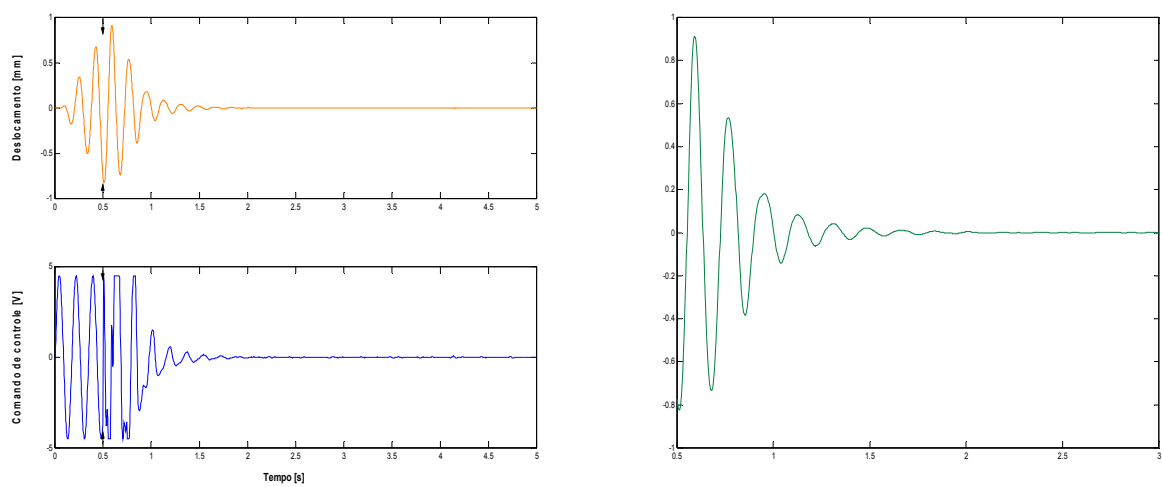
V.V Resposta freqüencial do Controlador V1



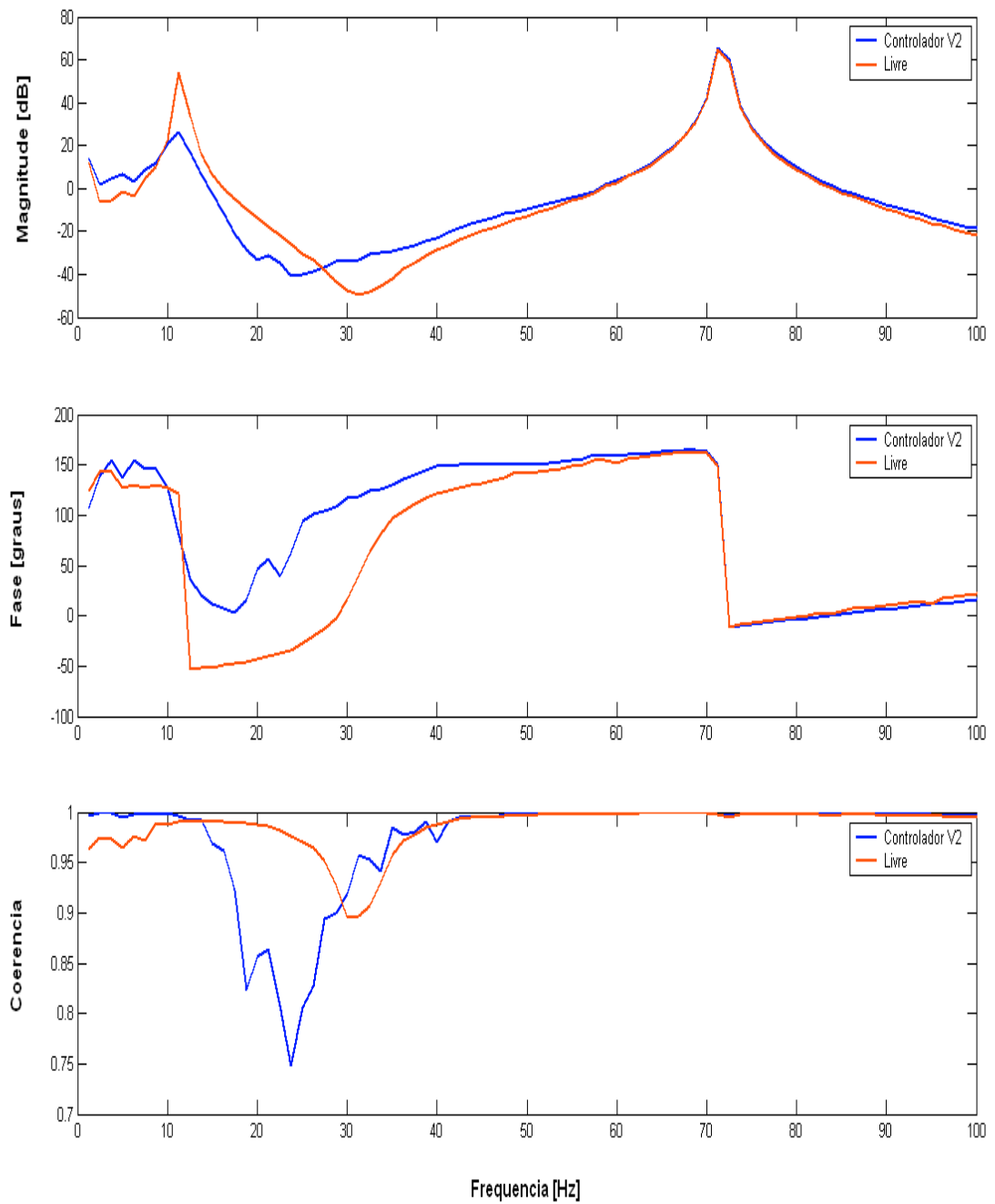
V.VI Superfície de Controle do Controlador V2



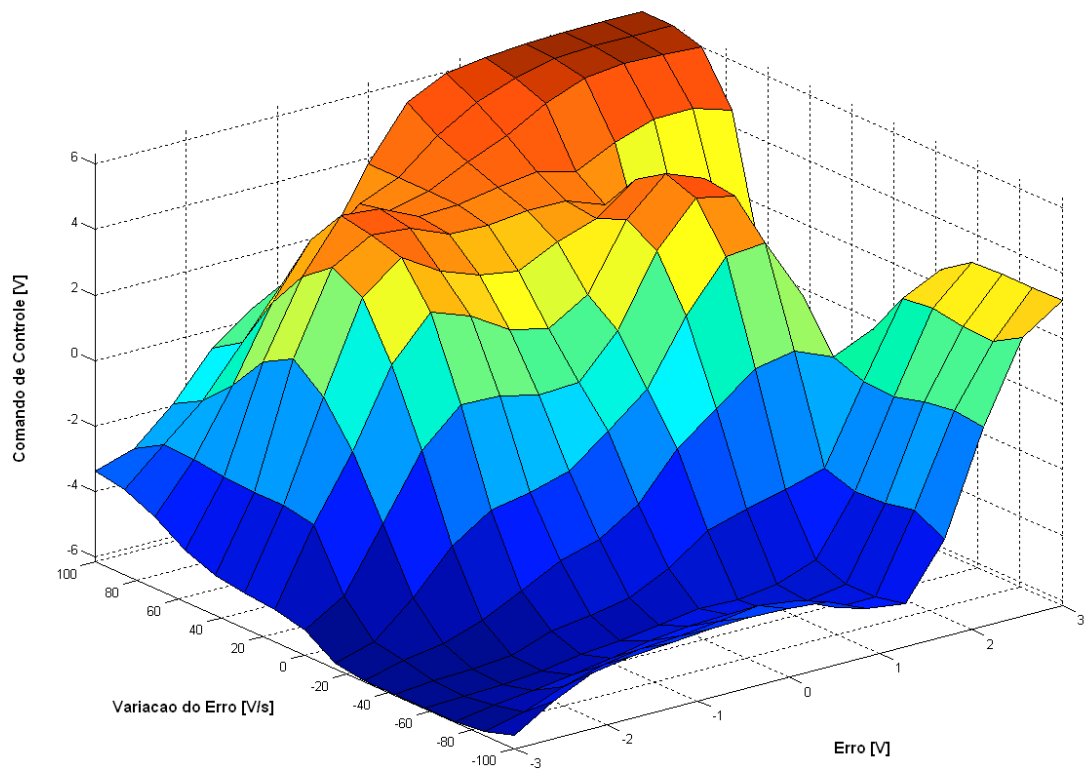
V.VII Resposta temporal do Controlador V2



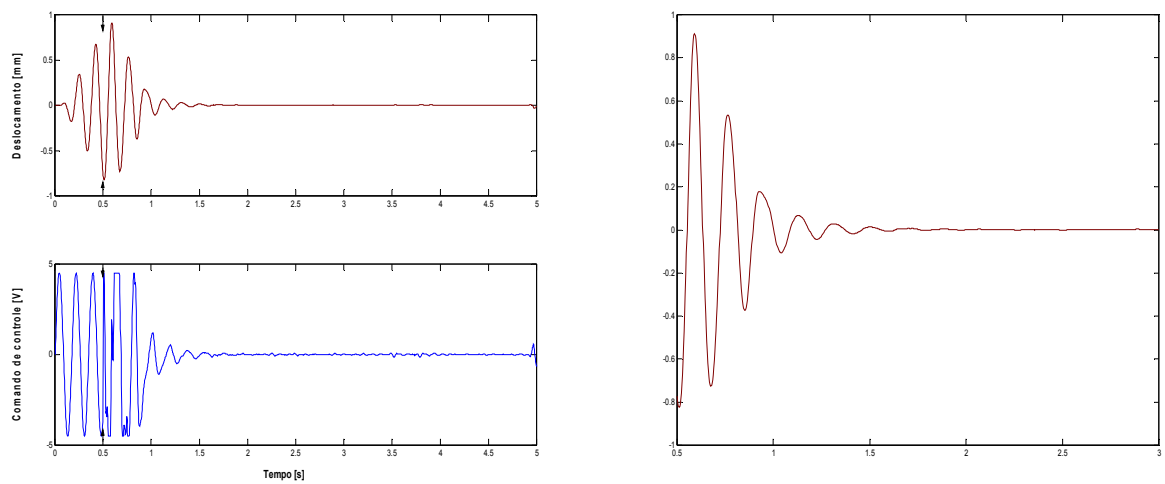
V.VIII Resposta freqüencial do Controlador V2



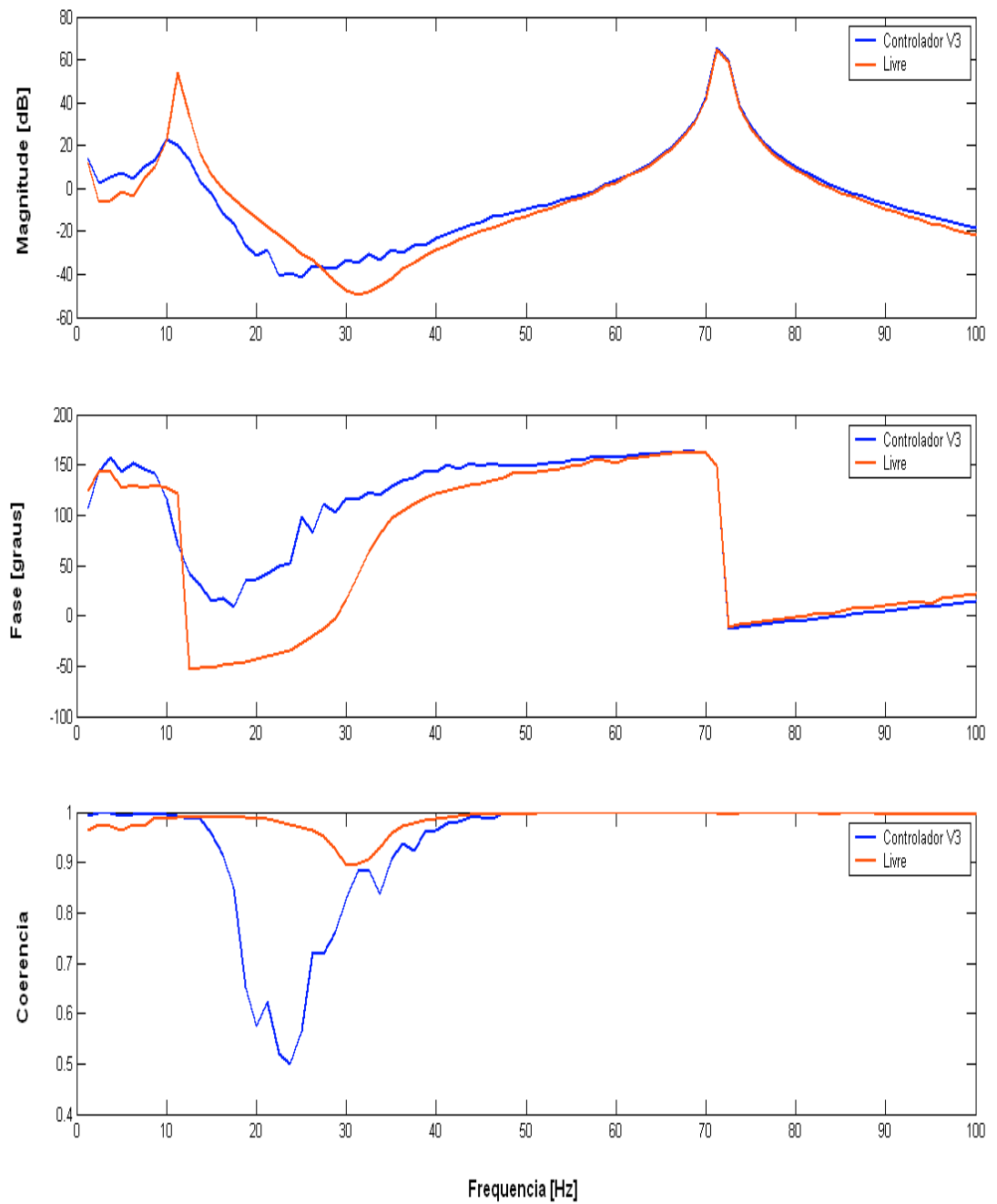
V.IX Superfície de Controle do Controlador V3



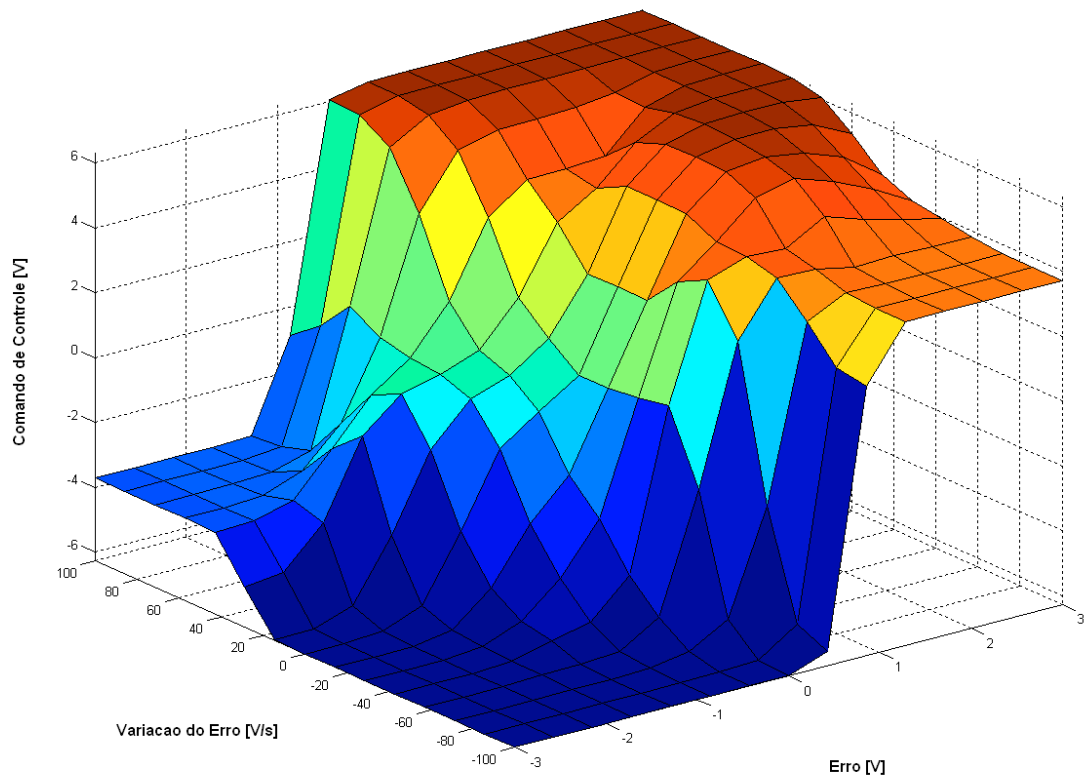
V.X Resposta temporal do Controlador V3



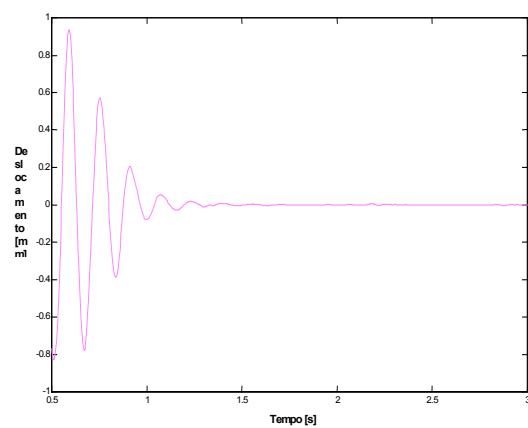
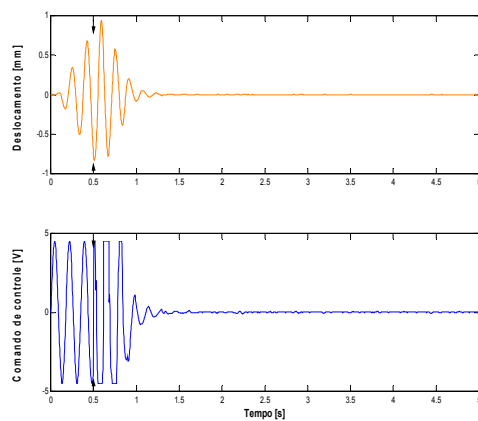
V.XI Resposta freqüencial do Controlador V3



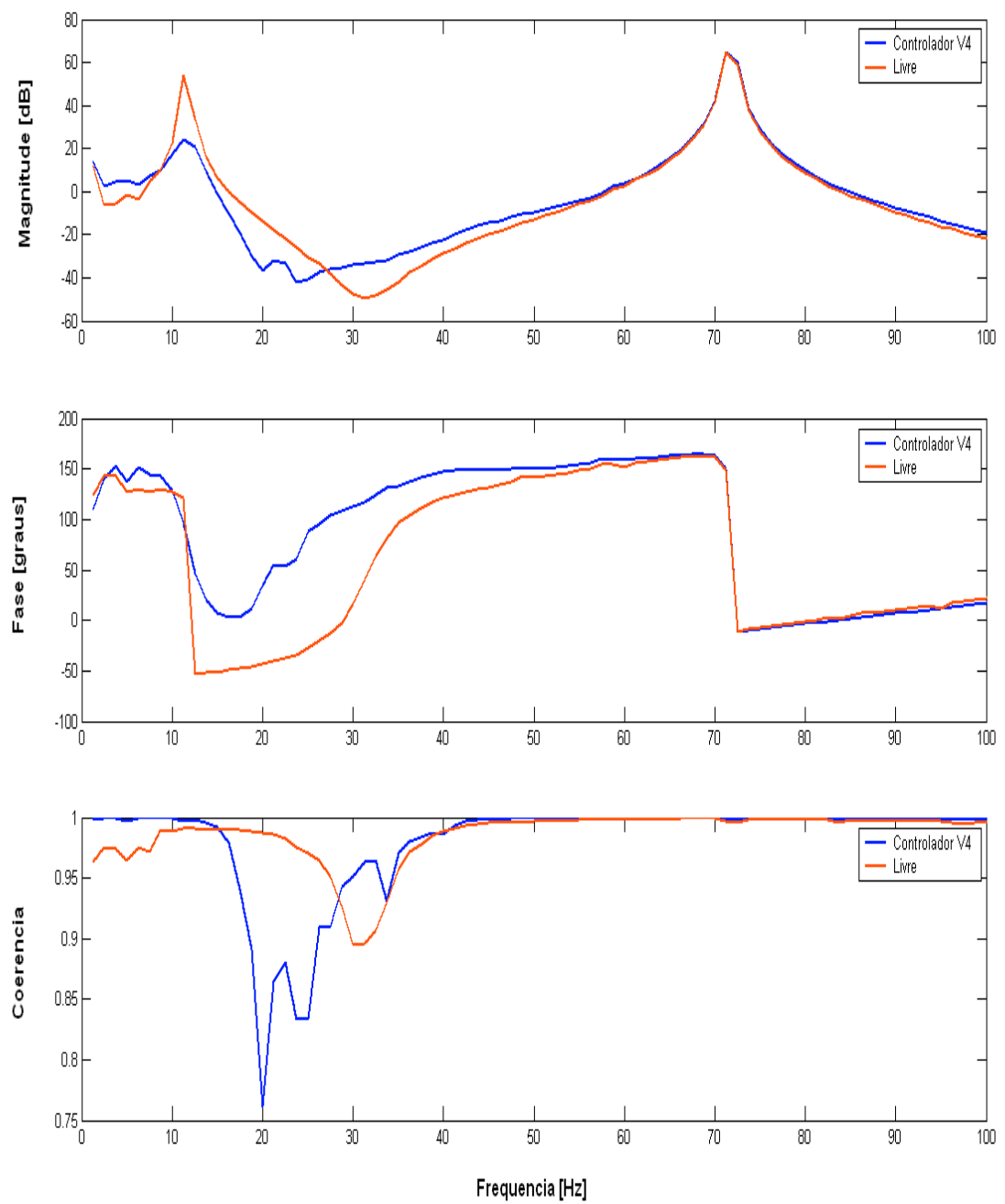
V.XII Superfície de Controle do Controlador V4



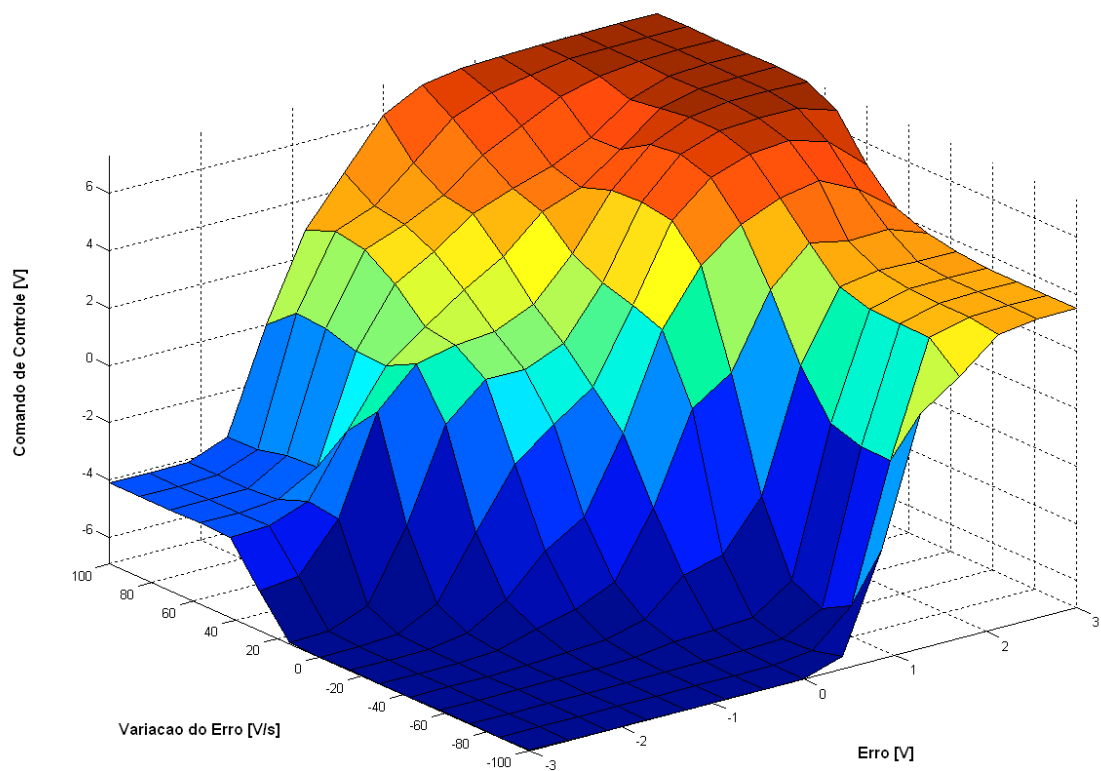
V.XIII Resposta temporal do Controlador V4



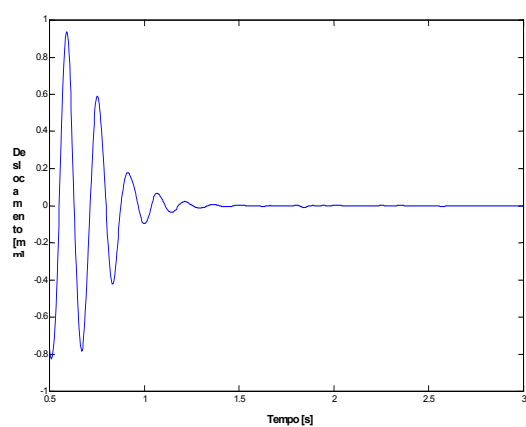
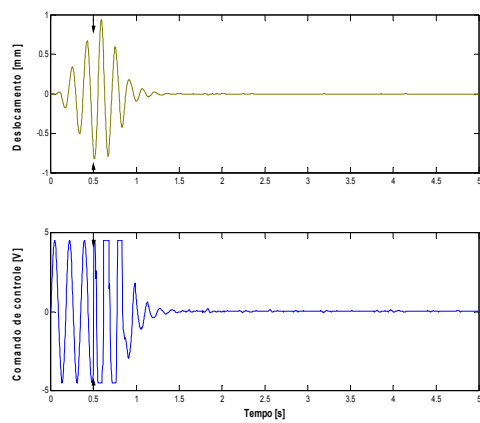
V.XIV Resposta freqüencial do Controlador V4



V.XV Superfície de Controle do Controlador V5



V.XVI Resposta temporal do Controlador V5



V.XVII Resposta freqüencial do Controlador V5

