MODELOS NEURO-FUZZY PARA IDENTIFICAÇÃO DE SISTEMAS APLICADOS À OPERAÇÃO DE CENTRAIS NUCLEARES

Antonio Carlos Pinto Dias Alves

TESE SUBMETIDA AO CORPO DOCENTE DA COORDENAÇÃO DOS PROGRAMAS DE PÓS-GRADUAÇÃO DE ENGENHARIA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE DOUTOR EM CIÊNCIAS EM ENGENHARIA NUCLEAR.

Aprovada por:

Prof. Roberto Schirru, D.Sc.

Prof. Aquilino Senra Martinez, D.Sc.

Dr. Marco Antonio Bayout Alvarenga, D.Sc.

Dr. Cláudio Márcio Nascimento Abreu Pereira, D.Sc.

Dr. Alexandre Gonçalves Evsukoff, Dr.

RIO DE JANEIRO, RJ - BRASIL

SETEMBRO DE 2000

ALVES, ANTONIO CARLOS PINTO DIAS

Sistemas Neuro-Fuzzy para Identificação de Sistemas Aplicados à Operação de Centrais Nucleares [Rio de Janeiro] 2000

IX, 165 p. 29,7 cm (COPPE/UFRJ, D. Sc., Engenharia Nuclear, 2000)

Tese - Universidade Federal do Rio de Janeiro, COPPE

1 - Modelos Neuro-nebulosos

2 - Controle

3 - Operação de Centrais Nucleares

I. COPPE/UFRJ II. Título (série)

Agradecimentos

Várias pessoas contribuíram para que este trabalho fosse feito. Agradeço ao professor e amigo Aquilino Senra Martinez pelas suas aulas, conselhos, por me ter oferecido o problema de controle nuclear abordado nesta tese e, principalmente pela sua amizade da qual tantas vezes me vali, sem ter a oportunidade de retribuir.

A Alexandre Evsukoff, colega e colaborador pela sessão de seu código e idéias e que, mesmo em prejuízo de suas atividades, sempre encontrou tempo para esclarecer várias dúvidas me indicando muitas vezes o caminho das pedras.

Aos colegas mestrandos, doutorandos e funcionários do LMP que nunca se furtaram de me ajudar, ora cedendo seus micros, ora discutindo problemas técnicos, algumas vezes, até me cedendo suas senhas. Mas, principalmente, por seu apoio e incentivo.

A todos os professores e funcionários do PEN pelo seu incentivo e pela solução de problemas práticos, tarefa na qual a professora Vergínia e a secretária Jô foram as que mais requisitei ao longo de todos esses anos.

Finalmente, deixo um agradecimento especial ao professor Roberto Schirru, verdadeira inspiração como ser humano, maior responsável pela existência desta tese. Não só por sua orientação durante toda a realização do trabalho de pesquisa, mas também pela dedicação, apoio, amizade e confiança indispensáveis com os quais ele me presenteou não só ao longo de todo este trabalho mas desde que o conheço.

Resumo da Tese apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Doutor em Ciências (D.Sc.)

SISTEMAS NEURO-FUZZY PARA IDENTIFICAÇÃO DE SISTEMAS APLICADOS À OPERAÇÃO DE CENTRAIS NUCLEARES

Antonio Carlos Pinto Dias Alves

Setembro/2000

Orientador: Roberto Schirru

Programa: Engenharia Nuclear

Uma central de energia nuclear possui vários sistemas e sub-sistemas complexos que, interligados e cooperativos, promovem o controle da usina. Ainda que sua operação seja automática a maior parte do tempo, a compreensão integral de sua lógica de funcionamento pode escapar mesmo aos operadores mais experientes devido à pouca interpretabilidade que tais controles oferecem. Essa dificuldade não ocorre apenas nas centrais nucleares mas em qualquer sistema de controle mais complexo. Os modelos neuro-nebulosos vem sendo utilizados há alguns anos na tentativa de superar essas dificuldades devido à sua capacidade de modelar de forma lingüística mesmo sistemas cujos comportamentos sejam extremamente complexos. Essa é uma forma humanamente bastante intuitiva de interpretação e esses modelos vem tendo crescente aceitação. Ocorre que mesmo os modelos neuro-nebulosos podem crescer até se tornarem de difícil interpretação, em função da complexidade dos sistemas sob modelagem. Geralmente, esse crescimento ocorre em função ou de regras redundantes ou de regras que cobrem um domínio muito pequeno do problema. Este trabalho apresenta um método de identificação de modelos neuro-nebulosos que permite não só que os modelos cresçam em função da complexidade existente mas que, antes, eles procurem se auto-adaptar para evitar a inclusão de novas regras. Essa abordagem permitiu chegar a modelos neuro-nebulosos de altas qualidades interpretativas mesmo de sistemas bastante complexos. O uso dessa modelagem ao controle do pressurizador de uma usina PWR permitiu verificar sua validade e como os modelos neuro-nebulosos assim construídos podem ser úteis no entendimento da operação automática de uma central nuclear.

Abstract of Thesis presented to COPPE/UFRJ as a partial fulfilment of the requerements for degree of Doctor of Science (D.Sc.)

NEURO-FUZZY MODELS FOR SYSTEM IDENTIFICATION APPLIED TO THE OPERATION OF NUCLEAR POWER PLANTS

Antonio Carlos Pinto Dias Alves

September/2000

Advisor: Roberto Schirru

Department: Nuclear Engineering

A nuclear power plant has a miriad of complex systems and sub-systems that, working cooperatively, make the control of the whole plant. Nevertheless their operation be authomatic most of the time, the integral understanding of their internal logic can be away of the comprehension of even experienced operators because of the poor interpretability those controls offer. This difficulty does not happens only in nuclear power plants but in almost every a little more complex control system. Neuro-fuzzy models have been used for the last years in a attempt of surpass these difficulties because of their ability of modelling in linguistic form even a system which behavior is extremelly complex. This is a very intuitive human form of interpretation and neuro-fuzzy models are gathering increasing acception. Unfortunatelly, neuro-fuzzy models can grow up to become of hard interpretation because of the complexity of the systems under modelling. In general, that growing occurs in function of redundant rules or rules thar cover a very little domain of the problem. This work presents an identification method for neuro-fuzzy models that not only allows models grow in function of the existent complexity but that beforehand they try to self-adapt to avoid the inclusion of new rules. This form of construction allowed to arrive to highly interpretative neuro-fuzzy models even of very complex systems. The use of this kind of technique in modelling the control of the pressurizer of a PWR nuclear power plant allowed verify its validity and how neuro-fuzzy models so built can be usefull in understanding the authomatic operation of a nuclear power palnt.

Índice

1 - Introdução 1

2 - Fundamentos teoricos 8

2 1 - Redes Neuronais 8

2 1 1 - Redes "backpropagation" 9

2 1 1 1 - Logica interna de processamento das unidades 11

2 1 1 2 - Treinamento 12

2 1 1 3 - Algoritmo backpropagation 13

2 1 2 - Redes de Base Radial 15

2 2 - Logica nebulosa 19

2 2 1 - Conjuntos nebulosos 23

2 2 2 - Funções de pertinência 25

2 2 3 - Proposições e regras nebulosas 26

2 2 4 - Procedimento de inferência nebulosa 27

2 2 5 - Tipos de inferências nebulosas 29

2 2 5 1 - Metodo de correlação de maximos e minimos 30

2 2 5 2 - Metodo de correlação de produtos 33

2 2 5 3 - Calculo do centro de gravidade ou centroide 34

2 3 - Definições 35

2 4 - Considerações 39

3 - Modelagem Neuro-Fuzzy 40

3 1- Hibridização de tecnicas inteligentes 40

3 2- Modelagem de sistemas 41

3 2 1 - Validade dos modelos 42

3 2 2 - Seleção da estrutura do modelo 45

3 2 2 1 - Modelos lineares 46

3 2 2 2 - Modelos não-lineares 46

3 2 3 - Funções de base 48

3 2 3 1- Funções de crista 48

3.2.3.2- Funções de base radial 49
3.2.3.3- Modelos polinomiais - As Splines 49
3.2.3.3.1 - Splines lineares 50
3.2.3.3.2 - Splines cúbicas 51
3.2.3.4 - Produto tensorial 52
3.2.3.4.1 - B-Splines 53
3.2.4 - Estimativa de parâmetros 54
3.2.4.1- O trabalho sobre os dados 55
3.2.5 - Os dados disponíveis 55
3.2.6 - Escolha de modelos 56
3.3 - Modelagem neuro-nebulosa 56
3.3.1- Limitações da modelagem neuro-nebulosa 59
3.3.1.1- A maldição da dimensionalidade 59
3.3.1.2- Quantidade de dados necessária 60
3.3.1.3- Qualidade dos dados 60
3.3.1.4- Conhecimento especialista 61
3.4- Bases de regras redundantes 61
3.5- Adaptação da estrutura do modelo 63
3.6- Modelo de Takagi-Sugeno 64
3.6.1- Modelo de Takagi-Sugeno de ordem zero 64
3.6.2- Modelo de Takagi-Sugeno de ordem um 65
3.7- Considerações 67

4 - Modelos adaptáveis 69
4.1 - Introdução 69
4.2 - A Flip-Net 70
4.2.1 - Uso de "backpropagation" para as funções de pertinência .. 72
4.3 - Anfis 75
4.3.1 - Redes adaptativas 75
4.3.2 - O Anfis 80
4.3.3 - Regras nebulosas simplificadas 82
4.3.4 - Exemplo 84
4.4 - Ajuste das funções de pertinência de saída 85

4.5 - Considerações 86

5 - Modelos expansíveis 88
5.1 - Um método de expansão de estrutura 88
5.1.1 - Uso do método em modelos SISO 89
5.1.1.1 - Exemplo 1 93
5.1.1.2 - Exemplo 2 96
5.1.2 - Uso do método em modelos MISO 98
5.2 - Um método aperfeiçoado 102
5.2.1 - DVS - Decomposição em Valores Singulares 103
5.2.2 - O problema reduzido 105
5.2.3 - Uso do método em problemas SISO 107
5.2.4 - Uso do método em modelos MISO 108
5.2.4.1 - Exemplo 3 108
5.2.4.2 - Exemplo 4 109
5.2.4.3 - Exemplo 5 110
5.3 - Considerações 112

6 - Um modelo expansível e auto-adaptável 114
6.1 - Objetivo 114
6.2 - A regra delta 115
6.3 - Aplicação a modelos SISO 118
6.4 - Aplicação a modelos MISO 121
6.5 - Algoritmo de aprendizagem 125
6.5.1 - Exemplo 1 126
6.5.2 - Algoritmo 130
6.6 - Exemplos 132
6.6.1 - Exemplo 2 132
6.6.2 - Exemplo 3 133
6.6.3 - Exemplo 4 134
6.6.4 - Exemplo 5 134
6.6.5 - Exemplo 6 137
6.6.6 - Exemplo 7 - Teste de Box & Jenkis 139

6.7 - Uso de Splines 140
6.7.1 - Exemplo 8 142
6.7.2 - Exemplo 9 143
6.7.3 - Exemplo 10 144
6.7.4 - Exemplo 11 145
6.8 - Considerações 146

7 -Resultados e conclusões 147
7.1 - Aplicação no controle do pressurizador de uma usina PWR 147
7.2 - Modelagem da oscilação espacial do fluxo de nêutrons 153
7.3 - O valor de λ 155
7.4 - Conclusões 157
7.5 - Futuras pesquisas 158

Referências bibliográficas 160

1

Introdução

A utilização das diversas técnicas de inteligência artificial em problemas de simulação, predição e controle, vem ocupando um lugar de destaque na literatura técnica. No caso da modelagem de sistemas físicos, pode-se considerar, mesmo em sistemas caóticos, que as variáveis envolvidas devem se manter dentro de determinados limites físicos admissíveis. Por exemplo, não podemos considerar a existência de temperaturas inferiores a -273,6 °C (zero absoluto). Este tipo de restrição possibilita o estudo desses sistemas dentro dos limites de interesse ou fisicamente exeqüíveis, não deixando margens a extrapolações indesejáveis. Quaisquer comportamentos não desejáveis das variáveis do sistema podem ser convenientemente manejados evitando-se, seja por definição, seja por aproximação, aqueles pontos onde o sistema sob estudo tenha um comportamento não definido (por exemplo, uma divisão por zero).

Este trabalho lida com sistemas físicos de naturezas linear e, principalmente, não linear desenvolvendo um algoritmo para modelagem e identificação desses sistemas em forma de modelos neuro-nebulosos (BOSSLEY, 1997, JANG, 1992). Tais sistemas físicos podem ser, a grosso modo, divididos em *sistemas estáticos*, onde a(s) saída(s) só dependa(m) das entradas do sistema num determinado momento e *sistemas dinâmicos* , onde a(s) saída(s) dependa(m) tanto das entradas atuais como das entradas e saídas passadas.

Na área nuclear são interessantes os objetivos de modelagem, diagnóstico e controle automático. Por exemplo, um sistema físico de interesse para controle automático nessa área é o funcionamento do pressurizador de uma usina nuclear PWR (ALVES, 1993). O controle automático, contínuo e suave desse equipamento pode assegurar tanto um funcionamento mais seguro quanto mais econômico de todo o sistema primário do reator da usina. Esta tese apresenta um modelo neuro-nebuloso para o controle desse equipamento. Quanto aos objetivos de diagnóstico, pode-se, por exemplo, observar o comportamento de algumas variáveis de estado dos circuitos

primário e secundário do reator para a análise de um evento de "trip" (ANONIMOUS, 1979).

A literatura costuma agrupar as técnicas de determinação de modelos matemáticos de sistemas reais em duas grandes categorias; a modelagem física e a identificação de sistemas (LJUNG, 1987). No caso de ser feita a modelagem física, uma formulação matemática dos fenômenos é utilizada como a fonte de conhecimento. No caso da identificação de sistemas, usa-se estudar o comportamento do mesmo sob condições controladas.

Quanto ao conhecimento que existe a priori, os sistemas sob modelagem podem ser agrupados em três diferentes níveis (LJUNG, 1987): o primeiro é a chamada *caixa branca* (white box), onde existe um completo ou muito grande conhecimento do sistema sob modelagem. Sistemas caixa branca permitem uma implementação de modelos quase imediata utilizando-se o conhecimento especialista disponível. Os modelos obtidos também podem ser eficientemente testados por um especialista com vistas à sua validação. Tais sistemas podem ser eficientemente implementados em *sistemas especialistas* (expert systems) construídos com linguagens de programação apropriadas. O conhecimento armazenado nesses sistemas é dito ser completamente "transparente".

No outro extremo, os sistemas *caixa preta* (black box) são aqueles nos quais existe muito pouco ou nenhum conhecimento a priori. Como não existe um conhecimento especialista disponível, os testes e a validação de tais sistemas são difíceis e incertos e, muitas vezes, resultados inesperados surgem na(s) saída(s) dos modelos. Tais sistemas podem, contudo, ser eficientemente implementados em redes neuronais. Nos próximos capítulos, aborda-se mais profundamente alguns conceitos fundamentais das redes neuronais. Pode-se adiantar que tais sistemas são, em geral, auto-adaptáveis, modificando seus parâmetros internos de forma a diminuir os erros na saída. Dessa forma, as redes neuronais realizam um tipo de aprendizagem baseado na observação das relações entre os padrões de saída e de entrada. O conhecimento daí auferido é, entretanto, de difícil observação e por isso tal conhecimento é referido como "opaco".

Outras técnicas baseadas em computação evolucionária (p. ex. os algoritmos genéticos) também são utilizadas, dependendo do sistema em estudo.

Na modelagem de um sistema intermediário, uma caixa cinza (grey box), procura-se incorporar algum conhecimento sobre o processo de identificação do modelo. Ainda que todos os resultados possíveis não possam ser previstos de antemão, a maioria das respostas dos modelos obtidos podem ser razoavelmente previstas e verificadas. A validação desses modelos pode ser feita estatisticamente, considerando-se as respostas dos modelos contra algum critério de verificação previamente definido. Nesse tipo de modelagem pode-se utilizar tanto as redes neuronais em treinamento supervisionado como sistemas especialistas de resposta estatística. Um outro método bastante interessante é o uso da lógica nebulosa.

Os sistemas físicos que possuem um comportamento capaz de ser descrito por meio de conceitos lingüísticos podem ser satisfatoriamente representados pelos conceitos da lógica nebulosa (ZADEH, 1965, 1996). Tais sistemas são, em determinado grau, adaptáveis e permitem que o conhecimento neles armazenado tenha um certo grau de transparência. A década de 1990 foi particularmente pródiga em sistemas que unem as capacidades das redes neuronais com os conceitos da lógica nebulosa, os sistemas neuro-nebulosos.

Os sistemas neuro-nebulosos conseguem unir as vantagens das redes neuronais e da lógica nebulosa de uma forma tão consistente que as desvantagens inerentes a cada técnica individualmente são praticamente canceladas. Os sistemas são, em geral, auto-adaptáveis e capazes de aperfeiçoar-se continuamente, armazenando o conhecimento adquirido em suas estruturas internas, como as redes neuronais. Tal conhecimento, contudo, não é de forma alguma *opaco*, podendo ser razoavelmente interpretado por um especialista, ainda que de forma superficial, no caso de existir uma profusão de regras redundantes. Nesse caso, tomou-se aqui a liberdade de dizer que o conhecimento armazenado é *translúcido*. Se, contudo, conseguirmos uma base de regras mínima e não redundante, são grandes as chances de uma alta interpretabilidade lingüística do modelo e, em tal caso, o conhecimento nele contido pode ser considerado *transparente*.

O objetivo desta tese foi o de obter uma forma algoritmizável de um tal modelo neuro-nebuloso de base de regras mínima e de alta interpretabilidade cujo conhecimento armazenado possa ser considerado como transparente. O modelo desenvolvido apresenta bons resultados em termos de precisão, generalização e clareza e foi aplicado em vários testes para validação. Também foi usado para modelar o controle automático de um pressurizador de usina PWR e a variação da concentração de xenônio em um reator nuclear como exemplos de sua aplicabilidade às necessidades de controle e diagnóstico na área nuclear.

Para tanto, alguns modelos anteriormente desenvolvidos foram usados tanto como embasamento como para comparações. A Flip-Net (MAEDA et al., 1993) é um modelo no qual uma base de regras definida por conhecimento especialista é refinada de forma que se possa verificar a validade e a adequação de determinadas regras previamente estabelecidas. Esse tipo de rede usa uma forma de retro-propagação dos dados de saída de forma a que a própria rede possa indicar onde estão os maiores ponto de erro. Será visto que a Flip-Net tem várias limitações. Por exemplo, sua base de regras (ou seja, sua estrutura) é fixa e não adaptável. Também, seus parâmetros não podem ser adaptados para novos padrões de entrada/saída. Na verdade, esta rede pode ser considerada como um sistema especialista que foi construido em forma de rede onde qualquer correção ou adaptação de sua estrutura ou de seus parâmetros internos deve ser feita em tempo de implementação.

O Anfis (JANG, 1992) é um modelo mais evoluído onde uma certa adaptação interna pode ser realizada pelo próprio sistema. Ele é considerado clássico no sentido em que demonstrou uma certa equivalência entre os sistemas nebulosos e as redes de base radial (HAYKIN, 1994). O Anfis permite uma razoável adaptação de seus parâmetros internos, mas não de sua estrutura, a qual também é definida em tempo de implementação. Ainda que a adaptação de parâmetros permita uma grande flexibilidade para o sistema, da forma como ela é implementada no Anfis essa adaptação leva a resultados indesejáveis em termos de transparência e de interpretação. Um outro porém é o fato de que a não capacidade de adaptação da estrutura limita consideravelmente tanto sua precisão como sua capacidade de generalização. Mais à frente, tanto o Anfis como as redes de base radial onde, o Anfis se baseia, são abordados.

Um outro trabalho que foi apresentado por NAKOULA (1997) propõe um método de adaptação tanto dos parâmetros como da estrutura de um modelo neuro-nebuloso. Nesse método, que será mais detalhado posteriormente, o erro sobre um conjunto de dados de treinamento é continuamente minimizado pela alteração da estrutura do modelo (introdução de novas regras na base de regras). A base de regras cresce até que uma determinada tolerância especificada seja atingida e cada nova regra é designada para o ponto responsável pelo maior erro geral em determinado momento. Os parâmetros do modelo são então alterados como conseqüência dessa adaptação da sua estrutura. O método possui vantagens e desvantagens. A capacidade de alteração da estrutura é uma evolução sobre os métodos anteriores. A adaptação dos parâmetros, contudo, ocorre apenas estaticamente, como conseqüência das mudanças na estrutura do modelo. Esse método tem a tendência de necessitar de um grande número de regras na base de conhecimento com um alto grau de redundância entre elas.

Uma maneira de diminuir a quantidade de regras na base de conhecimento foi apresentada recentemente (EVSUKOFF et al., 2000). Nesse método, a alteração da estrutura é feita como da maneira anterior mas a alteração dos parâmetros é feita de uma forma mais robusta, com base na resolução de um problema de mínimos quadrados. Foi verificado experimentalmente que tal adaptação dos parâmetros do sistema leva a uma base de regras bem menos redundante e mais precisa do que no método de Nakoula. Esse método também será desenvolvido num capítulo posterior. Apesar do progresso conseguido, apenas os parâmetros globais relacionados às regras são alterados. Parâmetros específicos de cada uma das regras como, por exemplo, sua abrangência e precisão não são afetados.

Nesta tese introduz-se uma maneira de superar esta última dificuldade. Baseado nos conceitos apresentados nos métodos anteriormente descritos, um novo método de adaptação tanto da estrutura como dos parâmetros internos de um modelo é apresentado. O método introduz novas regras ao modelo (alteração de estrutura) depois que uma adaptação tanto nos parâmetros globais como específicos de cada regra não conseguem mais diminuir sozinhos o erro sobre um conjunto de treinamento para uma tolerância especificada.

Adicionalmente, sobre o método de alteração de estrutura, esta tese faz um estudo a respeito de algumas formas de parametrização dos conjuntos de lógica nebulosa que podem ser utilizados em sistemas neuro-nebulosos o que, acredita-se poderá vir a ser de utilidade em estudos posteriores. Aqui também se desenvolvem considerações sobre uma relação empírica para o tamanho do passo de aprendizagem (especificamente, para a *constante de aprendizagem* λ) como uma função dos dados de treinamento envolvidos, de forma a superar o modo de tentativa-e-erro do uso de valores "pequenos", técnica tão comumente encontrada na literatura.

Esta tese foi escrita de maneira a apresentar seu embasamento teórico de forma encadeada para que o desenvolvimento do sistema proposto seja demonstrado de forma clara. Dessa forma, o capítulo 2 apresenta os pressupostos da lógica nebulosa e os tipos de inferências nebulosas mais utilizadas. Estabelece os fundamentos das redes neuronais artificiais, em especial, das redes de algoritmo backpropagation e das redes de base radial. Também é estabelecida nesse capítulo a terminologia utilizada ao longo desta tese.

O capítulo 3 desenvolve considerações sobre a modelagem neuro-fuzzy, a identificação de modelos de sistemas, a construção de bases de regras nebulosas e os modelos de Takagi-Sugeno (TAKAGI et al., 1985), um dos alicerces desta tese. Também se estabelecem as parametrizações de curvas que foram utilizadas.

O capítulo 4 aborda os modelos adaptáveis. Esses modelos podem alterar seus parâmetros internos (mas não sua estrutura) de maneira a adaptar sua saída aos requisitos do(s) sistema(s) sob estudo. São explanados os fundamentos e características tanto da Flip-Net como do Anfis. O Anfis é abordado em mais detalhes pois além de ser mais consistente possui características altamente desejáveis a esta tese.

O capítulo 5 refere-se aos modelos expansíveis. Esses modelos podem expandir suas estruturas mas são pobres em alterar seus parâmetros internos. Os sistemas de NAKOULA (1997) e de EVSUKOFF et al. (2000) são abordados com uma profundidade maior do que os apresentados no capítulo 3 pois eles servem de alicerce fundamental a este trabalho.

No capítulo 6 desenvolve-se o trabalho principal desta tese. É apresentada uma nova modelagem onde pode-se tanto ajustar como expandir o modelo de um sistema de acordo com as necessidades de precisão e de generalização. Faz-se uma abordagem teórica, apresenta-se um algoritmo de construção de modelos e também numerosos exemplos de teste.

O capítulo 7 apresenta o resultado obtido com a implementação de um sistema de controle neuro-nebuloso para um pressurizador de usina PWR. Discute-se a dificuldade de identificação e como o método introduzido no capítulo 6 pode ajudar um especialista a modelar esse controle. Também faz-se a modelagem da concentração de iodo em um reator como função da concentração de xenônio e da oscilação espacial do fluxo de nêutrons. Nas conclusões, tentou-se tanto quanto possível demonstrar a validade da modelagem proposta bem como estabelecer um caminho de pesquisa para trabalhos posteriores.

2

Fundamentos teóricos

2.1 - Redes Neuronais

Dentre as técnicas mais conhecidas de IA para aplicações em controle e reconhecimento de padrões estão as redes neuronais artificiais (RNA's) (JANG, 1992, HAYKIN, 1994, KOSKO, 1992). Elas trabalham segundo uma metáfora biológica com elementos que emulam a ação dos neurônios biológicos, tanto isolada como coletivamente. Como características, as RNA's possuem as habilidades de aprendizagem, generalização e abstração de dados. Ao lado de vários tipos como redes Hopfield, redes Kohonen e redes Counterpropagation, talvez as mais aplicadas sejam as redes tipo Backpropagation. A popularidade desse tipo de rede consiste em seu largo espectro de utilização, facilidade de implementação e razoável exatidão dos resultados.

Em todos os tipos de redes, a unidade fundamental de processamento é o chamado *neurônio artificial* como o mostrado na figura 2.1.

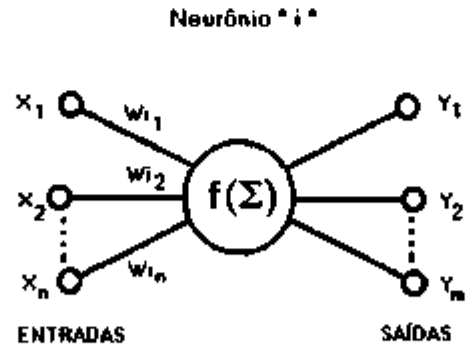


Figura 2.1 - Esquema simplificado de um neurônio artificial.

Também referenciado como *unidade* ou *nó*, ele possui uma ou mais entradas (1...n) que recebem sinais ativadores, uma lógica interna de processamento e também

uma ou mais saídas (1...m) que fornecem o resultado desse processamento a outras unidades da rede ou como saída final do modelo. Em geral, $n \neq m$.

Cada neurônio ou nó artificial é composto de um módulo de processamento e conexões de entrada e saída, chamadas "*sinapses*" ou, simplesmente, "*conexões*". A cada conexão ou sinapse, é associado um valor ou "*peso*" ($w_1, w_2, ..., w_n$) que será usado para valorizar ou desvalorizar a intensidade do sinal que passa por ele. Como entrada líquida, um nó típico receberá um valor dado pela expressão

$$\text{Entrada líquida} = \sum_j x_j \times w_j \qquad (2.1)$$

que é a soma ponderada das entradas.

Em qualquer tipo, a rede neuronal é construída em uma configuração "bruta", sendo, a seguir, "treinada" de forma a refinar seus parâmetros de funcionamento. A rede é considerada como treinada e pronta para uso quando suas respostas a determinados estímulos de entrada são considerados satisfatórios. Esse treinamento é feito normalmente nas conexões de entrada das unidades. Existe, contudo, uma outra maneira de treinar uma rede que é fazer a adaptação interna da lógica de processamento das unidades.

2.1.1 - Redes "backpropagation"

As redes backpropagation (HAYKIN, 1994) são redes multicamadas "feedforward", ou seja, nessas redes, o fluxo dos sinais de ativação segue sempre da entrada para a saída da rede. O refinamento de seus parâmetros de funcionamento é feito no sentido inverso com o uso de um algoritmo chamado "backpropagation", de onde a rede tira seu nome. São normalmente montadas em vários níveis e um exemplo está na figura 2.2 onde se vê uma rede BP de três níveis. O mais à esquerda é o nível de entrada onde os dados para processamento são apresentados à rede. O nível do meio é o chamado nível intermediário ou de processamento, onde os sinais recebidos do nível

anterior são processados de acordo com uma lógica interna mínima. Esse nível (ou níveis) intermediário é o maior responsável pelas características de resolução, generalização e abstração da rede e o bom desempenho da mesma está diretamente ligado ao número de unidades contidas nesse nível. O nível mais à direita é o chamado nível de saída onde os resultados da rede são entregues para uso. O nível de entrada, em geral, não realiza nenhum tipo de processamento tendo a função única de apresentar os dados de entrada às unidades do nível intermediário onde, aí sim, são manipulados matematicamente. Como já dito, às conexões entre os neurônios, chamamos, à semelhança com os neurônios biológicos, "sinapses" ou, simplesmente, "conexões".

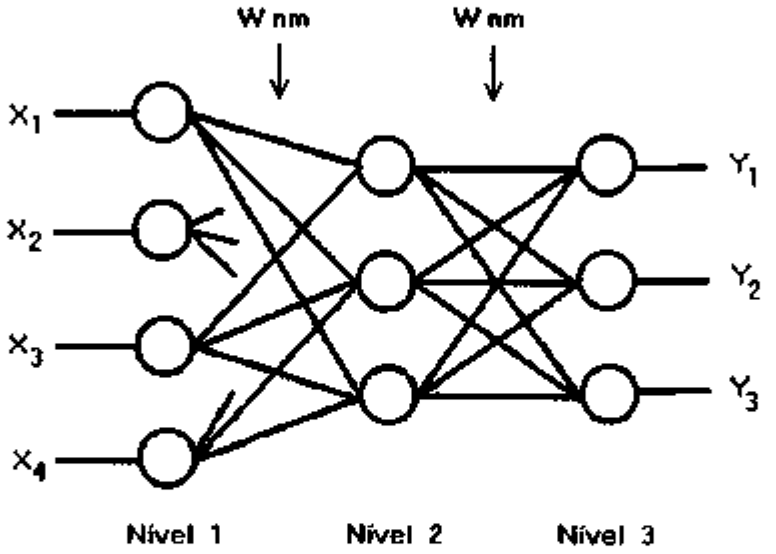


Figura 2.2 - Rede "backpropagation" de 3 níveis, 4 neurônios de entrada, 3 no nível intermediário e 3 de saída.

Ainda não existe uma maneira infalível de se determinar a quantidade de unidades necessárias ao nível intermediário. Contudo, pode-se arbitrar um número mínimo que é dependente da quantidade de unidades nos níveis de entrada e saída, da quantidade de padrões de entrada que a rede vai receber e também da quantidade de padrões de saída que a rede deve fornecer. Geralmente, chega-se a um bom compromisso por meio de tentativa e erro, analisando-se o comportamento da rede BP durante o treinamento. Também, sabe-se que quanto mais níveis intermediários uma

rede backpropagation tiver, maior será o seu poder de resolução. Por exemplo, uma rede com dois níveis intermediários consegue separar padrões de entrada que uma rede de nível único considera como "iguais".

2.1.1.1 - Lógica interna de processamento das unidades

A lógica interna de processamento de cada unidade geralmente é uma simples função matemática. Em redes BP usa-se uma função à qual chamamos de "*função de ativação*". Nessas redes, a função de ativação possui, em geral, a forma de um "S" ou "sigma" degenerado. Existem vários tipos de funções que propiciam essa forma mas as mais utilizadas em redes backpropagation são as funções "*Sigmoidal*" e *tangente hiperbólica* como mostram as figuras 2.3 (a) e (b). A função sigmoidal é definida como

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

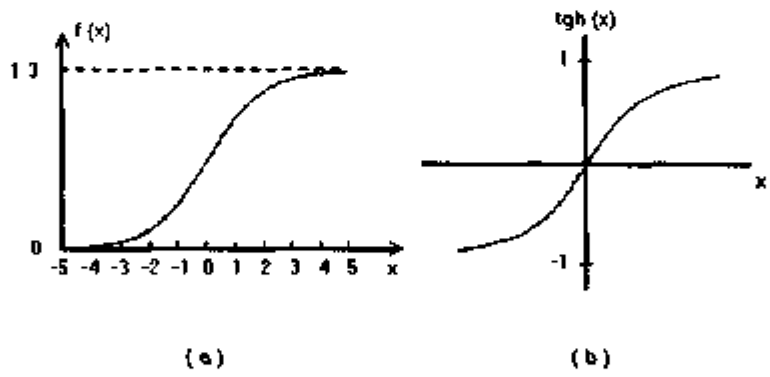


Figura 2.3 - Gráficos das funções (a)"Sigmoidal" e (b)"tgh (x)".

O uso de funções com esse formato, entre outras vantagens, propicia ajustes adequados às não-linearidades presentes nos dados de treinamento. Como pode ser visto nas figuras 2.1 e 2.2, uma unidade pode receber várias entradas simultaneamente. Poderia acontecer de o somatório dado por (2.1) atingir valores muito altos que, somados a outros valores possivelmente muito altos de outras unidades, viriam a

entregar aos nós do nível seguinte dados de difícil manejo, representando uma perda de eficiência ou dificuldades da rede BP em atingir uma configuração estável. A tal estado, chama-se "saturação" da unidade. As funções de ativação também tem a finalidade de manter as saídas das unidades em valores razoavelmente baixos (normalmente entre "0" e "1" ou entre "-1" e "1") de forma a facilitar os cálculos matemáticos envolvidos no processo de treinamento.

2.1.1.2 - Treinamento

Redes BP (e todos os demais tipos) devem sofrer, como dito anteriormente, um refinamento de sua configuração inicial. Esse refinamento é realizado, normalmente, nos pesos w_i das conexões das unidades de forma que, à aplicação de um determinado conjunto de entradas, a rede forneça um conjunto previamente determinado de saídas e é conhecido como "treinamento". Existem dois tipos básicos de treinamento: supervisionado e não-supervisionado.

No treinamento não-supervisionado, utiliza-se um conjunto de vetores de entrada e o algoritmo de treinamento modifica os pesos da rede de forma a que a rede forneça respostas consistentes de tal maneira que dois vetores parecidos venham a produzir a mesma saída. Este tipo de treinamento extrai as propriedades estatísticas do conjunto de vetores de entrada e os agrupa em classes. Ainda que existam aplicações interessantes onde o treinamento não-supervisionado seja atraente, normalmente, em uma rede BP, utiliza-se o treinamento supervisionado.

No treinamento supervisionado, utiliza-se um algoritmo no qual apresenta-se à rede BP um conjunto de pares entrada-saída, onde o grupo de entrada reflete a situação a ser identificada pela rede e o de saída é exatamente a saída desejada como resposta. Às primeiras apresentações, a rede normalmente apresenta como fim de processamento uma saída discordante daquela efetivamente desejada. Pelo uso do valor de saída desejado, a rede BP verifica o montante da discrepância entre seu sinal e o desejado (para mais ou para menos) e corrige seus parâmetros internos (normalmente, os pesos das conexões) de forma a minimizar o erro encontrado. Depois de um número incerto de iterações,

denominadas *passos de treinamento*, a rede BP normalmente encontra uma configuração estável na qual os erros entre os valores por ela calculados e os efetivamente desejados são, quando possível, arbitrariamente baixos.

A minimização dos erros da rede é feita propagando-se para trás, nível a nível, os erros encontrados. A esse algoritmo de treinamento chamamos de "*backpropagation algorithm*", de onde esse tipo de rede tira o seu nome, como já foi dito

2.1.1.3 - Algoritmo Backpropagation

O algoritmo *backpropagation* é um método de redução de erros que se baseia em informar aos níveis anteriores a quantidade de erro global atingida nos níveis posteriores. Ele pode ser resumidamente descrito como:

0- Inicializar os pesos da rede (em geral, com valores aleatórios e < 1).

ATÉ que o erro total seja do valor desejado FAÇA

1- Pegar o par entrada-saída de treinamento seguinte a partir de um conjunto previamente fornecido.
2- Aplicar os valores de entrada à entrada da rede.
3- Calcular a saída da rede.
4- Calcular o erro entre as saídas fornecidas pela rede e as saídas desejadas.
5- Ajustar os parâmetros (pesos) da rede de forma a minimizar os error.
6- Voltar para "1" desde que o erro total seja ainda não-aceitável.

Como já foi dito, normalmente os parâmetros a serem ajustados são os valores dos pesos das conexões entre as unidades. Maneiras alternativas de se treinar a rede seria fazendo um ajuste nas próprias funções de ativação das unidades ou nos parâmetros de condução das conexões (ligando-as ou desligando-as).

O erro "*E*" para uma unidade "j" do nível de saída é dado (HAYKIN, 1994) por

$$E_j = (D_j - S_j) f' (x_j) \qquad (2.3)$$

onde:

D_j = Valor desejado de saída da unidade "j".

S_j = Valor real de saída da unidade "j".

$f'(x)$ = Derivada da função de ativação

x_j = Soma ponderada das entradas da unidade "j".

A quantidade (D_j - S_j) reflete a grandeza do erro. A derivada f' faz um escalonamento desse erro de forma a forçar o algoritmo a fazer uma forte correção quando a soma Sj estiver próxima à região de rápido crescimento da curva sigmoidal (ou tgh). Uma maneira pela qual podemos definir o erro total na saída é:

$$E_{total} = \sqrt{\frac{\sum_p \sum_j (Djp - Sjp)^2}{np \times ns}} \qquad (2.4)$$

onde:

n_p = número de padrões de entrada.

n_s = número de neurônios do nível de saída.

Dependendo de qual tipo de ajuste será feito nos parâmetros da rede, se nos pesos das conexões ou nas formas das funções de ativação, esse erro será tratado pela rede de forma determinada. Contudo, um ponto em comum em qualquer caso é o uso de um método de "descida da colina", onde se determina o gradiente do erro e caminha-se na direção contrária. No caso de ajuste dos pesos, o erro calculado é usado em um algoritmo de cálculo chamado "regra delta" (HAYKIN, 1994) que é usado no passo 5 (ajuste dos pesos) do algoritmo backpropagation. Por problemas inerentes ao método, pode acontecer que, em determinadas situações a rede não consiga encontrar, caso exista, o ponto de erro mínimo desejado para os valores apresentados. A rede deixa de encontrar o ponto de "erro mínimo global" por se encontrar presa em um de vários

possíveis pontos de "erro mínimo local" que são, via de regra, maiores que o ponto de mínimo local. Quando tal caso ocorre, a rede não consegue atingir a precisão desejada porque torna-se incapaz de escapar desse ponto, visto que, em qualquer direção dentro desses pontos, o gradiente tende a aumentar. Tal fato chama-se "paralisia". Nos últimos anos, vários métodos foram criados para se contornar esse problema porém, nenhum deles conta com uma total robustez.

2.1.2 - Redes de Base Radial

As redes de função de base radial, mais conhecidas como redes de base radial, RBR's (ou Radial Basis Function Networks - RBFN's) (HAYKIN, 1994) podem ser usadas tanto para a aproximação de funções como para reconhecimento de padrões (JANG, 1992). Uma função é dita de base radial quando, de um ponto de vista local, seu valor é não negativo e alto em um determinado ponto C de seu domínio e decai até zero conforme a norma $| x - C |$ aproxima-se do infinito. Como exemplo mais imediato, temos a função gaussiana (fig. 2.4).

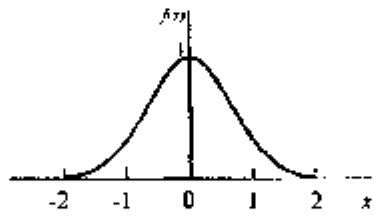


Figura 2.4 - Gráfico da função Gaussiana.

Em problemas de ajuste de curvas em um espaço multi-dimensional, o aprendizado é equivalente a achar uma superfície em tal espaço que proporcione o melhor ajuste aos dados de treinamento sendo esse melhor ajuste medido sob algum senso estatístico. Por outro lado, a capacidade de generalização consiste no uso adequado de tal superfície de forma a interpolar adequadamente os dados.

A motivação para o uso de funções de base radial em redes neuronais consiste no fato de que as unidades receptoras do nível oculto fornecem uma "base de funções" para

os vetores de entrada conforme eles são apresentados. Como se vê na figura 2.5, a forma básica de uma RBR é também de três níveis distintos.

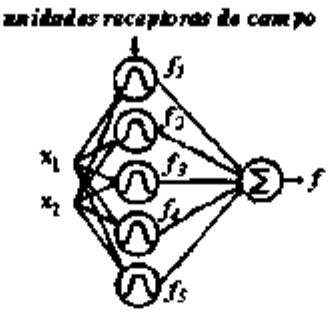


Figura 2.5 - Esquema de uma rede de base radial.

O nível de entrada consiste das unidades receptoras. O segundo nível é um nível oculto de dimensionalidade suficientemente alta para acomodar os espaços de entrada. O último nível é o de saída que fornece as respostas da rede. Normalmente, as transformações do espaço de entradas no nível oculto é não linear ao passo que as ativações do nível oculto sofrem uma transformação linear ao nível de saída.

Existem benefícios em se mapear um espaço de entrada não linearmente separável de um problema de classificação em um novo espaço de maior número de dimensões (JANG, 1992). Basicamente, é usado um mapeamento não linear para transformar um problema de classificação linearmente não separável em um novo problema linearmente separável. Assim, em problemas de interpolação de dados, podemos usar um mapeamento não linear para transformar um complicado problema de interpolação não linear em um problema mais fácil, que permita uma interpolação linear.

Consideremos, sem perda de generalidade, uma rede como a da figura 2.6 com um nível de entrada, um nível oculto e um nível de saída. Imagine-se esse nível de saída como constituído de uma única unidade. Essa rede é desenhada para realizar um mapeamento não linear do espaço de entrada para o nível oculto e, daí, um mapeamento linear do nível oculto para o nível de saída. Se designarmos para a entrada um espaço *p-dimensional*, e sendo a saída unidimensional, teremos que a rede realiza o mapeamento

(JANG, 1992)

$$s: \mathbf{R}^p \rightarrow \mathbf{R}^1$$

entre os espaços de entrada e saída. O mapeamento s é uma hipersuperfície $\Gamma \subset \mathbf{R}^{p+1}$ que, a princípio, é desconhecida e deve ser identificada a partir dos dados de treinamento.

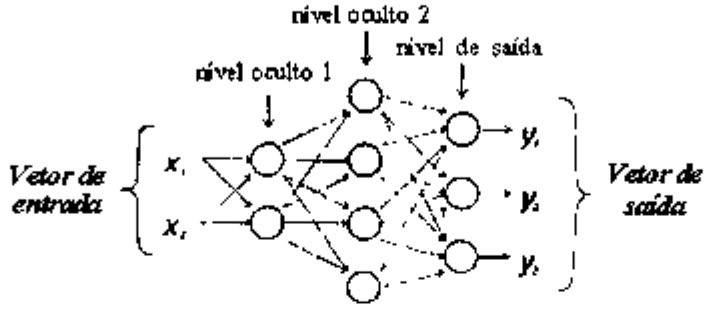


Figura 2.6 - Uma rede simples.

Essa identificação pode ser realizada em duas fases: 1) a fase de treinamento, que consiste na busca de um formato para a hipersuperfície Γ, busca essa baseada nos pares entrada / saída de treinamento e 2) a fase de generalização, na qual os ajustes da curva são feitos pela interpolação dos intervalos entre os pontos de treinamento e de tal forma que esse procedimento de interpolação permita uma máxima aproximação com a verdadeira hipersuperfície contínua desejada. Tal mapeamento nos leva ao *problema da interpolação* (JANG, 1992) que, em senso estrito, pode ser colocado da seguinte forma:

• *Dado um conjunto de N diferentes pontos* $\{ x_i \in R^p \mid i = 1,2, \ldots, N \}$ *e um conjunto correspondente de números reais* $\{ d_i \in R^1 \mid i = 1,2, \ldots, N \}$, *devemos achar a função* $F: R^N \rightarrow R^1$ *que satisfaça à condição de interpolação*

$$F(x_i) = d_i, \; i = 1,2, \ldots N \tag{2.5}$$

•

A técnica das funções de base radial consiste em escolher uma função F que possua a forma

$$F(\mathbf{x}) = \sum_{i=1}^{N} w_i \varphi(\|\mathbf{x} - \mathbf{x}_i\|) \tag{2.6}$$

onde $\{ \varphi (\|\mathbf{x} - \mathbf{x}_i\|) \mid i = 1, 2, \ldots, N \}$ é um conjunto de N funções arbitrárias conhecidas como funções de base radial com os pontos $\mathbf{x}_i \in R^p$ sendo tomados como os centros das funções e "$\|\cdot\|$" denota uma norma qualquer, normalmente euclidiana. Se inserirmos as condições de interpolação das equações (2.5) e (2.6) , obtemos um conjunto de equações lineares simultâneas para os coeficientes (pesos) desconhecidos da expansão $\{ w_i \}$.

$$\begin{bmatrix} \varphi_{11} & \varphi_{12} & \cdots & \varphi_{1N} \\ \varphi_{21} & \varphi_{22} & \cdots & \varphi_{2N} \\ \vdots & & & \vdots \\ \varphi_{N1} & \varphi_{N2} & \cdots & \varphi_{NN} \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_N \end{bmatrix} = \begin{bmatrix} d_1 \\ d_2 \\ \vdots \\ d_N \end{bmatrix} \tag{2.7}$$

onde a matriz dos coeficientes φ_{ji} , denotada Φ, tem dimensões N x N e é denominada de matriz de interpolação onde os φ_{ji} são tais que

$$\varphi_{ji} = \varphi (\|\mathbf{x}_j - \mathbf{x}_i\|) ; \quad i, j = 1, 2, \ldots, N \tag{2.8}$$

e

$$\mathbf{d} = [d_1, d_2, \ldots, d_N]^T \tag{2.9}$$

$$\mathbf{w} = [w_1, w_2, \ldots, w_N]^T \tag{2.10}$$

são vetores N x 1 que representam o *vetor de respostas desejado* e o *vetor de pesos lineares*, respectivamente. Podemos reescrever a equação matricial (2.7) na forma mais compacta

$$\Phi \mathbf{w} = \mathbf{X} \tag{2.11}$$

Seja definida a seguinte propriedade:

♦ *Sejam* $x_1, x_2, \ldots, x_N$ *pontos distintos em* R^p *Então a matriz de interpolação* Φ *cujo j-ésimo elemento é* $\varphi_{ji} = \varphi\left(\|x_j - x_i\|\right)$ *é positiva definida*

♦

Existe uma classe de funções de base radial que aderem a essa propriedade, por exemplo, as funções logísticas, como a tgh, as funções gaussianas e as funções tensoriais.

Segundo a literatura (HAYKIN, 1994), a expressão de φ(.) não é crucial em relação ao desempenho das RBR's. Esta tese mostra que, em se obedecendo determinados vínculos desejáveis, a escolha de φ( ) tem um peso considerável sobre a convergência de sistemas neuro-nebulosos baseados em funções de base radial. De qualquer forma, o valor de φ( ) computado pela i-ésima unidade é máximo quando o valor de entrada x é igual ao do centro da função de base da unidade. A saída da rede pode ser computada como a soma ponderada dos valores das funções de cada unidade, ou seja

$$f(\mathbf{x}) = \sum_{i=1}^{H} f_i w_i = \sum_{i=1}^{H} f_i R_i(\mathbf{x}) \qquad (2.12)$$

onde f_i é o peso de ativação da i-ésima unidade e $R_i(\mathbf{x})$ i = 1,2, ... , H é a resposta da i-ésima unidade receptora.

2.2 - Lógica nebulosa

A maioria das ferramentas computacionais para modelagem, relacionamentos e controle possuem um caráter predominantemente determinístico, preciso e agudo, ou seja, são do tipo "sim-ou-não", como a lógica booleana. Essa precisão assume que existe a certeza de que os parâmetros de um modelo representam exatamente seja nossa percepção do fenômeno modelado sejam as características intrínsecas do sistema sob

modelagem. Tal certeza indica que se assume que as estruturas e parâmetros do sistema são precisamente conhecidos ou seja, que não existem dúvidas a respeito dos seus valores e do seu comportamento. Essa precisão pode ser simbolizada em termos de uma lógica booleana como, por exemplo, a indicada na tabela 2.1.

Tabela 2.1 - Lógica "sim-ou-não" de um controle simples de temperatura.

SE a temperatura é superior a 35 graus	ENTÃO ligue o ventilador em alto
SE a temperatura é superior a 30 graus E inferior a 35 graus	ENTÃO ligue o ventilador em médio
SE a temperatura é superior a 25 graus E inferior a 30 graus	ENTÃO ligue o ventilador em baixo
SE a temperatura é inferior a 25 graus	ENTÃO desligue o ventilador

Contudo, duas limitações estão sempre presentes, qualquer que seja o sistema sendo modelado. A primeira é que as situações reais normalmente não são determinísticas ou perfeitamente definidas não podendo, portanto, serem descritas de forma precisa. A segunda reside no fato de que a descrição completa e total de um sistema real, não importa sua simplicidade, normalmente irá requerer uma especificação muito mais detalhada do que possa ser, mesmo depois de vários refinamentos, obtida por um ser humano que a esteja observando.

Existem algumas características que regem os sistemas que vemos no mundo real. Por exemplo, a maioria das situações são normalmente incertas em determinados enfoques. Devido à falta de informações, o estado futuro de um sistema pode não ser completamente conhecido. Um exemplo pode ser invocado do controle de temperatura de um ambiente, por exemplo, como exemplificado na tabela 2.2. Não só os conceitos de quente e frio são altamente pessoais como o estado futuro do sistema é altamente incerto devido às variações de condições internas e externas sobre as quais pouco ou nenhum controle se pode exercer. Este tipo de incerteza de caráter estocástico tem sido bem tratado há bastante tempo pelas estatística e teoria das probabilidades.

Tabela 2.2 - Regras incertas associadas à temperatura ambiente.

SE	a temperatura for muito quente	ENTÃO	ligue o ventilador em alto
SE	a temperatura for quente	ENTÃO	ligue o ventilador em médio
SE	a temperatura for acima do normal	ENTÃO	ligue o ventilador em baixo
SE	a temperatura for fria	ENTÃO	desligue o ventilador

Também, o pensamento humano inclui elementos ilógicos tais como a intuição e a inspiração. Juntamente com o conhecimento lógico adquirido, esses elementos desempenham um papel importante em atividades onde o julgamento, a avaliação e decisões baseadas em pouco conhecimento são necessárias. Desnecessário é dizer que é virtualmente impossível expressar tais elementos ilógicos em termos de uma lógica puramente booleana. Precisamos, para manejar adequadamente esse tipo de problema, não de uma lógica tipo "sim-ou-não" mas de uma lógica tipo "mais-ou-menos".

Dessa necessidade, foi desenvolvida uma teoria de sistemas incertos que foi chamada de "lógica nebulosa" (*fuzzy logic*) (ZADEH, 1996, ZIMMERMANN, 1994). Desde meados da década de 60, quando foi formalmente estabelecida, essa teoria vem sendo desenvolvida e refinada e hoje encontra aplicações em vários campos tais como a ciência da computação, a inteligência artificial, sistemas de controle, reconhecimento de padrões, robótica, reconhecimento de voz, etc. Além desses campos mais técnicos, ela também vem sendo aplicada com sucesso em campos voltados às ciências humanas e sociais como tomada de decisão, modelagem de sentimentos e linguagem humanos, finanças, modelagem psicológica de comportamento, etc.

Os conjuntos nebulosos (*fuzzy sets*) são um conceito matemático que foi proposto pelo professor L. A. Zadeh em 1965 (ZADEH, 1965). Sua grande característica é a habilidade de expressar a quantidade de ambigüidades existentes tanto no pensamento humano como em sistemas reais sujeitos a condições incontroláveis. Por exemplo, as incertezas contidas nas expressões "alta temperatura", "número pequeno" e "pessoa idosa" indicam que essas quantificações não são exatas, mas "nebulosas".

Como exemplo, analisemos o conceito de "meia-vida de um elemento radioativo". Podemos ver no gráfico da figura 2.7 que à medida que o valor da meia-vida dos elementos radioativos aumenta, gradativamente podemos associa-la ao termo lingüístico "meia-vida longa".

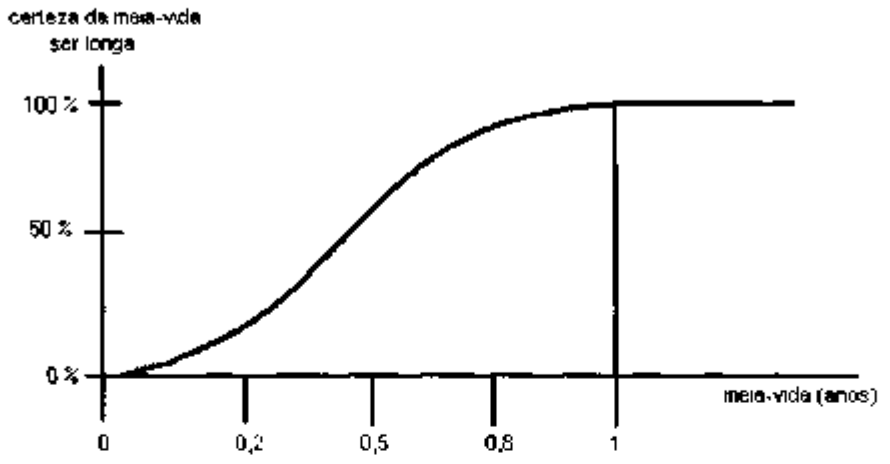


Figura 2.7 - Exemplo de curva da certeza da longevidade de um elemento radioativo.

Contudo, como as meias-vidas podem variar de intervalos de segundos a séculos, o que, realmente, entendemos por "meia-vida longa" ? Podemos lidar com essa ambigüidade atribuindo "graus de certeza" para as meias-vidas dos diversos elementos radioativos. Assumindo que uma meia-vida seja considerada longa a partir de um ano, um elemento cuja meia-vida seja de 2 anos pode ser considerado certamente como a tendo como "longa" e lhe atribuímos 100% de certeza de ele possuir essa característica. Por outro lado, o próprio conceito de "longa" é arbitrário.

Podemos expandir a idéia para traçarmos um gráfico, como o apresentado na figura 2.8, no qual redefinimos e apresentamos três tipos de meias-vidas: curta, média e longa. Nessa figura, um elemento cuja meia-vida seja de 50 meses pertence, com grau de certeza 90% ou 0,9 ao grupo dos elementos de meia-vida média. Contudo, com 10% (ou 0,1) de certeza, já se encontra no conjunto dos de meia-vida longa. Quanto maior a meia-vida, menos certeza se terá que ele pertença ao grupo de "média" e a certeza que ele pertence ao grupo de "longa" aumenta. A teoria dos conjuntos nebulosos é então, como seu nome implica, basicamente uma teoria a respeito do que não se tem muita certeza, mas que ainda assim se tem alguma. Assim, ela provê um meio de se lidar com

problemas nos quais as fontes de imprecisão são a ausência de rígidos critérios de pertinência a uma determinada classe. A quantificação dessas ambigüidades são o problema central da lógica nebulosa.

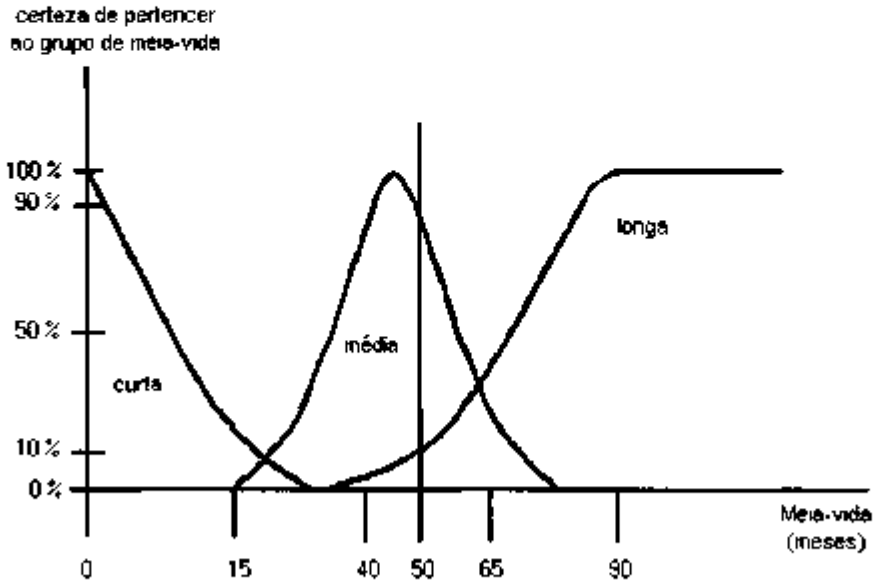


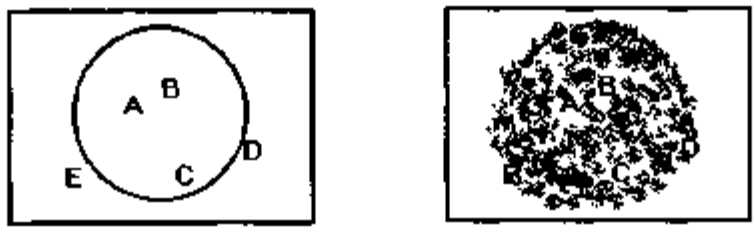
Figura 2.8 - Funções de pertinência das meias-vidas de elementos radioativos.

2.2.1 - Conjuntos nebulosos

Um conjunto clássico normalmente é definido como uma coleção de elementos ou objetos $x \in$ a um conjunto **X** o qual pode ou não ser finito, contável ou incontável. Cada elemento pode pertencer ou não a um conjunto **A**, $\mathbf{A} \subset \mathbf{X}$. Caso pertença, a declaração "x pertence a **A**" é verdadeira. Caso não, a declaração será falsa. Tal conjunto clássico pode ser definido de várias maneiras, por exemplo, analiticamente, postulando uma condição de pertinência tal como $\mathbf{A} = \{ x \mid x \leq 5 \}$. Qualquer número que não satisfaça à condição dada definitivamente não pertence ao conjunto **A**. Essa é uma condição "sim-ou-não".

Um conjunto nebuloso (fuzzy set) $\tilde{\mathbf{A}}$ é definido de tal forma que os elementos ou objetos que o compõem pertencem a $\tilde{\mathbf{A}}$ em *algum grau*. Esse grau pode variar continuamente de 0 a 1 sendo que "1" significa que o elemento pertence

indubitavelmente ao conjunto $\tilde{A}$ e "0" que o elemento está, sem nenhuma dúvida, fora do campo de ação de $\tilde{A}$. Os valores contínuos do intervalo aberto (0 , 1) indicam que o elemento pertence um pouco a $\tilde{A}$, tão mais fortemente quanto o valor seja mais próximo de "1" e tão mais fracamente em caso contrário. A figura 2.9 indica como podemos visualizar os conjuntos clássico e nebuloso.



Conjunto Clássico Conjunto Nebuloso

Figura 2.9 - Esquemas gráficos de conjuntos clássicos e nebulosos.

No conjunto clássico existe uma fronteira bem estabelecida entre os pontos interiores e exteriores do conjunto. Na figura, verificamos que os elementos "A", "B" e "C" pertencem ao conjunto clássico enquanto os elementos "D" e "E" não. No conjunto nebuloso essa fronteira não é muito clara e determinados elementos vão estar parte dentro e parte fora do conjunto. Na figura 2.9, os elementos "C", "D" e "E" estão parte dentro e parte fora do conjunto nebuloso.

Em forma analítica, podemos definir o conjunto nebuloso $\tilde{A}$ usando um par ordenado $(x, \mu\tilde{A}(x))$ de tal forma que, se existe uma coleção de objetos x que pertençam a um conjunto **X** então

$$\tilde{A} = \{ (x, \mu\tilde{A}(x)) \mid x \in \mathbf{X} \} \qquad (2.13)$$

onde $\mu\tilde{A}(x)$ é a chamada *função de pertinência* ou grau de pertinência do elemento x em $\tilde{A}$.

Voltando ao exemplo anterior da meia-vida de um elemento radioativo, vemos que na figura 2.8, definiu-se uma função de pertinência para "meia-vida média".

Poderíamos defini-la rigorosamente, por exemplo, como:

$$\mu_{média}(\text{meia-vida}) = max\left(0,\ 1 - \frac{|meia\text{-}vida - 40|}{25}\right) \qquad (2.14)$$

Nesse caso, a função de pertinência tem a forma da figura 2.10a. Assim, podemos ver, para várias meias-vidas, como um elemento se encaixa dentro dessa faixa. Um elemento com meia-vida de 50 meses pertence à faixa "meia-vida média" com grau de certeza 0,6. As meias-vidas de 15 e 65 meses dão um grau de certeza 0. O grau de certeza máximo ("1") é obtido para a meia-vida de 40 meses. Elementos com meias-vidas menores que 15 meses ou maiores que 65 meses não pertencem (obtêm grau "0") a essa faixa.

2.2.2 - Funções de pertinência

Como verificamos das figuras 2.8 e 2.10, as funções de pertinência que correspondem à nebulização de uma variável podem ser definidas de diversas maneiras. As formas mais comuns são em "triângulo" (figura 2.10a), "trapezoidal" (2.10b) e "sino" (2.10c). Dessas três, a mais largamente utilizada é o triângulo por fornecer um bom compromisso entre precisão, facilidades de codificação e cálculos computacionais. As equações que regem as figuras 2.10a, 2.10b e 2.10c são, respectivamente, (2.14), (2.15) e (2.16).

$$\mu_{média}(\text{meia-vida}) = \begin{cases} -0{,}05 \times meia\text{-}vida + 3{,}5 & ;\ 50 \le meia\text{-}vida \le 70 \\ 0 & ;\ meia\text{-}vida < 15 \ \ ou \ \ meia\text{-}vida > 70 \\ 1 & ;\ 35 \le meia\text{-}vida \le 50 \\ 0{,}05 \times meia\text{-}vida - 0{,}75 & ;\ 15 \le meia\text{-}vida \le 35 \end{cases} \qquad (2.15)$$

$$\mu_{média}(\text{meia-vida}) = \frac{1}{1 + (meia\text{-}vida - 40)^2} \qquad (2.16)$$

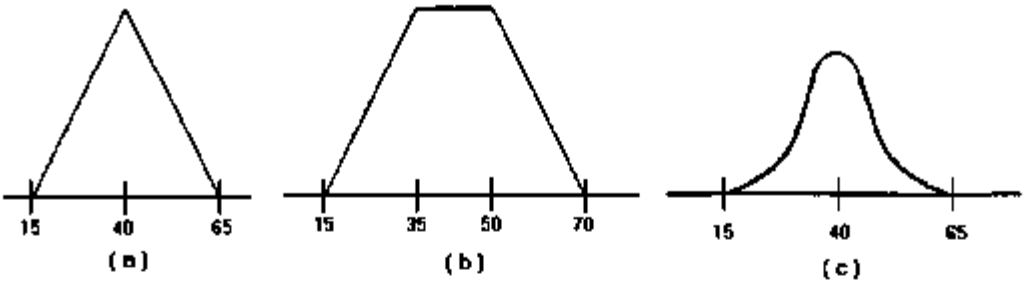


Figura 2.10 - Três tipos de função de pertinência para conjuntos nebulosos.

2.2.3 - Proposições e regras nebulosas

O estabelecimento analítico de uma proposição nebulosa não é significativamente diferente de uma não nebulosa. Ambas são analiticamente definidas como

$$x \text{ É } P \tag{2.17}$$

Sua interpretação, contudo, é diferente. Em forma não nebulosa, "x é P" sempre, nunca sendo diferente. Já em forma nebulosa, (2.17) deve ser interpretado como "x é próximo de P em algum grau". Essa pequena diferença é fundamental. Por exemplo, a afirmação "x é vermelho" implica em que a única cor refletida pelo objeto é a vermelha enquanto que a proposição "x é avermelhado" nos informa que o objeto reflete majoritariamente a cor vermelha mas também reflete outras cores com menor intensidade.

Proposições podem ser combinadas de forma a formarem regras. Em lógica de predicados, comumente usada em sistemas computacionais convencionais, são comuns regras do seguinte tipo:

SE x = 1 ENTÃO faça $y = 0$;
SE x = 2 ENTÃO faça $y = 5$;

Se a variável x pode assumir apenas alguns valores então essas regras são poucas e podem ser programadas sem muitos problemas. Contudo, se x puder assumir uma

gama muito grande de valores, a quantidade de regras necessárias cresce a um ponto onde fica virtualmente impossível seu manejo. Pior ainda ocorre quando existem várias variáveis envolvidas na parte "SE" (antecedente) de uma regra como em

$$\underline{\text{SE}}\ x = 2\ \underline{\text{E}}\ z = 3\ \underline{\text{E}}\ w = 0\ \underline{\text{ENTÃO}}\ \text{faça}\ y = 5; \qquad (2.18)$$

Imaginando-se que cada uma das variáveis x, y e z podem assumir valores discretos de 0 a 5, teremos a necessidade de implementar 6x6x6 = 216 regras ! Esse problema pode ser evitado se, ao invés de usarmos lógica de predicados, usarmos lógica nebulosa para construir as regras. A regra nebulosa equivalente à regra (2.18) acima é exatamente a mesma:

$$\underline{\text{SE}}\ x = 2\ \underline{\text{E}}\ z = 3\ \underline{\text{E}}\ w = 0\ \underline{\text{ENTÃO}}\ \text{faça}\ y = 5; \qquad (2.19)$$

A diferença está na sua interpretação. Agora, "x é igual a 2" em *algum grau* e, da mesma forma, as variáveis *z* e *w* (que juntamente com *x* formam a parte *antecedente ou das premissas* das regras) também se assemelham a 3 e 0 de alguma forma, ainda que essa semelhança seja nenhuma. Da mesma forma, a variável *y* (o conseqüente) também não mais assume o valor imutável de 5. Na verdade, seu valor passa a ser semelhante a 5 também em algum grau.

2.2.4 - Procedimentos de inferência nebulosa

Suponhamos que temos um sistema físico qualquer no qual existam duas variáveis de entrada x_1 e x_2 as quais devemos avaliar em conjunto para estabelecermos o valor que deve ser atribuído a uma variável *y* de saída. Ambas as variáveis de entrada podem assumir valores contínuos de 0 a 10 e *y* pode assumir valores de 0 a 20, por exemplo. Se usarmos lógica de predicados para estabelecermos as regras que relacionam as entradas à saída, ainda que discretizemos as variáveis de entrada em 10 passos, abandonando os valores contínuos entre os passos, ainda teríamos que lidar com 11x11 = 121 regras. Se utilizarmos lógica nebulosa esse número pode ser drasticamente reduzido para 4 regras.

Como já visto, um conjunto nebuloso é composto de um ou mais pares ordenados na forma $(x, \mu\tilde{A}(x))$ onde $\mu\tilde{A}(x)$ significa o grau de pertinência de x ao conjunto $\tilde{A}$, podendo variar dentro do intervalo contínuo e fechado [0 , 1]. Se fizermos a "fuzificação" das variáveis x_1 , x_2 e y segundo funções de pertinência como desenhadas na figura 2.11, podemos combinar as variáveis de entrada e saída de forma que as regras

SE	x_1 = pequeno	E	x_2 = pequeno	ENTÃO	y = pequeno;		
SE	x_1 = pequeno	E	x_2 = médio	ENTÃO	y = pequeno-médio;		
SE	x_1 = pequeno	E	x_2 = grande	ENTÃO	y = médio;		
SE	x_1 = médio	E	x_2 = pequeno	ENTÃO	y = pequeno-médio;		
SE	x_1 = médio	E	x_2 = médio	ENTÃO	y = médio;		
SE	x_1 = médio	E	x_2 = grande	ENTÃO	y = médio-grande;		
SE	x_1 = grande	E	x_2 = pequeno	ENTÃO	y = médio;		
SE	x_1 = grande	E	x_2 = pequeno	ENTÃO	y = médio;		
SE	x_1 = grande	E	x_2 = pequeno	ENTÃO	y = médio;		(2.20)

nos forneçam um valor contínuo da variável de saída y de acordo com o necessário para a solução do problema. Nesta tese usam-se indistintamente os termos *nebulização* e *fuzificação*.

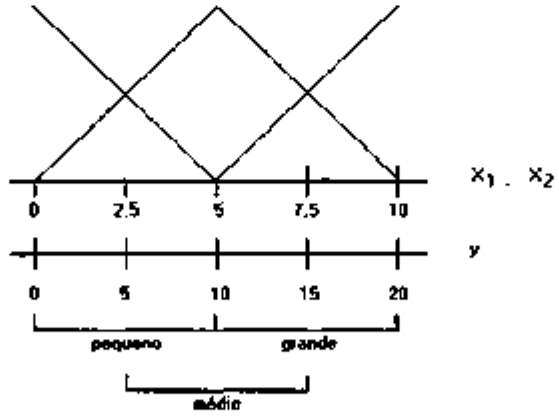


Figura 2.11 - Exemplo da nebulização das variáveis x_1, x_2 e y de um sistema físico.

2.2.5 - Tipos de inferências nebulosas

De uma forma geral, podemos agrupar as inferências realizadas por modelos neuro-nebulosos em três principais casos, esquematizados na figura 2.12 e que são as inferências nebulosas de tipos 1, 2 e 3 (JANG, 1992).

a) Inferência nebulosa do tipo 1:

A saída calculada pelo modelo é a soma ponderada da saída de cada regra induzida pela força de ativação de cada regra e pelos formatos das funções de pertinência delineadas nas saídas associadas sendo a associação das regras relacionadas por um dos operadores mínimo (min) ou produto (.) .

b) Inferência nebulosa do tipo 2:

Neste caso, a saída fornecida pelo modelo é calculada pela aplicação do operador máximo (max) às saídas associadas às regras ativadas sendo que, neste caso, existem diferentes modos de calcular-se o valor desfuzificado na saída como o cálculo do centróide de áreas, a bissetriz da área e a média dos máximos, entre outros.

c) Inferência nebulosa do tipo 3:

A saída de cada regra é uma combinação linear das variáveis de entrada mais um termo constante e a saída final é a média ponderada das saídas associadas às regras ativadas.

A preferência pela utilização de um determinado tipo é normalmente feita em função da aplicação. Por exemplo, para aplicações de controle, o cálculo da saída deve ser feito da forma mais rápida possível e então a inferência de tipo 2 é uma péssima candidata à escolha. Esse método é usado na Flip-Net.

Existem duas formas mais utilizadas de se fazer a combinação das variáveis de entrada em sistemas de inferência nebulosa tipo 2. Elas são o *método de correlação de*

mínimos e máximos e o *método de correlação de produtos* (SUGENO et al., 1992). A correlação de máximos e mínimos produz um corte nos conjuntos nebulosos de saída. A correlação de produtos, ao contrário, promove um escalonamento desses mesmos conjuntos.

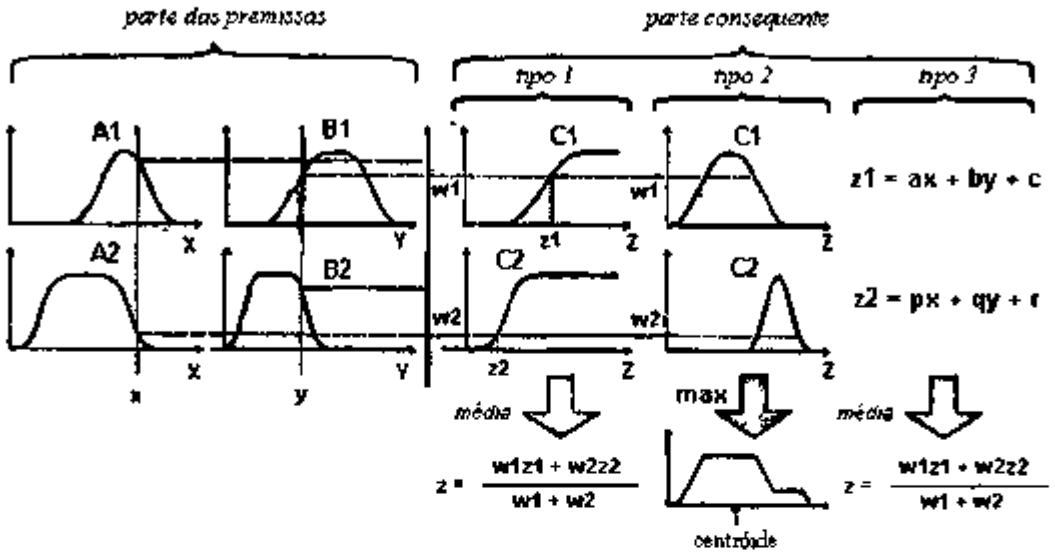


Figura 2.12 - Tipos de inferências nebulosas.

2.2.5.1 - Método de correlação de máximos e mínimos

No método de máximos e mínimos, a composição de relações nebulosas usa operações onde se determinam os máximos e mínimos dos relacionamentos entre as proposições nebulosas. O operador mínimo corresponde à operação lógica "AND" e o operador máximo corresponde à "OR". Matematicamente, consideremos que R seja uma relação nebulosa em X x Y e que S seja uma relação nebulosa em Y x Z. A composição de R e S, R ° S é uma relação nebulosa em X x Z da seguinte forma:

$$R \circ S \leftrightarrow \mu_{(R \circ S)}(x,z) = \vee(y) \{ \mu_{(R)}(x,y) \wedge \mu_{(S)}(y,z) \} \qquad (2.21)$$

Por exemplo, sejam $X = \{x_1,x_2\}$, $Y = \{y_1,y_2,y_3\}$ e $Z = \{z_1,z_2\}$ com x_1, . Podemos expressar as relações R e S pelas matrizes

$$R = \begin{array}{c} \\ x_1 \\ x_2 \end{array} \begin{array}{c} \begin{array}{ccc} y_1 & y_2 & y_3 \end{array} \\ \begin{bmatrix} 0,4 & 0,6 & 0 \\ 0,9 & 1 & 0,1 \end{bmatrix} \end{array} \qquad S = \begin{array}{c} \\ y_1 \\ y_2 \\ y_3 \end{array} \begin{array}{c} \begin{array}{cc} z_1 & z_2 \end{array} \\ \begin{bmatrix} 0,5 & 0,8 \\ 0,1 & 1 \\ 0 & 0,6 \end{bmatrix} \end{array}$$

A composição de R e S ocorre de forma que corresponde ao produto ordinário das matrizes, com a diferença de que o operador $\vee$ (máximo) substitui o de soma e o operador $\wedge$ (mínimo) substitui o de produto. Assim,

$$R \circ S = \begin{bmatrix} 0,4 & 0,6 & 0 \\ 0,9 & 1 & 0,1 \end{bmatrix} \circ \begin{bmatrix} 0,5 & 0,8 \\ 0,1 & 1 \\ 0 & 0,6 \end{bmatrix} =$$

$$= \begin{bmatrix} (0,4 \wedge 0,5) \vee (0,6 \wedge 0,1) \vee (0 \wedge 0); & (0,4 \wedge 0,8) \vee (0,6 \wedge 1) \vee (0 \wedge 0,6) \\ (0,9 \wedge 0,5) \vee (1 \wedge 0,1) \vee (0,1 \wedge 0); & (0,9 \wedge 0,8) \vee (1 \wedge 1) \vee (0,1 \wedge 0,6) \end{bmatrix} =$$

$$= \begin{array}{c} \\ x_1 \\ x_2 \end{array} \begin{array}{c} \begin{array}{cc} z_1 & z_2 \end{array} \\ \begin{bmatrix} 0,4 & 0,6 \\ 0,5 & 1 \end{bmatrix} \end{array}$$

Suponhamos agora que tenhamos o seguinte banco de regras nebulosas:

regra 1: SE x_1 é pequeno E x_2 é médio ENTÃO y é médio
regra 2: SE x_1 é grande E x_2 é pequeno ENTÃO y é grande (2.22)

Dadas duas entradas x_1 e x_2 "fuzificamo-las" na forma da figura 2.13 onde também nebulizamos a saída y sobre um espaço z. Os graus das regras são dados pelos mínimos das partes "SE".

$$W_1 = \min (\text{pequeno}(x_1) , \text{medio}(x_2))$$
$$W_2 = \min (\text{grande}(x_1) , \text{pequeno}(x_2)) \qquad (2.23)$$

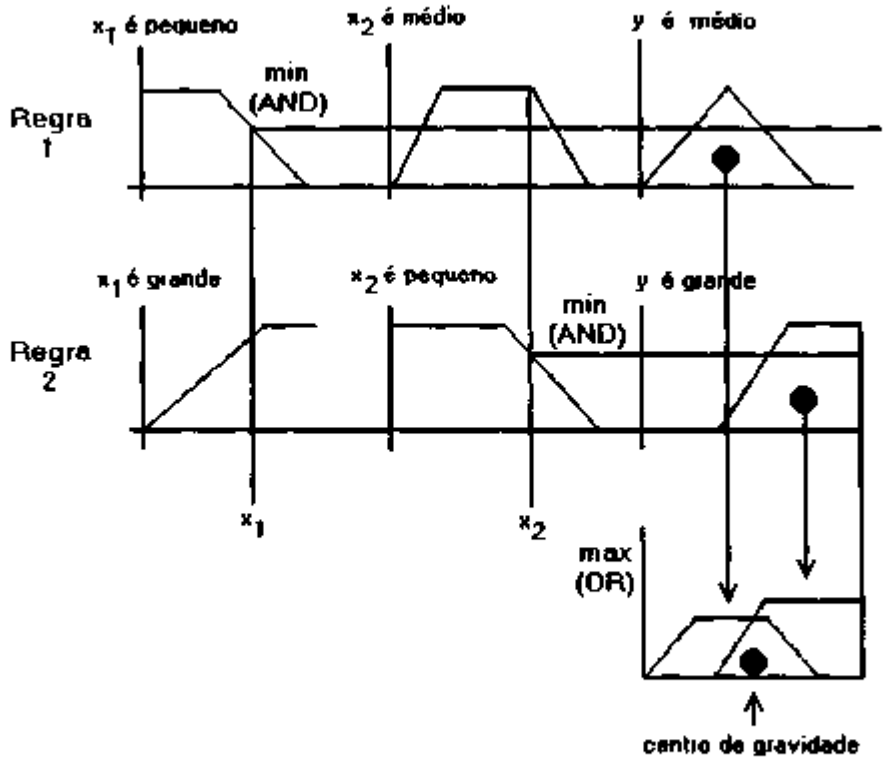


Figura 2.13 - Nebulização das regras (2.22) com a correlação de máximo e mínimos.

Os "cortes" na variável fuzificada de saída y nos fornecem duas figuras que são combinadas pelo operador *max* que, como já dito, corresponde ao "OR" lógico. Assim, a função de pertinência de saída B(z) é dada por:

$$B(z) = \max (W_1 \wedge \text{médio } y(z) , W_2 \wedge \text{grande } y(z)) \qquad (2.24)$$

Onde "^" significa que as funções de pertinência distribuídas sobre o espaço E(z) estão sendo "cortadas" nos pontos determinados por W_1 e W_2. Essa combinação nos dá uma nova figura na qual a saída y requerida é determinada ("desfuzificada") pelo cálculo do seu centro de gravidade.

No caso do problema proposto, suponhamos que temos como entrada x_1 = grande e x_2 = pequeno. A figura 2.13 mostra a situação. A saída desejada, *y*, fornecida pelo método de correlação de mínimos e máximos é obtida calculando-se o centro de gravidade da combinação das funções de pertinência de saída cortadas a partir do cálculo dos mínimos.

2.2.5.2 - Método de correlação de produtos

O esquema de correlação de máximos e mínimos tem um problema fundamental relacionado ao corte que o operador de mínimo produz nos conjuntos nebulosos de saída. O espaço de saída B(z) combinado a partir de E(z) tende a ser, na sua maior parte, plano. Isso faz com que se perca grande parte da informação contida em E(z) e que é relativa às formas das funções de pertinência de saída. Considere a figura 2.13. Nas regras 1 e 2 temos funções de pertinência de saída completamente diferentes as quais, sofrendo o corte mostrado, nos fornecem, ambas, uma figura trapezoidal. Isso causa a perda de grande parte da informação. Assim, mesmo que se procure fazer um refinamento no espaço E(z), esse refinamento pode ser perdido simplesmente pelo corte indiscriminado realizado pela correlação de máximos e mínimos.

No esquema de correlação de produtos procura-se preservar a informação distribuída no espaço E(z) por meio do escalonamento das funções de pertinência nele definidas. Depois de se obter W_1 e W_2 em (2.23) da mesma maneira que na correlação de máximos e mínimos, a função de saída B(z) é agora dada por

$$B(z) = W_1 \,.\, \text{médio}\, y(z) \cup W_2 \,.\, \text{grande}\, y(z) \qquad (2.25)$$

Onde "." significa produto, o que escalona as funções de pertinência definidas sobre B(z) e $\cup$ significa a união das funções assim obtidas. Agora, o exemplo nos fornece o espaço de saída como o mostrado na figura 2.14.

A saída *y* desejada é, da mesma forma, obtida calculando-se o valor do centro de gravidade da nova figura formada. É fácil observar que agora as formas das funções de pertinência definidas sobre E(z) são mantidas não importa qual seja a escala adotada. Em um exemplo muito simples como este pode parecer que não importa muito qual tipo de correlação seja adotada. Isso é verdade. Contudo, em sistemas mais complexos, a correlação de produtos é capaz de fornecer uma saída muito mais precisa quando se vai calcular o valor do centro de gravidade.

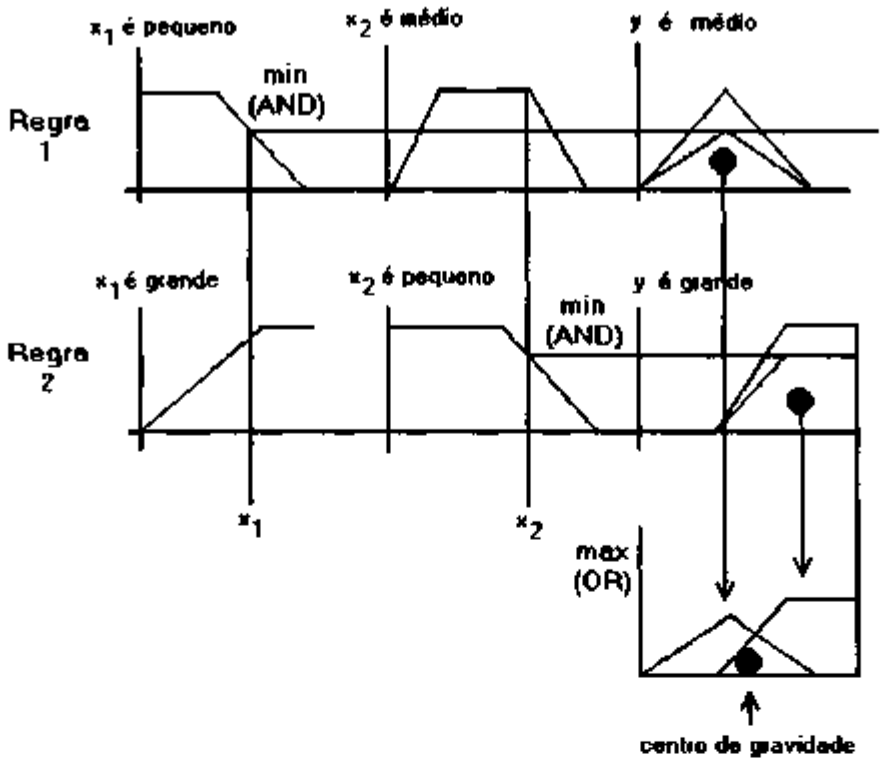


Figura 2.14 - Nebulização das regras (2.9) com a correlação de produtos.

2.2.5.3 - Cálculo do centro de gravidade ou centróide

Qualquer que seja o método utilizado para se fazer a inferência nebulosa de tipo 2, seja de correlação de máximos e mínimos, seja de correlação de produtos, a saída desejada é obtida pelo cálculo do centróide da combinação das figuras obtidas por corte ou escalonamento. O valor de saída é dado por

$$\hat{y} = \frac{\int B(z).zdz}{\int B(z)dz} \tag{2.26}$$

como é normal, da geometria, para o cálculo do centro de gravidade de uma figura. Em cálculos computacionais, por métodos numéricos, esse valor é dado, aproximadamente, por

$$\hat{y} = \frac{\sum_{k=1}^{K} B(z_k).\, z_k}{\sum_{k=1}^{K} B(z_k)} \qquad (2.27)$$

onde os z_k ($k = 1, \ldots, K$) representam os valores discretizados de z.

As funções de pertinência de saída podem ter, por exemplo, as formas da figura 2.15. É claro que a escolha adequada da maneira como se vai fuzificar as variáveis de entrada e saída é determinante para que o sistema se comporte de maneira adequada. Por exemplo, a fuzificação de x_1, x_2 e y das figuras 2.13 e 2.14 pode não ser a mais adequada para um correto funcionamento do sistema. Quanto mais complexo for o sistema sob consideração mais difícil é a construção das funções de pertinência. Alguns métodos (JANG, 1992, MAEDA, 1993, NAKOULA, 1997) tem sido propostos e efetivamente levam a um refinamento das formas das funções de pertinência de maneira a melhorar o funcionamento de sistemas de lógica nebulosa.

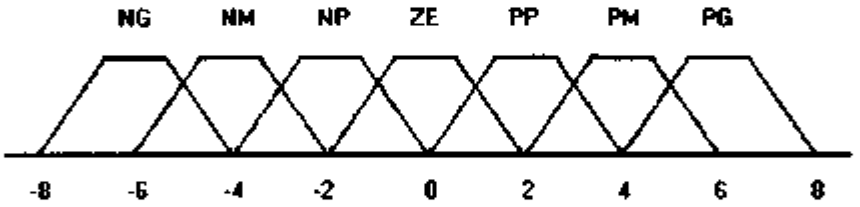


Figura 2.15 - Exemplo de funções de pertinência de saída definidas sobre um espaço E(z) de -8 a 8.

2.3 - Definições

Esta tese segue a terminologia usada em trabalhos anteriores (EVSUKOFF, 1998, EVSUKOFF et al., 2000). Assim, um conjunto nebuloso é aqui definido por

$$\mathbb{A} = \{(x, A(x)), x \in X\} \qquad (2.28)$$

onde o conjunto $X \subset R$ é o que denominamos *domínio de referência* (EVSUKOFF et

al., 2000) e $A(x) = \mu \tilde{A}(x)$ é a função que caracteriza o grau de pertinência de um qualquer elemento $x \in X$ ao conjunto nebuloso $\tilde{A}$. Os valores contidos no conjunto X podem ser descritos por um conjunto de termos lingüísticos. Por exemplo, numa determinada aplicação, o conjunto descritor $\mathbf{A} = \{ A_1 \ ... \ A_n \}$ pode ser considerado como sendo sua interpretação lingüística. Neste caso, cada um dos termos $A_i \in \mathbf{A}$ representa um conceito que engloba vários valores de X que tem seu significado vinculado à definição dada ao conjunto nebuloso $\tilde{A}_i$. Dado um elemento $x_0 \in X$, o valor resultante da função de pertinência $Ai(x_0)$ informa o quanto x_0 satisfaz ao conceito que é representado pelo conjunto nebuloso $\tilde{A}_i$ como se vê na figura 2.16 para os pontos m e n.

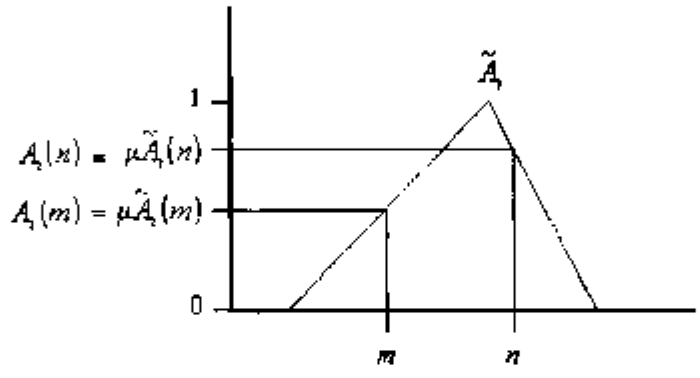


Figura 2.16 - Pertinências de um conjunto nebuloso para dois pontos distintos.

Uma *partição nebulosa* do domínio de referência X é definida (EVSUKOFF, 1998) como sendo a coleção de conjuntos nebulosos $\tilde{A}_i$ associados a cada termo $A_i \in \mathbf{A}$ como pode ser visto, no caso de uma partição triangular, na figura 2.17.

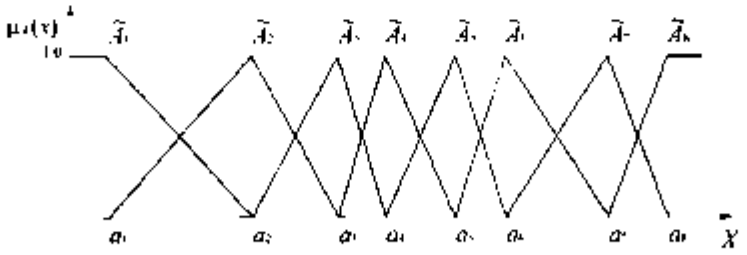


Figura 2.17 - Partições triangulares.

A descrição lingüística ou, simplesmente, *descrição*, de um dado elemento $x_0 \in X$, obtida por essa partição, é um conjunto nebuloso X'_0 que é definido sobre o conjunto descritor **A** como sendo (EVSUKOFF, 1998)

$$X'_0 = \left\{ \left(A_i , X_0(A_i) \right), A_i \in \mathbf{A} \right\} \tag{2.29}$$

Verificamos na figura 2.18 (EVSUKOFF, 1998) que a descrição do elemento $x_0 \in X$ está relacionada com o significado lingüístico de cada termo $A_i \in \mathbf{A}$ de tal forma que

$$X_0(A_i) = A_i(x_0) \tag{2.30}$$

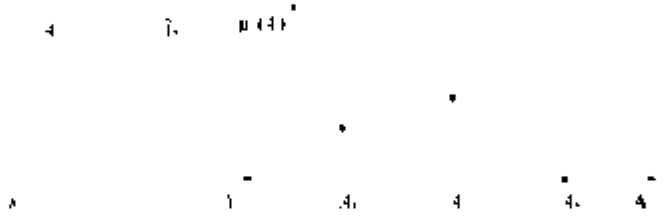


Figura 2.18 - Cálculo das descrições.

Kosko (1992) apresenta uma representação geométrica da descrição X'_0 em forma de um vetor em um hipercubo unitário $I^n = [\ 0\ ,\ 1\ ]^n$ de tal forma que (EVSUKOFF et al., 2000)

$$X'_0 = \left[X_0(A_1) \ldots X_0(A_n) \right] = \left[A_1(x_0) \ldots A_n(x_0) \right] \tag{2.31}$$

onde n é o número de termos da partição. E dessa forma o mapeamento entrada-saída é então visto como um mapeamento entre cubos unitários. A descrição X'_0 é assim calculada pela função vetorial que é definida pela partição A: $X \rightarrow I^n$ de forma que (EVSUKOFF et al., 2000)

$$X'_0 = \mathbf{A}(x_0) = \left[A_1(x_0) \ldots A_n(x_0) \right] \tag{2.32}$$

As partições nebulosas desta tese satisfazem às condições

$$\text{a)}\ \ \forall i,\ \exists x_0 \in X\ \ /\ \ A_i(x_0) = 1 \tag{2.33}$$

$$\text{b)}\ \ \sum_i A_i(x) = 1,\ \forall x \in X \tag{2.34}$$

Essas partições nebulosas podem ser formadas por diversos tipos de funções de pertinência como, por exemplo, as partições triangulares expostas na figura 2.17. Qualquer que seja a função de particionamento utilizada, ver-se-á que ela pode ser inteiramente parametrizada pelos pontos modais do espaço particionado.

Reportando-nos outra vez à figura 2.17, os parâmetros $\mathbf{a} = [\ a_1\ ...\ a_n\ ]^T$ definem os chamados *protótipos* (EVSUKOFF et al., 2000) dos conjuntos nebulosos da partição A(x). A identificação da estrutura do modelo nebuloso visa a determinar corretamente o número e a localização dos protótipos necessários que definam as partições sobre o referencial de cada variável.

Um dos objetivos da modelagem é o de construir modelos nebulosos que possam ser, posteriormente, interpretados por especialistas no domínio do problema. Para tanto, consideramos que um modelo lingüístico que relacione as variáveis x (entrada) e y (saída) pode ser descrito por uma base de regras que relaciona os termos dos conjuntos descritivos $A_i \in \mathbf{A}$ e $Bj \in \mathbf{B}$ em regras do tipo $A_i \rightarrow Bj$ e que são escritas como

$$\underline{\text{SE}}\ \ x \text{ é } A_i\ \ \underline{\text{ENTÃO}}\ \ \ y \text{ é } Bj \tag{2.35}$$

Essa base de regras pode ser representada pela relação nebulosa Φ definida pelo produto cartesiano $\mathbf{A} \times \mathbf{B}$ (EVSUKOFF, 1998). Cada regra dessa base pode ser ponderada pelo valor de pertinência $\Phi(A_i, Bj)$. Esse valor representa o quanto o termo A_i está associado ao termo Bj no modelo que é descrito por tal base.

Dessa forma, um valor $\Phi(A_i, Bj) = V > 0$ significa que a regra que associa A_i a Bj está presente em tal base e que possui um peso V. A relação nebulosa Φ pode ser representada por uma matriz contida no espaço $I^{n \times m}$ com n e m sendo, respectivamente,

as dimensões das descrições definidas pelas partições de **A**(x) e **B**(y). A tabela 2.3 apresenta um exemplo ilustrativo, onde a base de regras definida pelas expressões à esquerda é representada pela matriz à direita.

Tabela 2.3 - Correspondência base de regras x matriz de regras.

$A_1 \rightarrow B_2$	$\tilde{\Phi} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}$
$A_2 \rightarrow B_1$	
$A_3 \rightarrow B_2$	

2.4 - Considerações

Foram abordados os conceitos fundamentais da teoria da lógica nebulosa e das redes neuronais. Apesar deste trabalho não utilizar as redes backpropagation, o seu uso difundido e facilidade de entendimento levaram à sua escolha, juntamente com as redes de base radial, para a apresentação das redes neuronais. Adicionalmente, o algoritmo backpropagation faz uso do método de cálculo do gradiente, que será usado nesta tese.

Apresentou-se os conceitos da lógica nebulosa bem como os procedimentos de inferência nebulosa mais utilizados.

A terminologia usada nesta tese foi apresentada de forma a permitir um bom entendimento do trabalho desenvolvido e as possíveis parametrizações dos conjuntos nebulosos foram apresentados de forma sucinta.

3

Modelagem Neuro-Fuzzy

3.1 - Hibridização de técnicas inteligentes

Não importa a técnica de inteligência artificial por nós escolhida para lidar com um determinado problema, nenhuma solução nasce "pronta". A partir de um protótipo ou construção inicial, fazem-se necessários melhoramentos e refinamentos na técnica escolhida de acordo com o problema em foco. Alguns pesquisadores denominaram de "inteligência artificial sinergética" (MAEDA et al., 1995) à hibridização de técnicas de inteligência artificial de forma a aproveitar os melhores recursos de cada uma ou para que uma ou mais técnicas permitam um melhor desempenho de outra(s).

Desde o início da década de 1990, quando os pesquisadores começaram a utilizar procedimentos computacionais de emulação de processos humanos de decisão, as técnicas mais utilizadas então e ainda hoje são os sistemas especialistas, a lógica nebulosa, as redes neuronais e as técnicas evolucionárias, como os algoritmos genéticos. As classificações usuais em arquiteturas híbridas são divididas primordialmente em quatro (MAEDA et al., 1995): combinação, fusão, integração e associação. Essas classificações são derivadas dos relacionamentos topológicos entre os elementos inteligentes.

Na arquitetura de *combinação* as técnicas inteligentes em uso imitam certas atividades cerebrais de forma complementar. Como resultado, podemos expandir a capacidade de resolução de problemas pela combinação de técnicas inteligentes. Uma arquitetura híbrida típica é a combinação seqüencial de redes neuronais e sistemas baseados em regras ou lógica nebulosa.

Ainda que a combinação seja a arquitetura híbrida básica, em alguns casos, a *integração* de outros elementos inteligentes ajuda a determinar o comportamento total

do sistema. Por exemplo, uma integração pode selecionar os elementos mais apropriados para atingir um objetivo específico e combinar as respostas dos elementos selecionados.

Uma característica das tecnologias de computação neurais e evolucionárias é a forte capacidade de otimização matemática. Essa capacidade às vezes trabalha em aquisição de conhecimento e às vezes em adaptação do aprendizado. Quando outras técnicas incorporam essa característica elas são capazes de incrementar sua capacidade de aprendizado. Do ponto de vista topológico de arquiteturas híbridas, este tipo de arquitetura é uma *fusão* de técnicas inteligentes. Sistemas inteligentes flexíveis requerem uma arquitetura distribuída onde cada elemento trabalhe autonomamente e cooperativamente. A arquitetura de *associação* permite aos pesquisadores criar uma grande variedade de agentes inteligentes para diferentes situações. Infelizmente, este tipo de arquitetura ainda não atingiu um nível de maturidade que permita seu uso em aplicações de larga escala.

3.2 - Modelagem de sistemas

Este trabalho lida com a modelagem de sistemas como uma combinação de técnicas de IA utilizando uma abordagem de lógica nebulosa sobre uma topologia de redes neuronais. Procuramos identificar modelos neuro-nebulosos onde a interpretação lingüística seja um fato e que possuam altas capacidades de acuidade e de generalização. Os tipos de sistemas que nos interessam são de natureza muito geral, podendo ser estáticos, onde a saída só depende das entradas no momento presente ou dinâmicos, onde a saída depende tanto das entradas atuais como das entradas e saídas passadas. No desenvolvimento de modelos de sistemas, podemos contar tanto com um especialista e seu conhecimento "a priori" como conseguir uma aprendizagem pela observação e experimentação. Também, dados empíricos podem ser usados para adaptar e validar o comportamento dos modelos.

Como visto no capítulo 1, existem 3 níveis de síntese de modelos (LJUNG, 1987): a) *White Box* que exige um completo conhecimento do sistema. b) *Grey Box*

onde é necessário algum conhecimento do sistema e c) *Black Box*, onde não dispomos de qualquer conhecimento sobre o sistema a ser modelado. De acordo com essa separação, podemos estabelecer as modelagens mais apropriadas para cada tipo de sistema.

Os modelos de sistemas caixa branca exigem bastante conhecimento "a priori" dos sistemas sob investigação. A abordagem adequada pode ser a de sistemas especialistas (expert systems) ou, quando desejável, dos modelos nebulosos. Em ambos os casos temos um conjunto de regras altamente interpretáveis que levam à uma base de conhecimento totalmente transparente. Esses modelos, entretanto, são pouco flexíveis.

Já os modelos de sistemas caixa preta privilegiam a aprendizagem pela observação. A abordagem mais eficiente é a usada pelas redes neuronais, que mantém um tipo de conhecimento armazenado mas não facilmente interpretável. Esse conhecimento é praticamente opaco mas os modelos obtidos são altamente adaptáveis.

Com os modelos de sistemas caixa cinza (Grey Box), a metodologia mais eficiente parece ser a dos modelos neuro-nebulosos, que são representados por descrição lingüística e, em certo grau, adaptáveis. Denominamos aqui de *translúcido* ao conhecimento armazenado em sistemas caixa cinza.

Conforme aumentam as dimensões do modelo, o conjunto de dados suficientemente representativo do sistema cresce exponencialmente. Isso leva ao problema conhecido como "*maldição da dimensionalidade*" (BOSSLEY, 1997). Com as modelagens hoje conhecidas, mais do que 4 entradas praticamente inviabilizam o uso da abordagem neuro-nebulosa para a identificação de modelos de sistemas devido à explosão combinatória das variáveis de entrada.

3.2.1 - Validade dos modelos

Nossa preocupação primordial é sempre com a saída do modelo, sua validade e sua acuidade. Uma boa abordagem na construção de modelos de sistemas dinâmicos é

sua modelagem a partir da combinação de conhecimentos "a priori" com dados empíricos. Ou seja, dados:

$$\mathbf{u}^t = [\, \mathbf{u}(1), \mathbf{u}(2), \ldots, \mathbf{u}(t-1) \,] \tag{3.1a}$$

$$y^t = [\, y(1), y(2), \ldots, y(t-1) \,] \tag{3.1b}$$

devemos achar:

$$\hat{y}(\mathbf{u}^t, y^t) \text{ tal que } \hat{y} = y + e(t) \text{ com } e(t) \approx 0.$$

onde : $\hat{y}$ é a saída calculada pelo modelo

y é a saída fornecida pelo sistema (ou desejada do modelo)

$e(t)$ é o erro do modelo.

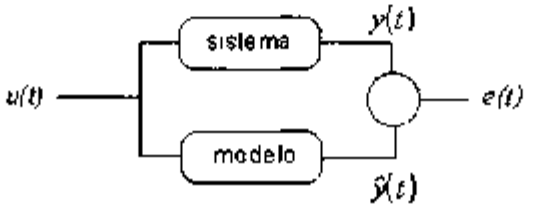


Figura 3.1 - Comparação entre um sistema e seu modelo.

O esquema é visualizado na figura 3.1 (BOSSLEY, 1997). Os objetivos a serem alcançados são:

1) Previsibilidade: Para qualquer entrada admissível (ou infinitas entradas) $\mathbf{u}^t$, devemos ter:

$$J_\infty = \sum_{\forall} E\, [\, (y(t) - \hat{y}(\mathbf{u}^t, y^t))^2 \,] \tag{3.2}$$

pequeno, onde J_∞ é o erro total do sistema para qualquer (infinito) conjunto de dados e E é o erro de um dado em particular. Contudo, como apenas um conjunto N finito de dados está disponível, então nos contentamos em ter

$$J_N = \frac{1}{N}\sum_{i=1}^{N}\left[(y(i) - \hat{y}(\mathbf{u}', y'))^2\right] \qquad (3.3)$$

pequeno. Identificamos a expressão acima como sendo o "erro médio quadrático" (EMQ ou MSE). Mesmo que $J_N \approx 0$, J_∞ pode ser grande. Por isso, a habilidade do modelo em predizer dados desconhecidos é fundamental.

2) Generalização:

Modelos complexos possuem alta acuidade mas baixa capacidade de generalização que é o quanto um modelo pode abstrair as características comuns a um grupo de dados. Adicionalmente, também possuem alta capacidade de adaptação que é o quanto o modelo pode modificar-se para atender a novos requisitos e, claro, alta complexidade. Já modelos simples possuem baixas acuidade e capacidade de adaptação mas tem como vantagens uma baixa complexidade e uma alta capacidade de generalização. Caimos num dilema de Generalização x Acuidade o que leva ao *princípio da parcimônia* (BOSSLEY, 1997):

"O modelo aceitável mais simples produz os melhores resultados".

3) Interpretabilidade:

O conhecimento adquirido e acumulado no modelo deve ser interpretável, ou seja, o modelo deve ser o mais "*transparente*" possível.

4) Eficiência:

Devemos procurar modelos pequenos o suficiente para que neles possam ser implementados algoritmos eficientes.

5) Adaptabilidade:

Novos dados devem modificar e aperfeiçoar o modelo.

3.2.2 - Seleção da estrutura do modelo

Normalmente, o modelo é restrito a uma família de funções definidas sobre um conjunto finito de entradas e saídas passadas, família essa que pode ser parametrizada por um vetor w tal que (BOSSLEY, 1997)

$$\hat{y}(\mathbf{u}^1, y^1) = \hat{y}(\mathbf{x}(t), \mathbf{w}) \tag{3.4}$$

sendo $\mathbf{x}(t)$ o regressor (as entradas), dado por

$$\mathbf{x}(t) = [\ \mathbf{u}^T(t-1), \ldots, \mathbf{u}^T(t-n_u), y(t-1), \ldots, y(t-n_y)\ ]^T \tag{3.5}$$

Na modelagem de uma caixa-preta, a estrutura do modelo é fixada e procura-se w de forma a maximizar ou minimizar algum tipo de estatística sobre os dados empíricos. A estrutura do modelo é a estrutura de $\hat{y}(\cdot)$ e o conteúdo do regressor. Este tipo de modelagem é chamado de "*modelagem paramétrica*" (BOSSLEY, 1997), já que o modelo recai sobre o vetor de parâmetros w. Um problema para essa abordagem é a escolha de uma estrutura adequada para o modelo. A alternativa pode ser o uso de uma "*modelagem não-paramétrica*" onde, ainda que o modelo dependa do vetor de parâmetros w, o tamanho desse vetor não é fixado a priori. Assim, tanto $\hat{y}(\cdot)$ como o regressor não são fixados de antemão e existe uma adaptação da estrutura do modelo até que se atinja uma que satisfaça aos objetivos.

Os modelos neuro-nebulosos permitem tanto a inclusão de conhecimento especialista como a sua adaptação aos dados empíricos disponíveis (BOSSLEY, 1997). O conhecimento especialista é usado para definir uma estrutura inicial para o modelo e os pesos desse modelo são ajustados de forma a coincidir com os dados empíricos disponíveis. Em sistemas não bem conhecidos, conforme o número de variáveis de entrada cresce, o estabelecimento da estrutura do modelo torna-se mais difícil, o que também aponta para a necessidade de uma aproximação não-paramétrica para o modelo.

Podemos subdividir os modelos caixa preta (Black Box) em dois casos: a)

lineares e b) não-lineares.

3.2.2.1 - Modelos lineares

O relacionamento entre as entradas e saídas de um sistema linear no tempo é dado pela equação diferencial linear

$$\hat{y}(t) = -a_1 y(t-1) \ldots - a_{n_y} y(t-n_y) + b_1 u(t-1) \ldots + b_{n_u} u(t-n_u) + e(t) \qquad (3.6)$$

onde

$e(t)$ = ruído branco, que representa erros ou ruídos nas medidas

n_u e n_y = número de entradas e saídas que aparecem no modelo

Esta estrutura é conhecida como modelo ARX (BOSSLEY, 1997) (auto-regressivo com entradas exógenas) que pode ser representado por uma regressão linear simples tal como

$$\hat{y}(t) = \mathbf{x}^T(t)\,\mathbf{w} \qquad (3.7)$$

onde identificamos o vetor de parâmetros $\mathbf{w}$ e o regressor $\mathbf{x}(t)$ como

$$\mathbf{w} = [\, a_1 \ldots a_{n_y} \, b_1 \ldots b_{n_u} \,]^T \qquad (3.8)$$

$$\mathbf{x}(t) = [\, y(t-1) \ldots y(t-n_y) \, u(t-1) \ldots u(t-n_u) \,]^T \qquad (3.9)$$

3.2.2.2 - Modelos não-lineares

Ainda que os modelos lineares possam aproximar bem sistemas não-lineares, à medida que cresce a complexidade dos sistemas e suas não-linearidades tornam-se prevalecentes, os erros de aproximação também tornam-se significativos. Nesses casos, uma aproximação não-linear torna-se inevitavelmente necessária. A equação diferencial linear (3.6) é estendida para

$$\hat{y}(t) = f(\mathbf{x}(t), \mathbf{w}) \tag{3.10}$$

onde $f(.)$ é uma função não-linear parametrizada flexível conhecida como modelo NARMAX (BOSSLEY, 1997) (média móvel não-linear regressiva com entradas exógenas).

O regressor, ($\mathbf{x}(t)$), contém, além das entradas e saídas prévias do sistema, os erros de modelagem anteriores (ou saídas prévias do modelo) ou seja,

$$\mathbf{x}(t) = f[\,y(t-1), \ldots, y(t-n_y), u(t-1), \ldots, u(t-n_u), \hat{y}(t-1), \ldots, \hat{y}(t-n_{\hat{y}})\,]^T \tag{3.11}$$

O NARMAX é uma estrutura recorrente cujos modelos são notoriamente difíceis de se identificar (mesmo para estruturas lineares) e a saída obtida é uma função não-linear de seus pesos. Para superar essas dificuldades, comumente emprega-se o modelo NARX (BOSSLEY, 1997) (não-linear auto-regressivel com entradas exógenas). Seu regressor restringe-se às entradas e saídas do modelo ou, como nos modelos lineares,

$$\mathbf{x}(t) = [\,y(t-1), \ldots, y(t-n_y), u(t-1), \ldots, u(t-n_u)\,]^T \tag{3.12}$$

Dessa forma, todos os sistemas em estudo nesta tese são considerados poderem ser eficientemente aproximados pelo modelo NARX.

Na identificação de sistemas, a estrutura do modelo normalmente não é conhecida "a priori", de forma que uma função não-linear flexível $f(.)$ deve ser escolhida de maneira que possa modelar sistemas diferentes. A maioria das estruturas não-lineares aderem a

$$\hat{y}(t) = \sum_{i=1}^{p} a_i(x) w_i \tag{3.13}$$

onde

$a_i(.)$ = função de base não-linear

w_i = seu parâmetro (ou peso) associado

3.2.3 - Funções de base

O modelo resultante consiste numa combinação linear de funções de base não-linear. Aqui existem duas abordagens: i) Fixar as funções de base e então produzir modelos lineares generalizados e ii) Permitir a adaptação das funções de base produzindo modelos mais flexíveis (e mais complexos). As funções de base utilizadas são escolhidas comumente dentre: a- Funções de crista b- Funções de base radial c- Modelos polinomiais d- Produto tensorial e- Funções de base ondulatória. As funções de base ondulatória não são tratadas nesta tese. Vejamos os quatro primeiros casos.

3.2.3.1- Funções de crista

As funções de crista (ver figura 3.2) são representadas por (BOSSLEY, 1997)

$$a_i(\mathbf{x}) = \sigma\left(\beta_0 + \beta' \mathbf{x}\right) \tag{3.14}$$

onde $\sigma(\cdot)$ é uma função univariacional definida numa projeção linear das entradas. Por exemplo, $\sigma(\cdot)$ pode ser a função Sigmóide com

$$\sigma(u) = \frac{1}{1 + e^{-u}} \tag{3.15}$$

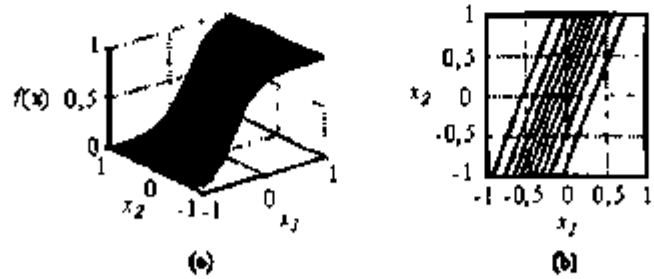


Figura 3.2 - Função de crista Sigmoidal (a) Mostra a superfície (b) Mostra os contornos lineares.

3.2.3.2 - Funções de base radial

As funções de base radial (ver figura 3.3) são representadas por

$$a_i(\mathbf{x}) = r(\| \mathbf{c}_i - \mathbf{x} \|_{w_i}) \tag{3.16}$$

onde $r(.)$ é uma função não-linear univariacional (BOSSLEY, 1997). A norma ponderada ($\| d \|_{w_i}$) é definida como dW^TWd onde W^TW é uma matriz simétrica a qual normalmente é escolhida para ser uma versão escalonada da matriz identidade, dessa forma produzindo uma função de base cujos contornos são radiais no espaço como, por exemplo, a função gaussiana, definida como

$$f(x) = exp(-x^2) \tag{3.17}$$

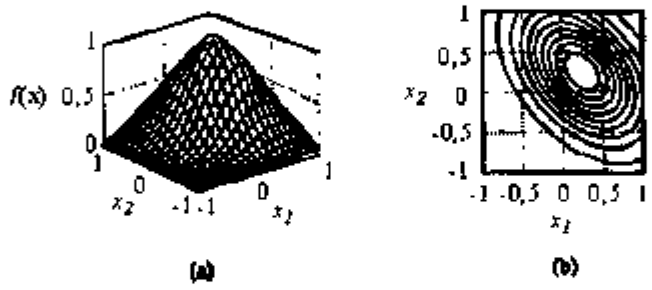


Figura 3.3 - Função Gaussiana (a) Saída da função. (b) Contornos radiais.

3.2.3.3 - Modelos polinomiais - As Splines

É um fato bem conhecido dos estudantes de ciências que podemos aproximar $n+1$ pontos de dados por polinômios de grau n. Contudo, existem casos onde tais funções podem levar a resultados errôneos. Nesses casos, podemos aplicar polinômios de menor ordem a um sub-conjunto dos pontos sob estudo. Um desses casos pode ser visto na figura 3.4. A função a ser modelada é uma função degrau que experimenta uma subida abrupta em $x = 0$. Os polinômios interpolantes de 2ª, 3ª e 4ª ordens experimentam oscilações indesejadas. Uma Spline cúbica, ao contrário, produz uma aproximação muito mais aceitável.

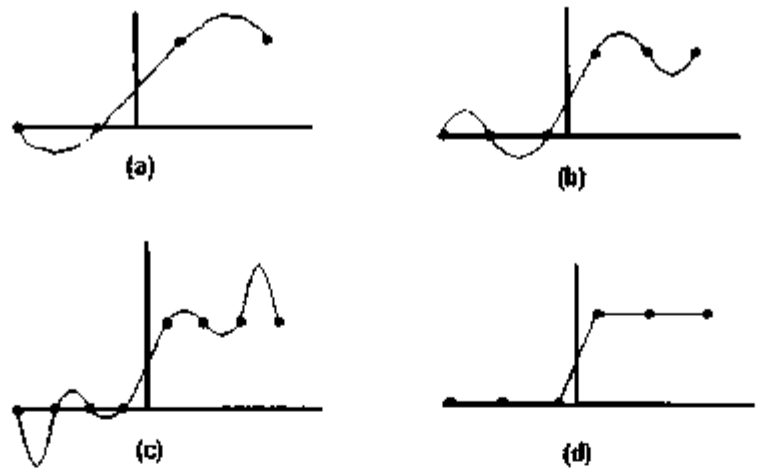


Figura 3.4 - Aproximação de uma função com polinômios de (a) 2° grau (b) 3° grau (c) 4° grau e (d) com splines cúbicas.

As Splines (CHAPRA et al., 1988, POWELL, 1981) são um tipo de funções polinomiais que realizam uma aproximação sobre conjuntos de dados unindo pedaços de sub-conjuntos desses dados. O nome "spline" vem da prática de se usar um pedaço de borracha achatada (cujo nome em inglês é "spline") para desenhar uma curva suave através de um conjunto de pontos (CHAPRA et al., 1988).

3.2.3.3.1 - Splines lineares

Sabemos que a conexão mais simples entre dois pontos é uma linha reta. Splines de primeira ordem para um grupo de pontos ordenados podem ser definidas como um conjunto de funções lineares (CHAPRA et al., 1988, POWELL, 1981) como

$$\begin{aligned} f(x) &= f(x_0) + m_0(x - x_0); \quad x_0 \le x \le x_1 \\ f(x) &= f(x_1) + m_1(x - x_1); \quad x_1 \le x \le x_2 \\ &\vdots \qquad \vdots \\ f(x) &= f(x_{n-1}) + m_{n-1}(x - x_{n-1}); \quad x_{n-1} \le x \le x_n \end{aligned} \tag{3.18}$$

onde m_i é a inclinação da reta que conecta os pontos, ou seja,

$$m_i = \frac{f(x_{i+1}) - f(x_i)}{x_{i+1} - x_i} \tag{3.19}$$

Estas equações podem ser usadas para avaliar o valor da função em qualquer ponto entre x_0 e x_n. Pode-se ver na figura 3.5(a) que nos pontos que conectam duas splines lineares adjacentes (chamado de nó) os valores das derivadas primeiras das funções tem uma mudança abrupta sendo assim descontínuas nesses pontos. Uma maneira de superar essa dificuldade é usar splines polinomiais de ordem superior, e assegurar a continuidade da derivada primeira (se necessário, da derivada segunda e assim por diante) nesses pontos pela imposição de um ou mais vínculos.

3.2.3.3.2 - Splines cúbicas

Para assegurar que as m-ésimas primeiras derivadas sejam contínuas nos nós, devemos usar splines de ordem m+1. Assim, splines quadráticas tem suas primeiras derivadas contínuas nos nós. O objetivo é encontrar funções polinomiais de segunda ordem para cada um dos intervalos entre os pontos e essas funções tem a forma geral

$$f_i(x) = a_i x^2 + b_i x + c_i \tag{3.20}$$

A figura 3.5(b) mostra, para os mesmos ponto de dados da figura 3.5(a), como fica a interpolação com splines quadráticas.

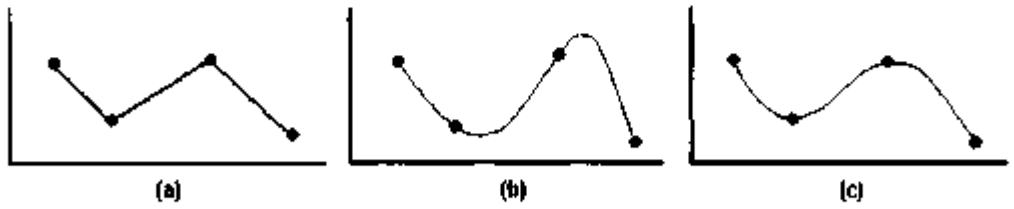


Figura 3.5 - Splines (a) lineares (b) quadráticas (c) cúbicas.

Quando usamos splines cúbicas procuramos conectar os nós com polinômios que têm a forma

$$f_i(x) = a_i x^3 + b_i x^2 + c_i x + d_i \qquad (3.21)$$

Pela sua própria definição, as condições de contorno dessas splines são:

(a) Os valores das funções devem ser iguais nos nós

$$f(x_n) = f(x_{n-1})$$

(b) As derivadas primeiras das funções que se conectam a um nó devem ser iguais

$$f'(x_n) = f'(x_{n-1})$$

(c) As derivadas segundas das funções que se conectam a um nó devem ser iguais.

$$f''(x_n) = f''(x_{n-1})$$

A figura 3.5(c) apresenta o gráfico de uma spline cúbica conectando os mesmos pontos das figuras 3.5(a) e (b).

3.2.3.4 - Produto tensorial

Produtos tensoriais são funções multivariacionais compostas do produto de várias funções univariacionais como, por exemplo, as Splines tensoriais . Sua forma é

$$a_i(\mathbf{x}) = \prod_{i=1}^{n} a_i'(x_i) \qquad (3.22)$$

onde $a_i'(x_i)$ são funções univariacionais simples (BOSSLEY, 1997). Splines tensoriais tem um subconjunto conhecido como B-Splines. Uma B-Spline quadrática aparece na figura 3.6. onde se pode observar que seus contornos radiais são perpendiculares aos

eixos. As funções obtidas por produto tensorial têm uma série de características desejáveis. Entre elas, as saídas podem ser eficientemente geradas, possuem uma representação esparsa, são polinomiais (aos pedaços) ao grau desejado ou seja, são continuamente diferenciáveis e seu comportamento pode representar regras nebulosas.

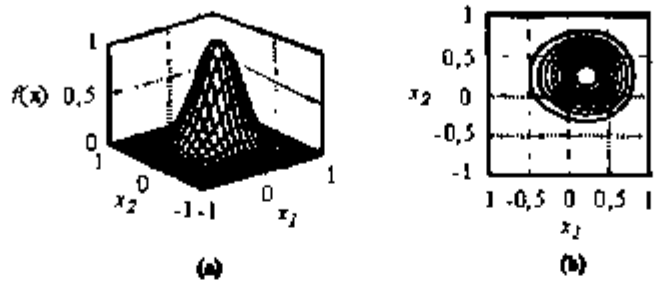


Figura 3.6 - B-Spline quadrática. (a) Saída da função. (b) Contornos radiais.

3.2.3.4.1 - B-splines

As B-splines (BOSSLEY, 1997, POWELL, 1981, RICE, 1983) são funções tensoriais que são semelhantes às Splines, mas com características diferentes. Elas também são funções polinomiais por partes que modelam funções com um grau de suavidade definível. A ordem dos polinômios locais que aproximam as funções definem a ordem da B-spline, que é denotada por *k*. A figura 3.7 mostra uma esquematização das B-splines de ordens 2, 3 e 4 bem como a aproximação que elas permitem para uma determinada função. As B-splines de ordem 1 são funções constantes, que não nos interessam. Como notamos, os triângulos podem ser, obedecidos certos vínculos, considerados B-splines de ordem 2.

As B-splines formam, na verdade, um conjunto de funções de base univariacionais que são definidas numa série de pontos chamados nós. Esses nós são os intervalos nos quais os polinômios são definidos. As funções de base são calculadas por meio de algoritmos estáveis (BOSSLEY, 1997, POWELL, 1981) que permitem o cálculo de todos os polinômios necessários para a modelagem do espaço de entrada. Essas curvas tem a interessante propriedade de particionar a unidade, de forma que podem ter uma aplicabilidade interessante em modelos neuro-nebulosos. Se definirmos

um conjunto de n funções de pertinência B-splines univariacionais de ordem k sobre cada uma das variáveis de entrada x_i , então uma função B-spline multivariacional $\mu A_i(\mathbf{x})$ pode ser calculada a partir do produto das n funções univariacionais $N_{k_i}^{i}(x_j)$ como

$$\mu A_i(\mathbf{x}) = \prod_{j=1}^{n} N_{k_i}^{i}(x_j) \tag{3.23}$$

onde i_j representa o índice da função de base definida em x_j, o qual contribui para a i-ésima função de pertinência multivariacional.

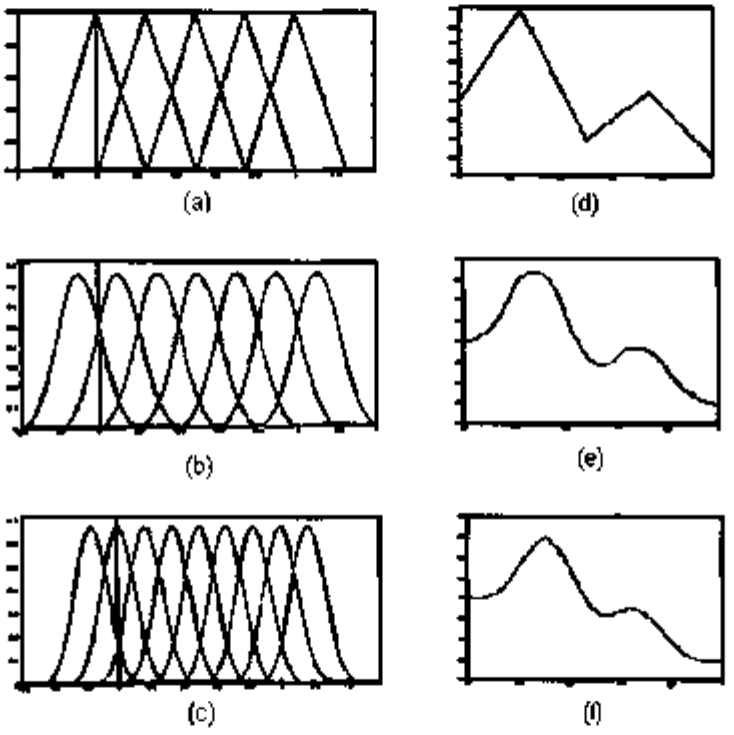


Figura 3.7 - B-Splines (a) ordem 2 (b) ordem 3 (c) ordem 4. (d) (e) (f) Uma função interpolada como B-Splines de ordens 2, 3 e 4, respectivamente.

3.2.4 - Estimativa de parâmetros

A estimativa dos parâmetros do modelo consiste no trabalho de, ao se estabelecer uma estrutura adequada ao modelo do sistema em análise, determinar-se o

grau de complexidade que deve ser associado à tal estrutura (BOSSLEY, 1997). Ou seja, dada uma estrutura de referência, ela pode fornecer saídas baseadas seja em simples estimativa direta ou por meio de cálculos bastante complexos. O quanto de acuidade, generalização e complexidade desejamos ou é efetivamente necessário é um dilema por vezes de difícil solução.

3.2.4.1 - O trabalho sobre os dados

Dado um conjunto de dados $D_N = \{ y(t) , \mathbf{x}(t) \}_{t=1}^{N}$ e uma estrutura para o modelo $f(\mathbf{x}(t) , \mathbf{w})$, trabalha-se fazendo-se inferências sobre **w** de tal forma que o modelo seja uma boa aproximação do sistema real. Assim, devemos minimizar

$$J_N = \frac{1}{N} \sum_{t=1}^{N} \left[\left(y(t) - \hat{y}(\mathbf{x}(t) , \mathbf{w}) \right)^2 \right] \qquad (3.24)$$

que nada mais é que uma função de custo do erro médio quadrático e onde os pares de treinamento são considerados graus de liberdade. Em identificação de sistemas, este método é chamado de "abordagem de predição de erros" (BOSSLEY, 1997). A complexidade da minimização é dependente da estrutura do modelo escolhida (função de saída do modelo em relação aos pesos lineares e não-lineares).

3.2.5 - Os dados disponíveis

Numa modelagem empírica, o modelo resultante é sempre altamente dependente tanto da disponibilidade quanto da qualidade dos dados. Para bem generalizar, um modelo depende que o conjunto de dados D_N represente bem o sistema sob análise (BOSSLEY, 1997). Quando existem muitas dimensões, a capacidade de generalização de qualquer modelo sofre em vista de que nem todas as regiões do espaço de entrada são suficientemente representadas pelos dados sendo isso uma conseqüência direta da "*maldição da dimensionalidade*".

Entre as técnicas utilizadas para superar esse problema algumas tem-se mostrado particularmente satisfatórias. O pré-processamento dos dados para extração de características, por exemplo, permite que o espaço de entrada seja reduzido dimensionalmente, simplificando o trabalho de modelagem. A despeito de qualquer pré-processamento dos dados, deve ser identificado um modelo suficientemente parcimonioso de tal modo que a equivalência entre os dados e a estrutura do modelo seja adequada.

3.2.6 - Escolha de modelos

Havendo um conjunto de diferentes modelos candidatos a descrever um sistema, devemos eleger um único que melhor se encaixe nos dados disponíveis. Existem alguns critérios técnicos para isso (BOSSLEY, 1997) como, por exemplo, o teste de hipóteses, o teste LDR, o AIC, o critério FPE, o MDL, o GCV e o NIC, entre outros. Todos eles aderem ao princípio da parcimônia.

Nesta tese, o modelo mais adequado é selecionado segundo o critério de diminuição do valor do EMQ J da equação (3.24).

3.3 - Modelagem neuro-nebulosa

Um sistema nebuloso é, a grosso modo, um modelo não-linear cujo comportamento pode ser descrito por um conjunto de regras lingüísticas. A maior crítica aos sistemas nebulosos é que eles são *matematicamente* opacos, não existindo um representação matemática formal de seu comportamento. A vagueza das descrições lingüísticas tornam os sistemas nebulosos baseados somente em conhecimento qualitativo inadequado para modelar mesmo sistemas simples. Para superar essas dificuldades, dados de teste devem ser usados para ajustar e validar o modelo levando aos modelos neuro-nebulosos. Tais técnicas permitem que tanto uma descrição lingüística como dados empíricos sejam utilizados na modelagem do sistema.

Teorema:

- Quando:

1) Operadores algébricos são usados para implementar funções nebulosas lógicas

2) Entradas definidas são fuzificadas por nebulização singela

3) Emprega-se desfuzificação por centro de gravidade

Então:

O mapeamento de entrada-saída de um sistema nebuloso pode ser representado por uma combinação linear de funções (normalizadas) de pertinência nebulosas na forma

$$\hat{y}(x) = \sum_{i=1}^{r} \left(\frac{\mu A_i(x)}{\sum_k \mu A_k(x)} \right) w_i \qquad (3.25)$$

onde w_i é referido como "peso".

-

Assim, a saída resultante de uma estrutura neuro-nebulosa é simplesmente uma soma ponderada de funções de base não-linear (ver figura 3.8). Assumindo-se que as funções são fixas, o mapeamento não-linear é controlado pelo conjunto de parâmetros ajustáveis w, conhecidos como *pesos* e que podem ser identificados a partir dos dados empíricos.

Comparativamente com técnicas convencionais conhecidas como "modelos lineares generalizados" (BOSSLEY, 1997), os sistemas nebulosos podem ser considerados como aproximadores universais que podem aproximar qualquer função contínua definida num domínio compacto com uma precisão arbitrária. Enquanto os pesos dos modelos lineares generalizados são adaptados para seguir os dados empíricos produzindo, assim, modelos opacos, os sistemas nebulosos fornecem uma forma de visualizar essa adaptação tornando-se modelos mais transparentes.

As propriedades desejadas de um sistema neuro-nebuloso são: 1- representação lingüística das variáveis; 2- computacionalmente eficiente; 3- espaços compactos; adicionalmente, 4- partição da unidade

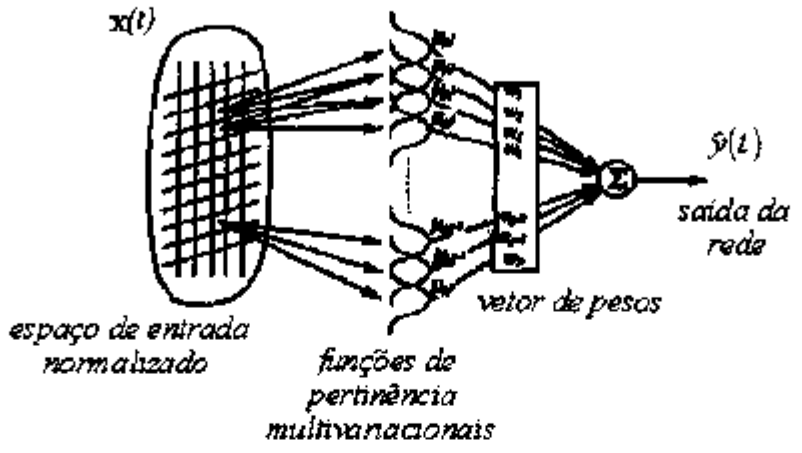


Figura 3.8 - Estrutura básica de um modelo neuro-nebuloso.

Sistemas nebulosos possuem a propriedade de poder particionar a unidade em suas funções de base, o que é uma propriedade desejável para uma boa capacidade de generalização nos modelos lineares generalizados. Graficamente, as funções de base devem ter formas parecidas como as que foram vistas na figura 2.17. A figura 3.9(a) (BOSSLEY, 1997) representa um particionamento mal comportado pois após a normalização das funções obtemos um espectro altamente irregular, como pode ser visualizado na figura 3.9(b).

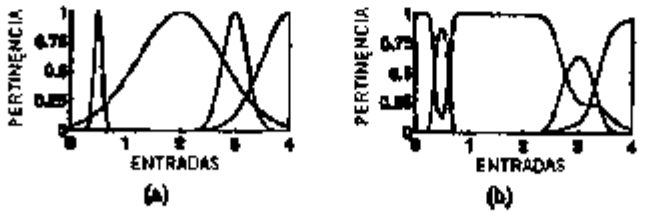


Figura 3.9 - Não partição da unidade. (a) Funções definidas. (b) Funções normalizadas.

3.3.1 - Limitações da modelagem neuro-nebulosa

A modelagem neuro-nebulosa não é de forma alguma uma panacéia, sofrendo de várias limitações. Uma dificuldade natural é a da escolha do modelo a ser utilizado, que já foi analisada. Existem outras, das quais aqui aborda-se algumas.

3.3.1.1 - A maldição da dimensionalidade

O número de conjuntos de entrada multivariacionais p de um modelo nebuloso pode ser representado por (BOSSLEY, 1997)

$$p = \prod_{i=1}^{n} P_i \tag{3.26}$$

que é uma função exponencial das dimensões de entrada, já que P_i é o número de conjuntos nebulosos de entrada de uma determinada variável i. Conforme o número n de dimensões cresce linearmente, o tamanho do modelo p cresce exponencialmente (ver figura 3.10).

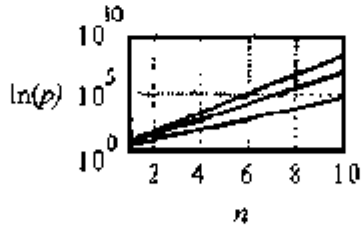


Figura 3.10 - Aumento exponencial do modelo x aumento linear das entradas.

Dessa forma o custo computacional e de implementação também crescem exponencialmente a patamares proibitivos impossibilitando a aplicação prática da modelagem neuro-nebulosa em muitas dimensões. Como desvantagem adicional, tem-se uma perda significativa em termos de interpretabilidade.

3.3.1.2 - Quantidade de dados necessária

Imagine-se que se considere que N=100 amostras de dados sejam suficientes para representar o espaço de estudo de um determinado sistema univariacional. Em um sistema n-dimensional de mesma densidade, tornam-se necessárias N^n amostras para manter-se a mesma representação. Em altas dimensões, a reunião de um conjunto de dados de tal magnitude que seja razoavelmente representativo do mapeamento de entradas e saídas do sistema é , por qualquer método, um desafio de alta complexidade.

3.3.1.3 - Qualidade dos dados

Essencialmente, os problemas associados com os dados são devidos à sua quantidade, distribuição e ruído condições essas além do esforço de controle do projetista. Caso os dados estejam pobremente distribuídos no espaço de entrada, as funções de base e/ou seus pesos associados podem não ser excitados. Por exemplo, a figura 3.11 ilustra o caso de um componente cuja resistência diminui com a temperatura (BOSSLEY, 1997), sendo aquela nula quando esta é de 25°C. Os dados disponíveis em (a) ilustram esse comportamento, mas não os dados disponíveis em (b).

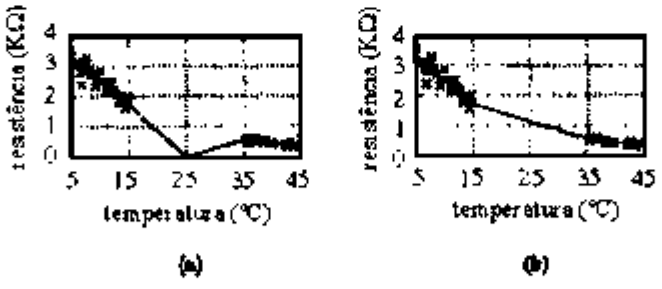


Figura 3.11 - Dados disponíveis (a) Suficientes. (b) Insuficientes.

De acordo com as técnicas de otimização, pesos não excitados são considerados não utilizados e, assim, fixados em zero. Pesos zerados, é claro, em nada contribuem para o modelo. Uma das técnicas de se evitar esses casos é fazer com que os pesos eventualmente zerados recebam um valor que seja a média dos valores dos pesos vizinhos. A maior parte dos problemas associados com a qualidade dos dados são

devidos a uma incoerência entre a estrutura do modelo e os dados. Uma melhor correspondência entre eles pode ser conseguida por meio das técnicas de adaptação de estrutura. Estas técnicas funcionam eficientemente para minimizar os efeitos dos inevitáveis problemas associados aos dados empíricos, como ruídos.

3.3.1.4 - Conhecimento especialista

Em sistemas neuro-nebulosos, quando está disponível conhecimento especialista de qualidade, pode-se usá-lo para compensar alguns dos problemas associados à qualidade dos dados. Caso contrário, tanto a estrutura do modelo como os pesos das funções de base devem ser extraídos dos dados empíricos. Pelas características desejadas de um sistema neuro-nebuloso, a estrutura deve não apenas se comportar quantitativamente como o sistema real mas também devem fornecer uma interpretação qualitativa (lingüística) do sistema em suas regras nebulosas.

Aqui, ocorre um sério problema quando as funções de base são mal condicionadas, tanto no estabelecimento da estrutura como no seu desenvolvimento (BOSSLEY, 1997). Em determinados casos os dados estão tão distantes da realidade sugerida pelas funções que freqüentemente recaem sobre suas fronteiras provocando problemas de convergência, generalização pobre e funções de pertinência excessivamente sobrepostas ou inconvenientemente espaçadas.

3.4 - Bases de regras redundantes

Pelos motivos acima, a modelagem das regras constitui-se num passo importante da abordagem neuro-nebulosa. Regras inconsistentes podem levar a modelos que não convirjam enquanto que uma base composta por regras redundantes leva a um modelo de alta complexidade e de baixa capacidade de generalização. Já vimos que o não particionamento da unidade entre as funções de pertinência leva a resultados inconvenientes em termos de normalização. Existem ainda outros aspectos indesejáveis que ocorrem quando não se observa esse particionamento.

Observe-se as regras da figura 3.12(a). As funções de pertinência se superpõem largamente, por volta de 90%, implicando num modelo altamente redundante. Na figura 3.12(b) vemos que as funções de pertinência estão construídas de forma que existem regras que cobrem um domínio muito pequeno do problema.

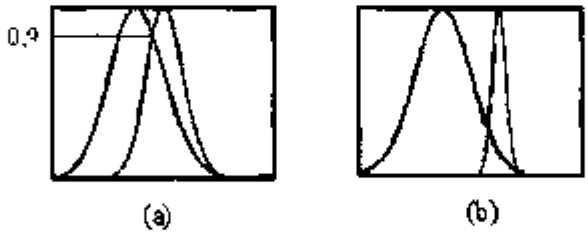


Figura 3.12 - (a) Funções de pertinência sobrepostas. (b) Domínio pequeno.

Tais regras podem, em geral, ser eliminadas e/ou fundidas sem diminuição apreciável da acuidade do modelo. Por exemplo, aplicando-se métodos adequados, a base de regras extremamente confusa da figura 3.13(a) passa a ter o aspecto muito mais razoável que se vê na figura 3.13(b).

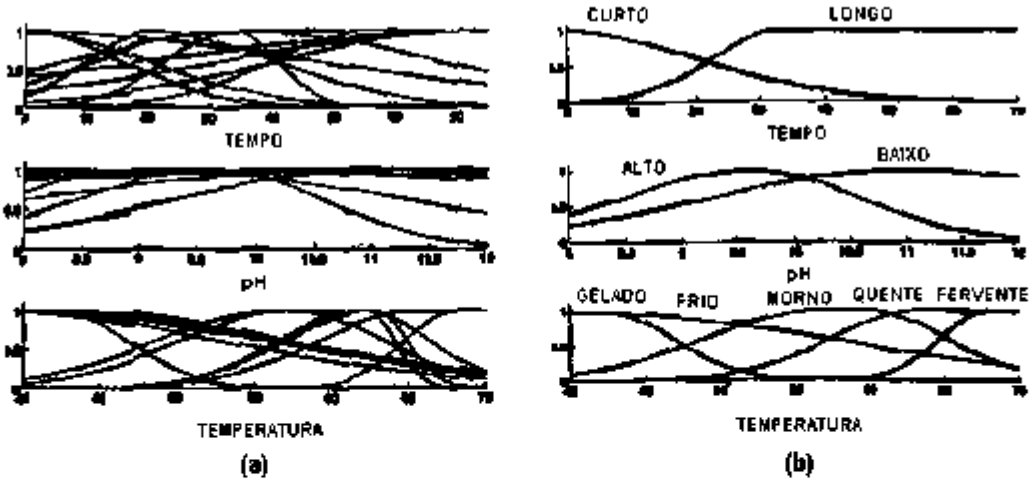


Figura 3.13 - Modelo neuro-nebuloso. Bases de regras (a) original e (b) revista.

3.5 - Adaptação da estrutura do modelo

De forma a aumentar a interpretabilidade dos modelos obtidos, adapta-se a posição, a quantidade e a forma das funções de base para reduzir o número de graus de liberdade que existem nos modelos neuro-nebulosos (BOSSLEY, 1997). As figuras 3.14 (a) e (b) mostram uma adaptação na quantidade das funções de pertinência enquanto que as figuras 3.14 (c) e (d) mostram uma típica adaptação de posição e forma de funções de base. Como se observa, o acréscimo de uma nova função de pertinência deve levar a uma modificação nas formas de outras funções de pertinência já existentes de maneira que se mantenha o particionamento da unidade entre funções adjacentes.

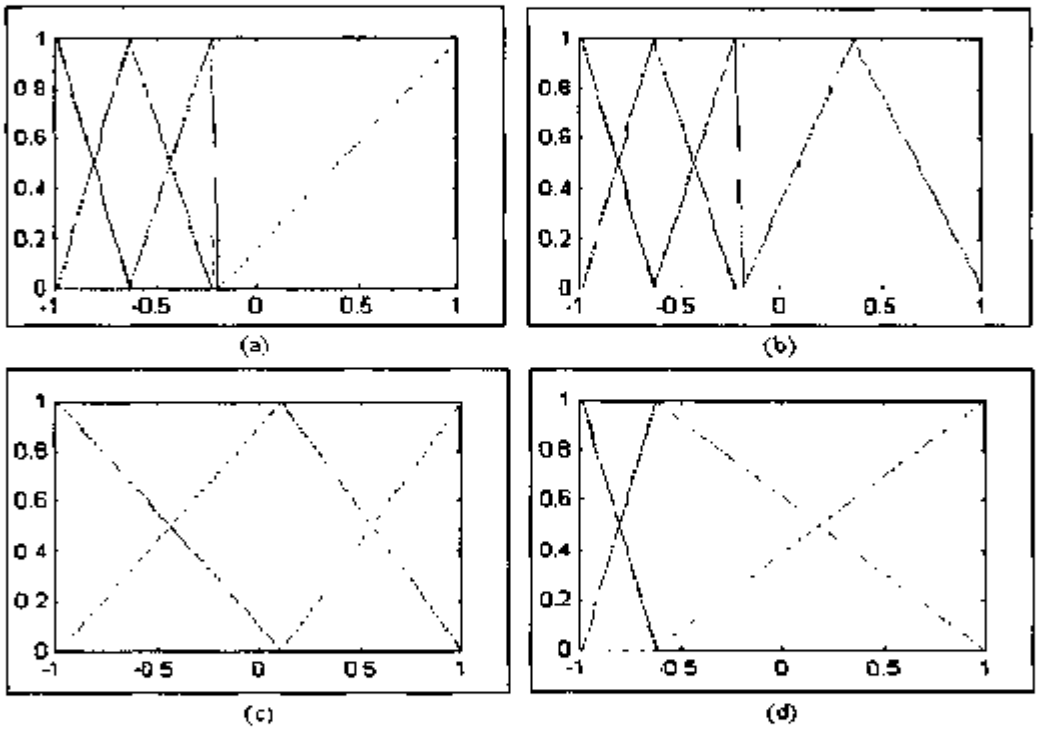


Figura 3.14 - Adaptação da estrutura de um modelo.

O objetivo final é produzir um modelo que não só represente fidedignamente os pontos de dados usados para a identificação mas que também generalize bem. O princípio da parcimônia fornece um balanço entre as fontes competitivas de erro e deve ser utilizado durante a identificação do sistema. Isso pode ser conseguido pelo uso das técnicas de adaptação. A adaptação da estrutura seja por meio do aumento das funções

de base, seja por adaptação das formas dessas funções, tenta encontrar a estrutura mais adequada para o modelo.

3.6 - Modelo de Takagi-Sugeno

Essa modelagem pode ser representada por uma série de regras nebulosas na forma

$$R_i : \underline{SE}\ (x \text{ é } A_i)\ \ \underline{ENTÃO}\ \ (y = f_i(x_i)) \tag{3.27}$$

onde $f_i(x)$ definido em $x_i \subset x$ é um modelo local usado para aproximar a resposta do sistema na região do espaço de entrada representado pelo antecedente. Esta é uma inferência nebulosa de tipo 3 e a saída do modelo é calculada como sendo a soma normalizada

$$\hat{y} = \frac{\sum_{i=1}^{n} \mu A_i(x) f_i(x_i)}{\sum_{i=1}^{n} \mu A_i(x)} \tag{3.28}$$

conforme foi esquematizado na figura 2.12. O modelo ANFIS, que será visto no próximo capítulo, pode utilizar este tipo de estrutura. A forma da função $f_i(x)$ não é previamente especificada. Assim, a grosso modo, toda e qualquer função definida em um domínio real pode ser usada em (3.28), desde as mais exóticas até as mais simples, como as funções constantes.

3.6.1 - Modelo de Takagi-Sugeno de ordem zero

Valendo-se do fato de não existir, a princípio, restrições quanto à forma de $f_i(x)$ em (3.28), usando um único valor como uma partição de saída ao invés de funções de pertinência convencionais ou de formas complexas, Takagi e Sugeno mostraram (TAKAGI et al., 1985) que podemos construir regras nebulosas adequadas e de formulação concisa na forma

$$\underline{\text{SE}} \quad x_1 \text{ é } A_1 \quad \underline{\text{ENTÃO}} \quad y = \theta_1 \tag{3.29}$$

onde θ_1 é um puro número real adequado conhecido como *singleton*.

Para duas variáveis de entrada, as regras da base são da forma

$$\underline{\text{SE}} \quad x_1 \text{ é } A_1 \quad \underline{\text{E}} \quad x_2 \text{ é } B_2 \quad \underline{\text{ENTÃO}} \quad y = \theta_1 \tag{3.30}$$

Apesar de sua extrema simplicidade, prova-se (ver capítulo 5) que sistemas neuro-nebulosos assim construídos podem aproximar uma grande variedade de funções com uma precisão arbitrária. As partições de saída assumem as formas exibidas na figura 3.15

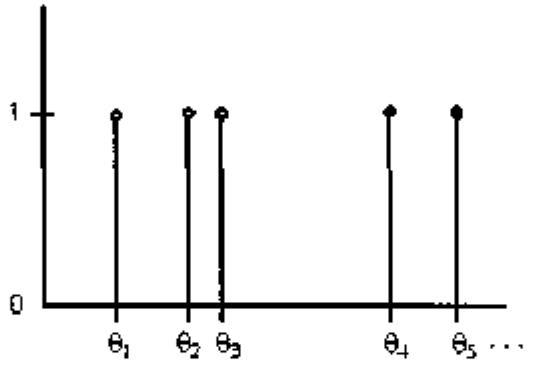


Figura 3.15 - Singletons definindo as partições de saída.

3.6.2 - Modelo de Takagi-Sugeno de ordem um

Uma maneira mais complexa de se expressar regras da forma de (3.27) é utilizar um polinômio como $f_i(x)$. Regras assim construídas possuem, em geral a forma

$$\underline{\text{SE}} \quad x_1 \text{ é } A_1 \quad \underline{\text{ENTÃO}} \quad y = p_1 x + r_1 \tag{3.31}$$

e um sistema cuja base de regras tem regras desse formato é conhecido como *modelo de Takagi-Sugeno de ordem um* (TAKAGI et al., 1985) No caso de haverem duas

variáveis de entrada, (3.31) evolui para

$$\underline{\text{SE}}\ \ x_1 \text{ é } A_1\ \ \underline{\text{E}}\ \ x_2 \text{ é } B_2\ \ \underline{\text{ENTÃO}}\ \ y = p_1 x + q_1 y + r_1 \qquad (3.32)$$

A extensão para m variáveis é direta, onde se particiona a saída com polinômios de m+1 termos.

Para o objetivo deste trabalho, a saída de um sistema nebuloso pode ser determinada diretamente, sem necessidade de defuzificação, em sistemas nebulosos de Takagi-Sugeno (TSK). No modelo de TSK de ordem zero, as conclusões das regras são, então, valores numéricos reais armazenados num vetor θ. Considerando-se partições nebulosas normalizadas e ortogonais que obedeçam às condições (2.33) e (2.34), a saída de tal modelo é dada por (EVSUKOFF et al., 2000):

$$\hat{y}_0 = \mathcal{X}_0\theta = \sum_{1 \le i \le n} A_i(x_0).\theta_i \qquad (3.33)$$

Esta equação mostra a saída de um modelo do tipo TSK de ordem zero como sendo uma combinação linear de modelos não-lineares locais. Em modelos nebulosos, a saída $\hat{y}$ ($\hat{y} \in Y$) é calculada a partir da entrada $x_0 \in X$ pelas etapas de fuzificação, inferência e defuzificação. Na fuzificação a descrição $\mathcal{X}_0$ é calculada em (2.32) a partir da entrada x_0. A etapa de inferência calcula a descrição de saída $\mathcal{Y}_0$ baseando-se em $\mathcal{X}_0$ e na base de regras Φ combinadas com o uso de um operador composição-projeção ($\circ$) na forma (EVSUKOFF et al., 2000)

$$\mathcal{Y}_0 = \mathcal{X}_0 \circ \Phi \qquad (3.34)$$

Na etapa de defuzificação a saída $y_0 \in Y$ é calculada a partir da descrição de saída $\mathcal{Y}_0$ de acordo com a partição de saída $\mathbf{B}(y)$. A descrição de saída $\mathcal{Y}_0$ em (3.34) pode ser calculada para todo $Bj \in \mathbf{B}$ na forma do produto matricial (EVSUKOFF et al., 2000)

$$Y_0(B_i) = \sum_{1 \le i \le n} X_0(A_i)\Phi(A_i, B_i) \tag{3.35}$$

A defuzificação da saída pode ser calculada vetorialmente por (EVSUKOFF et al., 2000)

$$\hat{y}_0 = \mathbf{\Psi}_0 \mathbf{b} \tag{3.36}$$

Agora, as equações (3.33) e (3.36) são iguais se

$$\theta = \Phi \mathbf{b} \tag{3.37}$$

Assim, a identificação de um modelo TSK de ordem zero é considerada equivalente à identificação de um modelo nebuloso lingüístico, desde que os conjuntos nebulosos que definem a partição de saída possam ser estabelecidos por um especialista (EVSUKOFF et al., 2000). Este fato é a base dos sistemas apresentados no capítulo 5 e se constitui também no fundamento desta tese.

3.7 - Considerações

Vimos aqui que a modelagem de regras nebulosas deve ser feita de forma criteriosa de maneira a que uma base de regras possa ter não só boa interpretabilidade lingüística como conter regras que efetivamente cubram um sub-domínio de interesse. Devem ser evitadas tanto a superposição de funções de pertinência como aquelas que cubram um domínio muito pequeno no espaço de entrada. Essa última condição algumas vezes não pode ser evitada, por exemplo, em pontos de descontinuidade. Definida uma boa forma de modelagem, o procedimento de inferência nebulosa deve ser escolhido de forma a simplificar o algoritmo de desfuzificação e uma boa escolha pode recair entre os modelos TSK de ordens zero ou um.

A saída de um modelo TSK de ordem zero pode ser considerada uma combinação linear de modelos não-lineares locais e a identificação de tal modelo é

equivalente à identificação de um sistema neuro-nebuloso de partições em forma lingüística.

A modelagem paramétrica exige o ajuste dos pesos das funções de base o que, além de ser consumidor de recursos computacionais, pode levar a bases de regras extensas e de difícil interpretação. O uso de uma modelagem não-paramétrica parece ser a melhor alternativa na maioria dos casos, mesmo havendo um aumento da quantidade de pesos a serem calculados. A maneira mais fácil de se evitar o ajuste dos pesos é, simplesmente, não utilizá-los, fazendo-se o ajuste diretamente nas funções de base.

Essa é a abordagem desta tese, onde a abordagem não-paramétrica para a identificação de sistemas permite uma maior transparência lingüística para os modelos obtidos, devido à exigência do particionamento da unidade entre funções de pertinência adjacentes.

Neste capítulo, novas parametrizações de curvas adequadas à representação de conjuntos nebulosos foram apresentadas de forma sucinta.

4

Modelos adaptáveis

4.1 - Introdução

Em sistemas de controle por lógica nebulosa, os testes, depuração e sintonia da base nebulosa de conhecimento muitas vezes é trabalhosa e de grande consumo de tempo (MAEDA et al., 1993). Essas atividades são mais difíceis de serem executadas nesses casos do que em sistemas especialistas convencionais. Ao invés dos sistemas especialistas tradicionais nos quais as regras vão sendo gradativamente descartadas até que apenas uma seja ativada, a lógica nebulosa usa uma variedade de regras simultâneas de forma a extrair um resultado por inferência nebulosa. Dessa forma, é muito mais difícil entender-se o comportamento da base como um todo. Em alguns casos limite, grandes bases de conhecimento nebuloso podem ser de tão baixa qualidade que sejam impossíveis de se aplicar a situações reais.

Para superar esses problemas, alguns pesquisadores propuseram métodos (MAEDA et al., 1991) como, por exemplo, implementar procedimentos nebulosos de inferência em redes neuronais e então aplicar algoritmos de treinamento nessas redes de forma a evitar as dificuldades em adquirir e refinar o conhecimento nebuloso. Tais métodos eram, contudo, na sua maioria, muito simples, apenas modificando as implicações das operações booleanas "E" e "OU" durante o processo de aprendizado. Isso prejudica a manutenção e a clareza do conhecimento nebuloso que são duas das maiores vantagens da lógica nebulosa.

Com a intenção de contornar esse problema, alguns pesquisadores japoneses propuseram uma outra técnica para usar bases nebulosas usando uma representação em rede do procedimento de inferência nebulosa. Chamaram a essa técnica de "Flip-Net" (MAEDA et al., 1993), a qual representa a estrutura lógica tanto das regras nebulosas como das funções de pertinência. Usando a Flip-net eles introduziram um algoritmo de treinamento supervisionado para refinar as funções de pertinência. Nesse método, um

algoritmo tipo backpropagation refina as formas das funções de pertinência e os relacionamentos lógicos entre as regras e as funções não são alterados.

4.2 - A Flip - Net

As regras nebulosas para "n" entradas e uma saída podem ser escritas como:

$$\begin{aligned}
&\underline{\text{SE}}\ x_1 \text{ é } A_{11}\ \underline{\text{E}}\ x_2 \text{ é } A_{21} \ldots \underline{\text{E}}\ x_n \text{ é } A_{n1}\ \underline{\text{ENTÃO}}\ y \text{ é } B_1 \\
&\underline{\text{SE}}\ x_1 \text{ é } A_{12}\ \underline{\text{E}}\ x_2 \text{ é } A_{22} \ldots \underline{\text{E}}\ x_n \text{ é } A_{n2}\ \underline{\text{ENTÃO}}\ y \text{ é } B_2 \\
&\quad\vdots \\
&\underline{\text{SE}}\ x_1 \text{ é } A_{1m}\ \underline{\text{E}}\ x_2 \text{ é } A_{2m} \ldots \underline{\text{E}}\ x_n \text{ é } A_{nm}\ \underline{\text{ENTÃO}}\ y \text{ é } B_m
\end{aligned} \qquad (4.1)$$

Onde $x_1, x_2, \ldots, x_n$ são as variáveis de entrada. "y" é a variável de saída e "m" é o número de regras nebulosas. A_{ij} e B_j são os conjuntos nebulosos. Aqui, também usa-se A_{ij} e B_j como as funções de pertinência.

Reconhecemos que acontece de algumas das regras referirem-se apenas a uma parte das "n" variáveis de entrada e que também o mesmo conjunto nebuloso pode ser usado em várias regras nebulosas. Nesses casos, simplesmente assume-se que $A_{ij} = 1$ ou que $A_{ij} = A_{ik}$ para alguns i,j e k.

Estendendo o procedimento de inferência nebulosa apresentado antes para um caso geral, o procedimento seria, por exemplo, que a regra de grau W_j é dada pelo menor dos graus dos conjuntos nebulosos usados na parte SE (correlação de mínimos). Ou seja,

$$W_j = \min \{ A_{ij}(x_i) \mid i \in [1,n] \} \qquad (4.2)$$

A função B(z) sintetizada na saída é:

$$B(z) = \max \{ W_j B_j(z) \mid j \in [1,m] \} \qquad (4.3)$$

Onde z é uma variável muda para a integral de defuzificação. Agora, de acordo com a equação (2.26), a saída da inferência nebulosa "$\hat{y}$" tomada de forma contínua é:

$$\hat{y} = \frac{\int B(z).zdz}{\int B(z)dz} \tag{4.4}$$

ou, em cálculos digitais onde z_k (k = 1,..., K) representa os valores discretizados de z:

$$\hat{y} = \frac{\sum_{k=1}^{K} B(z_k).z_k}{\sum_{k=1}^{K} B(z_k)} \tag{4.5}$$

Ocorre que a Flip-net é uma rede de fluxo computacional do procedimento de inferência nebulosa descrito acima. Essa é uma inferência nebulosa de tipo 2, onde a saída é calculada pelo método de correlação de máximos e mínimos. A figura 4.1 representa uma flip-net com duas variáveis de entrada e duas regras nebulosas (n=2, m=2). Os cálculos correm da direita para a esquerda, como é usual. Os neurônios na primeira coluna representam as variáveis muda e de entrada. Devido ao somatório em (4.5) um neurônio deve representar a variável muda de saída z, cujo valor é um vetor $(z_1,z_2,....z_k)$.

Os neurônios na segunda coluna representam as funções de pertinência de entrada e saída A_{ij} e B_i. Os neurônios na terceira coluna são para a operação de cálculo de mínimo em (4.2). O penúltimo neurônio representa o cálculo da síntese da função de pertinência de saída B(z) definida em (4.3) e o último neurônio satisfaz a (4.5). Na figura, as linhas pontilhadas representam conexões nas quais trafegam vetores.

Note que a Flip-net fornece uma perfeita representação do procedimento de inferência definido nas equações (4.2) a (4.5). Assim, a saída de cada neurônio tem um significado definido, tal como: pertinência ao conjunto nebuloso, regra ou saída.

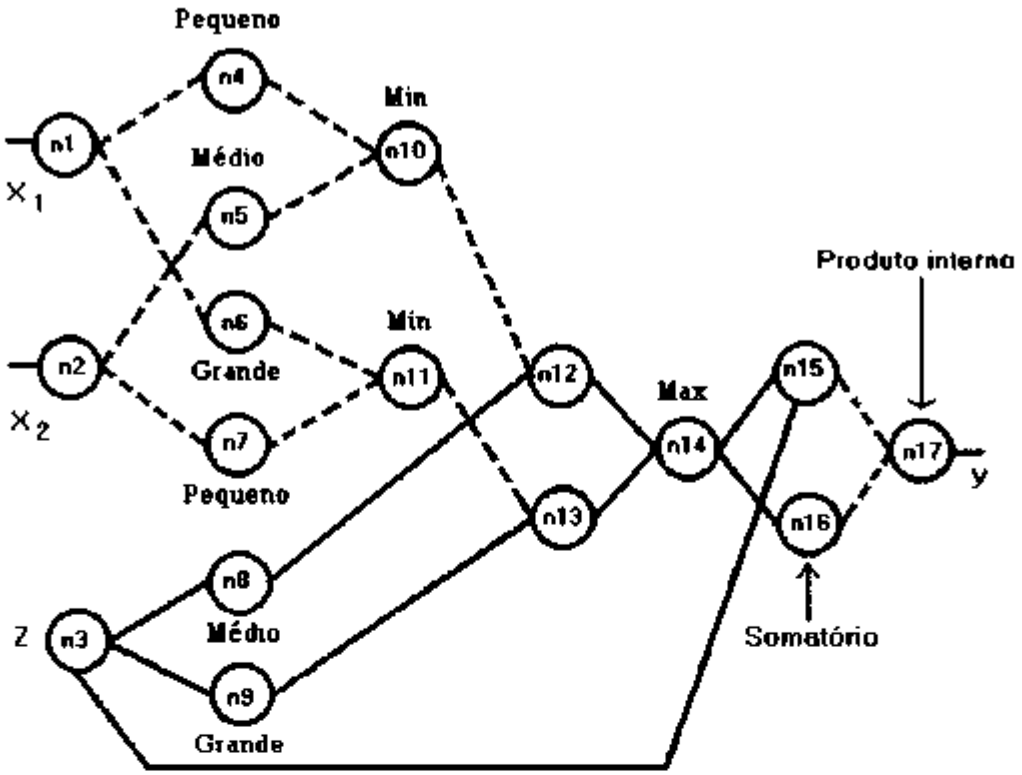


Figura 4.1 - Esquema da Flip-net que reproduz as regras dadas na tabela 4.1.

4.2.1 - Uso de backpropagation para refinar as funções de pertinência

Um dos procedimentos mais demorados no desenvolvimento de sistemas nebulosos consiste na determinação da forma das funções de pertinência. Para o refinamento da Flip-net foi utilizado um algoritmo de treinamento supervisionado.

Usando o procedimento de inferência nebuloso definido pelas equações (4.2) a (4.5), a entrada $x = (x_1, x_2, ..., x_n)$ é usada para calcular a saída "$\hat{y}$". Assim, escrevemos o relacionamento entrada-saída como

$$\hat{y} = f(x, \alpha) \tag{4.6}$$

onde $\alpha = (\alpha_1, \alpha_2, ..., \alpha_p)$ são os parâmetros que definem a forma das funções de pertinência. "P" denota o número de parâmetros. O refinamento das funções de

pertinência procura os parâmetros "α"$_{opt}$ ótimos de forma a que o erro de saída

$$E(\alpha) = \sum (\hat{y} - f(x_s : \alpha))^2 \qquad (4.7)$$

seja mínimo quando $\alpha_{opt} = a$, onde $\{ (x_s, y_s) \mid 1 \leq s \leq S \}$ é um conjunto de dados de treinamento.

Este é um problema típico de otimização não linear. Existem diversos métodos para resolve-lo. Um deles é o método da *regra delta* de descida do gradiente:

REPETIR ATÉ o equilíbrio
{ para cada i ($1 \leq i \leq p$) FAÇA
{ $\Delta E_i = (E(\alpha + \varepsilon I_i) - E(\alpha)) / \varepsilon$ (4.8)
$\alpha_i \leftarrow \alpha_i - \eta \Delta E_i$ (4.9)
}
}

Onde ε é um número pequeno e positivo, I_i é um vetor unitário cuja i-ésima componente é "1" e η é um coeficiente que controla a taxa de aprendizagem.

Outro, é aproximadamente o procedimento de inferência nebulosa (4.6) com uma rede neuronal treinada por algoritmo backpropagation. Devido ao fato de que backpropagation é essencialmente um método de decréscimo para solução de problemas de otimização não linear, pode-se minimizar diretamente sem qualquer aproximação.

O algoritmo usa um método de cálculo com uso de derivadas parciais (MAEDA et al., 1991). A quantidade de computação necessária é, a grosso modo, uma constante multiplicada pelo procedimento de inferência nebulosa, ou seja, não é proporcional ao número de parâmetros a serem ajustados. Para minimizar (4.7) por esse método, a derivada parcial $\partial(E) / \partial(\alpha_i)$ deve ser calculada para todo α_i. No algoritmo proposto, $\partial(E) / \partial(\alpha_i)$ são calculadas retornando-se ("backpropagating") o erro de saída $(\hat{y} - y)$ através da Flip-net. O cálculo é uma aplicação direta da derivação usando a regra da

cadeia. Em caso de haverem pontos de mínimo e máximo não diferenciáveis, pode-se evitá-los por interpolação.

O procedimento de inferência nebulosa (4.1) deve ser representado por uma rede de fluxo computacional como, por exemplo, a Flip-net apresentada na figura 4.1. Nesse caso, tem-se duas regras nebulosas, como na tabela 4.1.

Tabela 4.1 - Regras esquematizadas na Flip-net da figura 4.1.

regra 1:	SE	x_1 é pequeno	E	x_2 é médio	ENTÃO y é médio
regra 2:	SE	x_1 é grande	E	x_2 é pequeno	ENTÃO y é grande

Os graus das regras são dados pelos mínimos das partes "SE".

W_1 = min (pequeno(x_1) , médio(x_2))

W_2 = min (grande(x_1) , pequeno(x_2))

A função de pertinência de saída é dada por:

$B(z)$ = max (W_1 . médio $y(z)$, W_2 . grande $y(z)$)

Onde, como dito antes, z é uma variável muda para a integral de defuzificação.

O valor de saída $\hat{y}$ da inferência nebulosa é, como vimos, dado por (4.5):

$$\hat{y} = \frac{\sum_{k=1}^{K} B(z_k).\, z_k}{\sum_{k=1}^{K} B(z_k)} \qquad (4.5)$$

Sendo que z_k representa os valores discretizados de z. Agora devemos tomar os coeficientes dados pelas derivadas parciais

$$\partial(E) / \partial(On_j) = \sum_{\cdot} (\hat{y} - y) . \partial(y) / \partial(On_j) \qquad (4.10)$$

para todos os valores de saída On_j dos nós da rede "backpropagando-se" o erro total de saída ($\hat{y} - y$) através da rede pois os parâmetros α das funções de pertinência encontram-se nos nós n_1 - n_9. $\partial(E) / \partial(\alpha)$ pode ser calculado como

$$\partial(E) / \partial(\alpha) = \partial(E) / \partial(On_j) . \partial(On_j) / \partial(\alpha) \qquad (4.11)$$

onde o nó n_j inclui o parâmetro α.

O método proposto, na verdade, é o mesmo do método de descida do gradiente definido pelas equações (4.8) e (4.9) exceto que ΔE, na equação (4.8) é substituído na equação (4.11) por $\partial(E)/\partial(\alpha)$.

4.3 - Anfis

4.3.1 - Redes adaptativas

Uma rede adaptativa qualquer como, por exemplo, a da figura 4.2 (JANG, 1992), é, de uma forma geral, uma rede multinível na qual cada unidade realiza uma determinada função sobre os sinais que recebe de acordo com os diversos parâmetros que lhe dizem respeito. Suponhamos, sem perda de generalidade, que uma determinada rede adaptativa possua L níveis e que o k-ésimo nível possua *k* unidades. Podemos denotar a unidade da i-ésima posição deste k-ésimo nível por U_i^k e sua saída como O_i^k. Já que a saída de tal unidade depende das entradas que receber e de seus parâmetros intrinsecos, podemos escrever que

$$O_i^k = f(U_i^k) = f(O_1^{k-1} , \ldots , O_k^{k-1} , a , b, c, \ldots)$$

onde a, b, c, ... são os parâmetros característicos da unidade em questão.

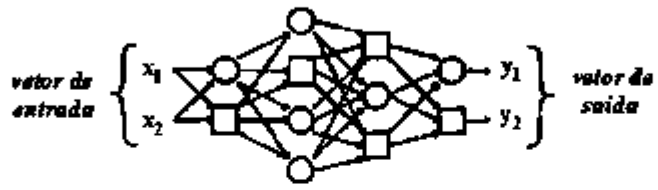


Figura 4.2 - Esquema de uma rede adaptativa.

Considerando-se que o conjunto de dados de treinamento tenha P entradas, podemos definir um erro associado à p-ésima ($1 \leq p \leq P$) entrada como:

$$Ep = \sum_{m=1}^{i} \left(T_{m,p} - O^{l}_{m,p} \right)^2 \tag{4.12}$$

onde $T_{m,p}$ é a m-ésima componente do p-ésimo vetor de saída desejado e $O^{l}_{m,p}$ é a m-ésima componente do vetor de saída real produzido pela rede à apresentação do p-ésimo vetor de entrada. Em vista disso, podemos considerar o erro sobre todos os pontos como:

$$E = \sum_{p=1}^{P} Ep \tag{4.13}$$

Da mesma forma que na Flip-Net, devemos desenvolver um procedimento de cálculo para implementar um método de descida do gradiente sobre o espaço de parâmetros das unidades da rede. Para uma unidade qualquer interna i do nível k, a *taxa de erro* pode ser deduzida a partir da regra da cadeia:

$$\frac{\partial Ep}{\partial o^{l}_{i,p}} = \sum_{m=1}^{(k+1)} \frac{\partial Ep}{\partial o^{k+1}_{m,p}} \frac{\partial o^{k+1}_{m,p}}{\partial o^{k}_{i,p}} \tag{4.14}$$

onde $1 \leq k \leq L$ -1. Pode-se ver que a taxa de erro de uma unidade interna pode ser expressa como uma combinação linear das taxas de erro das unidades dos níveis

seguintes. Em primeiro lugar, devemos calcular a taxa de erro $\frac{\partial Ep}{\partial O}$ para o p-ésimo dado de treinamento e para cada unidade de saída O. Essa derivada parcial é expressa (a partir de (4.12)) por

$$\frac{\partial Ep}{\partial O'_{i,p}} = -2\left(T_{i,p} - O'_{i,p}\right) \tag{4.15}$$

Agora, sendo α um parâmetro da rede adaptativa a equação (4.14) modifica-se para

$$\frac{\partial Ep}{\partial \alpha} = \sum_{O' \in S} \frac{\partial Ep}{\partial O'} \frac{\partial O'}{\partial \alpha} \tag{4.16}$$

sendo S o conjunto de unidades cujas saídas dependem de α. Dessa forma, a derivada do erro total E com respeito ao parâmetro α é:

$$\frac{\partial E}{\partial \alpha} = \sum_{p=1}^{p} \frac{\partial Ep}{\partial \alpha} \tag{4.17}$$

O acréscimo (ou decréscimo) ao valor de α, $\Delta\alpha$, pode ser definido como

$$\Delta\alpha = -\eta \frac{\partial E}{\partial \alpha} \tag{4.18}$$

com η , a *taxa de aprendizagem*, podendo ser expressa por

$$\eta = \frac{k}{\sqrt{\sum_{\alpha} \left(\frac{\partial E}{\partial \alpha}\right)^2}} \tag{4.19}$$

onde k é o tamanho do *passo de aprendizagem*. Normalmente, altera-se k para variar a *velocidade de convergência*. Nesta tese, ver-se-á que o valor de k, muitas vezes, é dependente do problema sob análise.

A identificação dos parâmetros pode ser feita, de forma mais robusta, combinando-se o método do gradiente com o uso de uma estimativa de mínimos quadrados (EsMQ).

Por simplicidade, sem perdermos qualquer generalidade, suponhamos que nossa rede adaptativa possua apenas uma saída tal que

$$\text{saída} = F(\mathbf{I}, P) \tag{4.20}$$

onde **I** é o vetor de variáveis de entrada e P é o conjunto de parâmetros da rede. Se existe uma função H tal que a função composta $H \circ F$ seja linear em alguns elementos de P, então esses elementos podem ser identificados pelo método de estimativa de mínimos quadrados.

Formalmente, o conjunto P pode ser decomposto em dois conjuntos tais que

$$P = P_1 \oplus P_2 \tag{4.21}$$

(onde $\oplus$ representa uma soma direta) tal que $H \circ F$ seja linear nos elementos de P_2. Aplicando-se H sobre a equação (4.20) temos que

$$H(\text{saída}) = H \circ F(\mathbf{I}, P) \tag{4.22}$$

que é linear sobre os elementos de S_2. Arbitrando-se valores para os elementos de S_1 podemos passar N pontos de dados pela equação (4.22) e obter a equação matricial

$$\mathbf{AX} = \mathbf{B} \tag{4.23}$$

onde **X** é um vetor desconhecido de elementos que são os parâmetros procurados em P_2. Sendo $|P_2| = M$, (número de parâmetros lineares) as dimensões de **A** , **X** e **B** são, respectivamente, N x M , M x 1 e N x 1.

Normalmente o número de pares de treinamento é maior do que M o que leva a um problema sobredeterminado para o qual, em geral, não existe uma solução exata para (4.23). Contudo, a estimativa de mínimos quadrados de **X** , **X*** pode minimizar o erro médio $\| \mathbf{AX} - \mathbf{B} \|^2$.

A fórmula mais conhecida para o cálculo de **X*** faz uso da pseudo-inversa de **A**:

$$\mathbf{X}^* = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{B} \qquad (4.24)$$

onde:

$\mathbf{A}^T$ é a transposta de **A**, como é normal

$(\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T$ é a pseudo-inversa de **A** , **A** sendo não-singular.

Como o cálculo da pseudo-inversa é alto consumidor de recursos computacionais e torna-se mal definido se $\mathbf{A}^T\mathbf{A}$ for singular, pode-se usar (e o Anfis usa) fórmulas seqüenciais para o cálculo da estimativa dos mínimos quadrados de **X**. Especificamente, seja a_i^T o i-ésimo vetor linha da matriz A e b_i^T o i-ésimo elemento de **B**. Então, **X** pode ser calculado iterativamente pelo uso das fórmulas seqüenciais

$$X_{i+1} = X_i + P_{i+1} a_{i+1} (b^T_{i+1} - a^T_{i+1} X_i) \qquad (4.25a)$$

$$P_{i+1} = Pi - \frac{P_i a_{i+1} a_{i+1}^T P_i}{1 + a_{i+1}^T P_i a_{i+1}}, \quad i = 0,1, \ldots, P\text{-}1 \qquad (4.25b)$$

onde P_i é a chamada *matriz de covariância*. A estimativa dos mínimos quadrados **X*** é igual a $\mathbf{X}_P$. As equações (4.25) são aplicadas, em engenharia, no *"Filtro de Kalman"*.

Com o método estabelecido acima (JANG, 1992), pode-se ajustar os parâmetros de uma rede adaptativa pelo uso combinado das técnicas de descida do gradiente e estimativa de mínimos quadrados. Cada época desse aprendizado híbrido é composto de um passo à frente e outro atrás. No passo à frente, apresentam-se dados à rede e os sinais passeiam pelas unidades até que as matrizes **A** e **B** da equação (4.23) sejam obtidas e os parâmetros em P_2 sejam identificados por EsMQ. No passo atrás, com o cálculo do erro

entre a saída fornecida pela rede, $\hat{y}$ e a saída desejada, y, as taxas de erro são propagadas para trás e os parâmetros P_2 são atualizados pelo método do gradiente.

4.3.2 - O Anfis

Anfis é o acrônimo para Adaptive-Network-Based Fuzzy Inference System (JANG, 1992). É um tipo de rede adaptativa que é funcionalmente equivalente a um sistema de inferência nebulosa.

Consideremos, novamente sem perda de generalidade, que tenhamos um sistema com duas entradas e uma saída e uma base de regras de Takagi-Sugeno de ordem um, equivalente a um sistema de inferência nebulosa de tipo 3 composta de duas regras como

R 1: SE x_1 é A1 E x_2 é B1 ENTÃO y é $p_1x_1 + q_1x_2 + r_1$
R 2: SE x_1 é A2 E x_2 e B2 ENTÃO y é $p_2x_1 + q_2x_2 + r_2$

A inferência nebulosa de tipo 3 proposta e a arquitetura Anfis equivalente estão esquematizadas na figura 4.3 (JANG, 1992). Cada unidade i no nível 1 possui uma função

$$O_i^1 = \mu_{A_i}(x) \tag{4.26}$$

onde x é a entrada aplicada à unidade i e A_i é a variável lingüística associada a essa unidade (pequeno, grande, etc.). No Anfis, aplica-se uma parametrização de curva em forma de sino para o cálculo da pertinência. No caso de uma gaussiana, os parâmetros, chamados de *parâmetros das premissas*, da unidade i seriam a_i, b_i e c_i, que nos dão a curva (JANG, 1992)

$$\mu_{A_i}(x) = exp\left\{-\left[\left(\frac{x - c_i}{a_i}\right)^2\right]^{b_i}\right\} \tag{4.27}$$

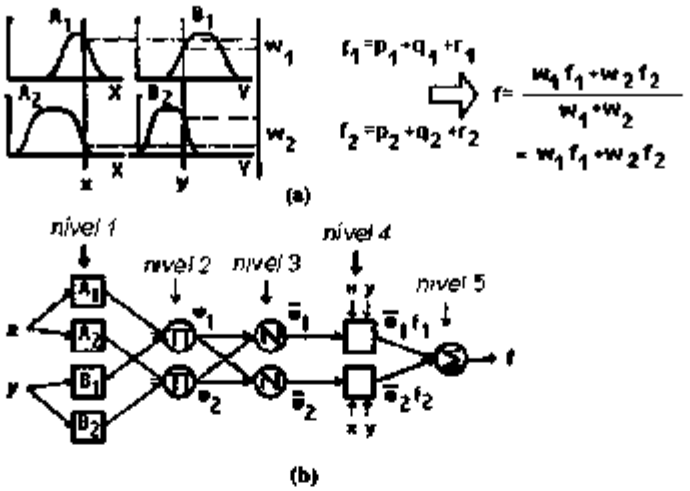


Figura 4.3 - (a) Inferência nebulosa de tipo 3. (b) Modelo ANFIS equivalente.

O nível 2 faz uma combinação das premissas das variáveis x_1 e x_2 de tal forma que

$$w_i = \mu_{A_i}(x_1) \times \mu_{B_i}(x_2) \quad i = 1,2 \tag{4.28}$$

As saídas do nível 3 são denominadas *valores de ativação normalizados*. Cada i-ésima unidade do nível 3 calcula o valor da ativação da i-ésima regra como sendo a razão entre a força de ativação da regra e a soma das forças de ativação de todas as regras, ou

$$\bar{w}_i = \frac{w_i}{w_1 + w_2} \quad , i = 1,2 \tag{4.29}$$

Cada unidade no nível 4 perfaz a função

$$O_i^4 = \bar{w}_i f_i = \bar{w}_i (p_i x_1 + q_i x_2 + r_i) \tag{4.30}$$

com $\bar{w}_i$ dado como sendo a saída do nível 3 e $\{ p_i , q_i$ e $r_i \}$ é o conjunto de parâmetros da unidade i . Esses parâmetros são denominados *parâmetros consequentes*.

A unidade solitária no nível 5 calcula a saída geral da rede como uma soma ponderada dos sinais que recebe, ou seja:

$$\hat{y} = O_i^5 = \sum_i \bar{w}_i f_i = \frac{\sum_i w_i f_i}{\sum_i w_i} \tag{4.31}$$

A figura 4.4 (JANG, 1992) apresenta a topologia Anfis equivalentes a sistemas nebulosos com inferências tipo 1.

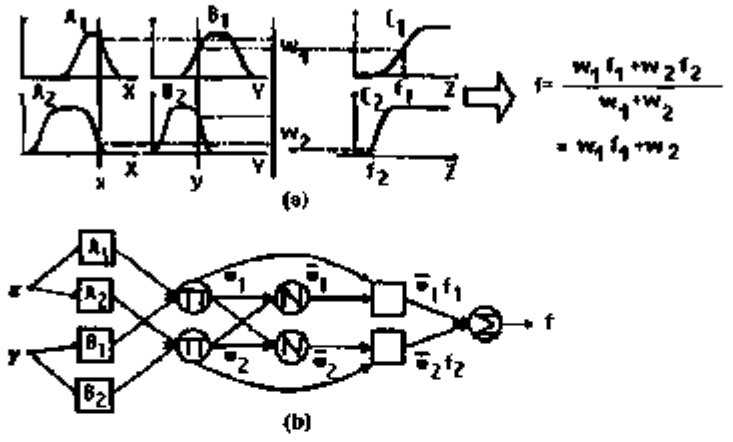


Figura 4.4 - (a) Inferência nebulosa de tipo 1. (b) Arquitetura ANFIS equivalente.

4.3.3 - Regras nebulosas simplificadas

Como vimos no capítulo 3, existe um tipo de regras nebulosas mais simples conhecido como Sistema de Takagi-Sugeno de ordem zero. Uma regra desse tipo é

$$\underline{\text{SE}}\ \ x_1 \text{ é } A_1\ \ \underline{\text{E}}\ \ x_2 \text{ é } B_2\ \ \underline{\text{ENTÃO}}\ \ y = \theta_1$$

sendo θ_1 um determinado valor fixo. Com este sistema de regras mais simples, pode-se provar que, obedecidas certas circunstâncias, o sistema de inferência nebulosa resultante possui uma capacidade ilimitada de aproximar qualquer função não linear definida sobre um suporte compacto. Seja o seguinte teorema, conhecido como *Teorema de Stone-Weierstrass*:

♦*Seja o domínio D um espaço compacto N-dimensional e seja F um conjunto de funções reais contínuas em D que satisfaçam aos seguintes critérios*

1) *Função identidade* $f(x) = 1$ *está definida em F*

2) *Separabilidade* *Para quaisquer dois pontos* $x1 \neq x2$ *em D, existe uma função f em F tal que* $f(x1) \neq f(x2)$

3) *Dependência algébrica* - *Se f e g são quaisquer duas funções em F, então fg e af + bg estão em F para quaisquer dois números reais a e b*

Então F é densa em C_D, *o conjunto de funções contínuas reais em D Ou seja, para qualquer* $\varepsilon > 0$ *e qualquer função g em* C_D, *existe uma função f em F tal que* $|g(x) - f(x)| < \varepsilon$ *para todo* $x \in D$

♦

Nos sistemas de inferência nebulosa, os domínios são sempre compactos, mesmo para um grande domínio definido para as variáveis de entrada. Isso é uma decorrência dos teoremas da análise real, onde qualquer domínio fechado e limitado em R^N é considerado compacto. Esse fato satisfaz à condição geral proposta no cabeçalho do teorema. O teorema também exige que um sistema nebuloso seja capaz de calcular a função identidade $f(x) = 1$. Como faz parte do pressuposto de um sistema nebuloso TSK de ordem zero que a parte conseqüente de uma regra seja igual a 1, mesmo um sistema de uma única regra satisfaz a tal condição.

A condição de separabilidade é satisfeita em qualquer sistema de inferência nebulosa que possa calcular com funções estritamente monotônicas em cada variável de entrada. O ajuste da funções de pertinência na parte de premissas das regras possibilita tal fato. A dependência algébrica necessária para a satisfação do teorema de Stone-Weierstrass exige que um sistema nebuloso seja capaz de aproximar somas e produtos de funções. Pela escolha de uma classe apropriada de funções de pertinência, um sistema nebuloso TSK de ordem zero é capaz de realizar tais aproximações (JANG,

1992). Assim, o Anfis como uma representação em rede de tal sistema satisfaz todos os critérios do teorema de Stone-Weierstrass.

O sistema desenvolvido nesta tese satisfaz também às condições de Stone-Weierstrass sendo baseado principalmente em sistemas TSK de ordem zero.

A equivalência funcional entre um sistema de inferência nebulosa e uma RBR pode ser estabelecida se algumas condições forem satisfeitas (JANG, 1992). Especificamente, tais condições seriam:

i - O número de unidades receptivas for igual ao número de regras.

ii - A saída de cada regra for composta de uma constante.

iii- As funções de pertinência de cada regra forem gaussianas de mesma variância.

iv- O operador T-norma usado para cômputo da ativação das regras for *produto*.

v- Tanto a RBR como o sistemas nebuloso usarem o mesmo método para calcular a saída.

Caso essas considerações sejam obedecidas, aperfeiçoamentos feitos tanto nas RBR's como no Anfis podem ser aproveitados em ambos. Esta equivalência Anfis-RBR's foi um avanço em relação à Flip-Net descrita anteriormente. O Anfis é mais consistente, mais robusto e é capaz de um auto-ajuste mais completo do que a Flip-Net.

4.3.4 - Exemplo

Usou-se o ANFIS para modelar a função *sinc*, definida como

$$z = sinc(x, y) = \frac{sen(x)}{x} \cdot \frac{sen(y)}{y} \qquad (4.32)$$

usando-se como dados de treinamento 121 pontos em grade regular nos intervalos $x = y = [\,-10\,,\,10\,]$. A arquitetura ANFIS utilizada consta de 16 regras, tendo-se 4 funções de

pertinência designadas para cada variável de entrada. Adicionalmente, foi necessário ajustar-se 72 parâmetros: 24 na parte das premissas das regras e 48 nas partes conseqüentes. A figura 4.5 (JANG, 1992) mostra em (a) e (b) o particionamento inicial do espaço de entrada para as variáveis x e y, respectivamente. Em (c) e (d) vemos as formas finais das funções de pertinência ao fim do treinamento.

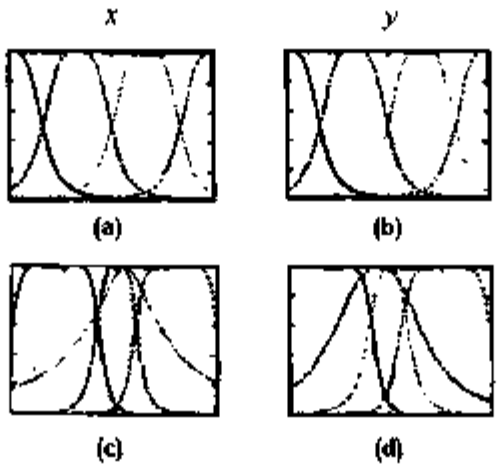


Figura 4.5 - Funções de pertinência (a) e (b) Iniciais. (c) e (d) Finais.

4.4 - Ajuste das funções de pertinência de saída

O método de descida do gradiente pode ser usado com eficiência no ajuste das posições das funções de saída de um modelo neuro-nebuloso (PASSINO et al., 1998). Normalmente esse ajuste é realizado em modelos neuro-nebulosos que usam inferência de tipo 2. Nesses casos, os métodos de correlação de mínimos ou de produtos fazem uso de funções contínuas explícitas para fuzificar a variável de saída y.

Consideremos que a saída y obedeça à relação (4.20). Usando-se a descida do gradiente para ajustar as posições dessas funções de saída, procuramos minimizar a relação (3.3) segundo os passos definidos pelas equações (4.12) a (4.19) onde os parâmetros α_i são os centros das funções de pertinência de saída.

Esse método apresenta a vantagem de realmente "sintonizar" as formas das funções de pertinência de saída de forma a diminuírem o critério de erro definido por

(3.3). Contudo, ele apresenta muitas desvantagens que depõem contra sua utilização. A primeira e mais evidente delas é o uso dos métodos de inferência nebulosa de tipo 2, grande consumidor de recursos computacionais, inviável de se usar em sistemas de controle em tempo real.

Uma outra desvantagem importante é que, ao se ajustar as partes conseqüentes das regras sem ajuste das antecedentes, corre-se o risco de, ao final, ter-se uma base de regras semelhante à da figura 3.13(a) com funções de pertinência conseqüentes inadequadas como as das figuras 3.12(a) e (b). Um caso ainda pior ocorre quando as regras são inconsistentes. Tal fato levaria a uma "paralisia" do treinamento.

A alternativa poderia ser o ajuste simultâneo das partes antecedentes e conseqüentes das regras. Em bases de regras simples isso pode ser eficaz. Em bases de regras mais complexas corre-se o risco do modelo nunca atingir a convergência por causa das possíveis oscilações que o modelo vai sofrer em torno de vários pontos de mínimos locais percorridos ora pelo ajuste da parte antecedente, ora pelo ajuste da parte conseqüente

4.5- Considerações

Podemos estabelecer equivalências consistentes entre as topologias das redes neuronais com sistemas nebulosos. Essas equivalências nos levam a sistemas neuro-nebulosos que podem mapear eficientemente sistemas físicos e matemáticos. Esse mapeamento pode ser aperfeiçoado pelo uso de algoritmos de aprendizagem que refinem os parâmetros das funções de pertinência.

A Flip-Net usa desse refinamento mas é um modelo muito pouco flexível pois não só sua topologia deve ser definida em tempo de implementação como seus auto-ajustes são limitados, pois ela não gera as suas próprias regras, apenas refina as definidas por um especialista. Assim, é incapaz de superar eventuais regras erradas ou conflitantes. Uma outra limitação importante é o uso da inferência nebulosa de tipo 2. O cálculo do centróide é grande consumidor de recursos computacionais, o que limita

consideravelmente sua aplicabilidade a sistemas de tempo real.

O Anfis foi relativamente bem sucedido tanto como aproximador universal, na modelagem de funções não-lineares, como na identificação de parâmetros para sistemas de controle. Ele modela as inferências nebulosas de tipo 1 e 3, o que é uma vantagem sobre a Flip-Net. Existem porém algumas desvantagens no Anfis. A mais evidente é que, após ajuste, os formatos das funções de pertinência levam a uma não partição da unidade entre elas, ou seja, elas são não normalizadas. Isso pode levar (e leva) a bases de regras completamente sem sentido lingüístico como, por exemplo, as das figuras 4.5 (a) e (b) que reproduzem o desenho original da base de regras obtida na modelagem de uma função de duas variáveis de entrada. Não é necessário qualquer julgamento mais rigoroso para estabelecer que tal base de regras é inadequada para efeitos interpretativos lingüisticamente. Uma outra desvantagem evidente do Anfis é sua estrutura fixa, incapaz de expansão e/ou contração.

O ajuste de funções de pertinência nas partes conseqüentes das regras geralmente leva a bases de regras extensas e complexas, que desejamos evitar.

No próximo capítulo aborda-se sistemas que foram desenvolvidos tendo em mente essas dificuldades e propõem formas alternativas de extração de regras para um sistema de inferência nebulosa.

5

Modelos expansíveis

5.1 - Um método de expansão de estrutura

Vários trabalhos, como os de Yager e Filev (1993, 1994) tem como foco a busca de métodos eficientes para a construção de modelos nebulosos. Baseando-se nesses autores, NAKOULA (1997) apresentou um método para a extração de regras lingüísticas para a construção de um sistema de inferência nebulosa com base de regras expansível. Nessa abordagem também existe uma preocupação de se extrair e manter, tanto quanto possível, as propriedades lingüísticas do modelo. O método inspira-se nos sistemas TSK de ordem zero que, como já vimos, possui todas as características necessárias para ser um aproximador universal. Em um sistema TSK de ordem zero a base de conhecimento é composta de regras do tipo

$$\underline{\text{SE}} \quad x_1 \text{ é } A_1 \quad \underline{\text{E}} \quad x_2 \text{ é } B_1 \quad \underline{\text{ENTÃO}} \quad \hat{y} = \theta_k \qquad (5.1)$$

Apesar de não usar diretamente os modelos TSK de ordem zero em seu trabalho, NAKOULA (1997) fez uso de um valor fixo θ_k como saída das regras e a ele associou um peso variável w_{ik}, função, como se deduz, dos conjuntos nebulosos de entrada e do valor de saída de cada regra.

Podemos, a grosso modo, dividir o problema de identificação de modelos em dois grandes grupos: sistemas de uma única entrada e uma única saída (SISO - Single Input, Single Output) e sistemas de múltiplas entradas e uma saída (MISO - Multiple Inputs, Single Output). Os sistemas de múltiplas entradas e múltiplas saídas (MIMO - Multiple Inputs, Multiple Outputs) podem, em geral ser tratados como um conjunto de sistemas MISO e não são especificamente tratados nesta tese.

5.1.1 - Uso do método em modelos SISO

Considere-se um conjunto de treinamento T onde cada elemento $t \in T$ é formado pelo par $(x,y)_t$ de tal forma que $y_t = f(x_t)$. O objetivo do método consiste na construção de um sistema de regras nebulosas que estime um valor $\hat{y}_t = \hat{f}(x_t)$ que minimize um dado critério de erro. O critério escolhido foi o do erro quadrático definido por

$$J = \frac{1}{N}\sum_{\forall t \in T} \xi_t(x,y)^2 = \frac{1}{N}\sum_{\forall t \in T}\left(\hat{y}_t - y_t\right)^2 \tag{5.2}$$

onde N é o número total de pares de treinamento e $(\hat{y}_t - y_t)$ é o chamado *resíduo de identificação* para cada ponto (EVSUKOFF et al., 2000).

Os dois desafios consistem em identificar tanto a estrutura como os parâmetros de um modelo nebuloso que, sem perder de vista a característica de interpretação lingüística de tais modelos, minimize o erro definido em (5.2) até um valor aceitável. A identificação da estrutura do modelo nebuloso visa a determinação (conforme as notações definidas no capítulo 2) do vetor de protótipos $a = [a_1, \ldots, a_n]^t$ que define a partição A(x). A identificação dos parâmetros consiste em calcular as componentes do vetor θ de saída das regras.

A estratégia de NAKOULA et al. (1997) é a de se fazer a identificação de um conjunto de pontos aprendidos $P \subset T$ onde cada elemento $(\alpha,\varphi)_t \in P$, $\varphi_t = f(\alpha_t)$ define uma regra do tipo da de um modelo TSK de ordem zero que, para sistemas de uma única entrada e uma única saída, pode ser escrita, à semelhança de (5.1) como

$$\underline{\text{SE}} \quad x \text{ é } A_t \quad \underline{\text{ENTÃO}} \quad \hat{y} = \theta_t \tag{5.3}$$

sendo $A_t = \alpha_t$ o protótipo da função de pertinência triangular que define o conjunto nebuloso A_t e $\theta_t = \varphi_t$ o valor da saída correspondente. Já que $(\alpha,\varphi)_t \in T$, a saída do sistema nebuloso tem erro nulo para os pontos aprendidos. Baseado nisso, quando $P = T$

(todos os pontos de treinamento forem aprendidos), o modelo terá erro nulo e sua convergência está, portanto, garantida.

O processo de identificação se inicia com a definição do domínio do problema. Definiu-se esse domínio como $\Omega = \{w_1, w_2\}$, sendo $w_1 = x_{min}$ e $w_2 = x_{max}$ dentro dos limites da variável. Quando de sua inicialização, o algoritmo atribui esses limites do domínio aos protótipos dos conjuntos nebulosos associados às duas primeiras regras do modelo como na figura 5.1 (EVSUKOFF et al., 2000) de modo que

$$\mathbf{a}\,(k=1) = [\,a_1\;\; a_2\,] = [\,w_1\;\; w_2\,] \tag{5.4}$$

Os pontos associados aos limites do domínio são então incluídos no conjunto de pontos aprendidos fazendo-se

$$P\,(k=1) = \{\,(\alpha,\varphi)_1\,;\,(\alpha,\varphi)_2\,\} \tag{5.5}$$

onde $\alpha_1 = w_1$ e $\alpha_2 = w_2$. Da mesma forma, os valores das saídas correspondentes são atribuídos como saída das regras de forma que

$$\theta\,(k=1) = [\,\varphi_1\,\varphi_2\,] = [\;f(w_1)\;f(w_2)\;] \tag{5.6}$$

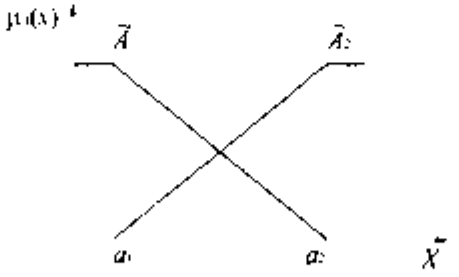


Figura 5.1 - Conjuntos nebulosos iniciais.

Dada uma tolerância especificada, enquanto o critério de erro (5.2) se mantiver acima dela, o algoritmo busca novos pontos de forma a incluí-los no seu conjunto de pontos aprendidos P. O processo torna-se, assim, iterativo. Em uma iteração genérica k

em que o sistema tenha p pontos aprendidos, o novo ponto $(\alpha,\varphi)_{p+1}$ a ser incluído é escolhido como sendo o de maior distância da aproximação segundo o critério

$$(\alpha,\varphi)_{p+1} = \arg\max_{\forall i \in I}\left(\xi_i(x,y)^2\right) \tag{5.7}$$

Esse novo ponto $(\alpha,\varphi)_{p+1}$ calculado em (5.7) e selecionado para ser aprendido, define uma nova regra do tipo (5.3) e redefine toda a partição dos protótipos em $\mathbf{A}$ como se observa na figura 5.2 (EVUSKOFF et al., 2000). O vetor de protótipos inclui o novo ponto aprendido segundo

$$\mathbf{a}(k) = \mathbf{a}(k\text{-}1) \cup \{\alpha_{p+1}\} \tag{5.8}$$

O vetor de parâmetros de saída das regras também é atualizado, nele incluindo-se o valor de saída correspondente ao novo ponto aprendido de forma que

$$\theta(k) = \theta(k\text{-}1) \cup \{\varphi_{p+1}\} \tag{5.9}$$

O erro total (5.2) é então calculado para o novo modelo. O processo iterativo continua até que esse erro seja menor que uma tolerância arbitrariamente definida.

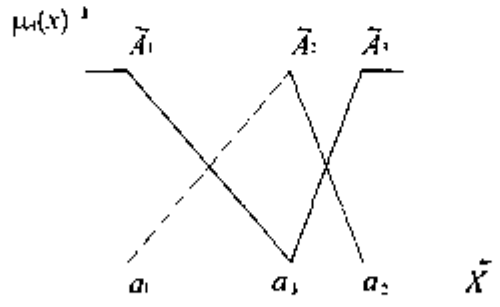


Figura 5.2 - Protótipos na primeira iteração.

O uso dos pesos na base de regras pode ser melhor compreendido a partir do fato de que o modelo escolhido pode ser descrito por uma base de regras lingüísticas ponderadas da forma

$$\underline{\text{SE}}\ \ x_1 \text{ é } A_i\ \underline{\text{E}}\ x_2 \text{ é } B_i\ \ \underline{\text{ENTÃO}}\ \ \hat{y} = \theta_k \quad \text{com peso } w_{ijk} \tag{5.10}$$

Os pesos w_{ijk} associados a cada regra são um número real no intervalo fechado [0 , 1]. Esse peso representa o grau de confiança (ou de validade) da regra. Pode-se rearranjar (5.10) na forma

$$\underline{\text{SE}}\quad x_1 \text{ é } A_i\quad \underline{\text{E}}\quad x_2 \text{ é } B_i\quad \underline{\text{ENTÃO}}\quad \hat{y} = w_{ijk} / \theta_k \tag{5.11}$$

Se utilizarmos uma representação triangular (fig.5.3) para as funções de pertinência, podemos calcular os pesos das regras segundo a fórmula

$$w_m = \frac{1}{X_m - X_{m+1}}\left(x_0 - X_m\right) + 1 \tag{5.12}$$

onde X_m e X_{m+1} são respectivamente os valores modais de C_m e C_{m+1}. Assim, todas as regras da base podem ser escritas como

$$\underline{\text{SE}}\quad x_1 \text{ é } A_i\quad \underline{\text{ENTÃO}}\quad \hat{y} = w_m/C_m + (1-w_m)/C_{m+1} \tag{5.13}$$

ou seja, a saída $\hat{y}$ é calculada como

$$\hat{y} = \frac{w_m X_m + \left(1 - w_m\right)X_{m+1}}{w_m + \left(1 - w_m\right)} \tag{5.14}$$

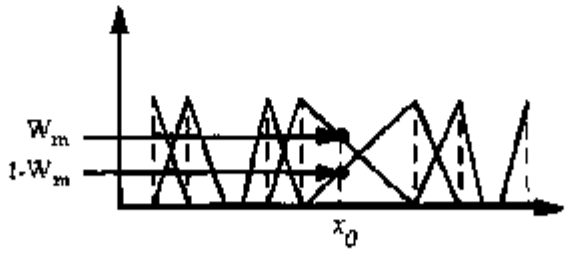


Figura 5.3 - Pesos das regras em funções de pertinência triangulares.

5.1.1.1- Exemplo 1

Baseados nisso, seja o conjunto de pontos de aprendizagem (x, y) plotados na figura 5.4 (NAKOULA, 1997). Afim de se determinar um modelo nebuloso que represente corretamente a relação $y = f(x)$, na fase de inicialização são definidos dois conjuntos nebulosos no espaço de entrada (A_1 e A_2) e outros dois sobre o espaço de saída (B_1 e B_2). Esta seria a estrutura mínima do modelo nebuloso. Os parâmetros a serem identificados são:

(i) α_1 e α_2, os valores modais atribuídos a A_1 e A_2.

(ii) φ_1 e φ_2, os valores modais atribuídos a B_1 e B_2.

(iii) os pesos w_{11}, w_{12}, w_{21}, w_{22} correspondentes às regras

$$\underline{\text{SE}} \quad x \text{ é } A_i \quad \underline{\text{ENTÃO}} \quad y \text{ é } B_j \quad \text{com pesos } w_{ij} : \quad i, j = 1,2 \qquad (5.15)$$

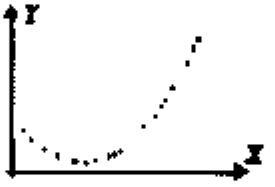


Figura 5.4 - Um exemplo de função $y = f(x)$.

De início, as estimativas são feitas como

$$\alpha_1 = x_{min}, \quad \alpha_2 = x_{max}, \quad \varphi_1 = y_{min}, \quad \varphi_2 = y_{max},$$

onde x_{min}, x_{max}, y_{min}, y_{max}, representam os valores mínimo e máximo das coordenadas x e y respectivamente, extraídos dos dados de aprendizagem. As funções de pertinência assim obtidas cobrem todo o domínio de entrada-saída do problema como se vê na figura 5.5 (NAKOULA, 1997). Note-se que a inferência usada no modelo é, na verdade, de tipo 2. Os pesos das regras são então calculados de forma a que o modelo nebuloso reproduza precisamente os pontos de aprendizagem de mínima e máxima abcissas. Desse cálculo, obtemos

$$w_{11} = 0.76 \; ; \; w_{12} = 0.24 \; ; \; w_{21} = 0 \text{ e } w_{22} = 1$$

ou seja, a base de regras inicial contém as seguintes regras:

$$\begin{aligned} &\text{R1: } \underline{\text{SE}} \quad x \text{ é } A_1 \quad \underline{\text{ENTÃO}} \quad y \text{ é } (0{,}76 / B_1 + 0{,}24 / B_2) \\ &\text{R2: } \underline{\text{SE}} \quad x \text{ é } A_2 \quad \underline{\text{ENTÃO}} \quad y \text{ é } 1/B_2. \end{aligned} \tag{5.16}$$

Pode-se observar que o modelo conseguido efetua uma interpolação linear entre os pontos aprendidos P_1 e P_2, interpolação essa de baixa qualidade para o sistema (curva $y = f(x)$) sob análise.

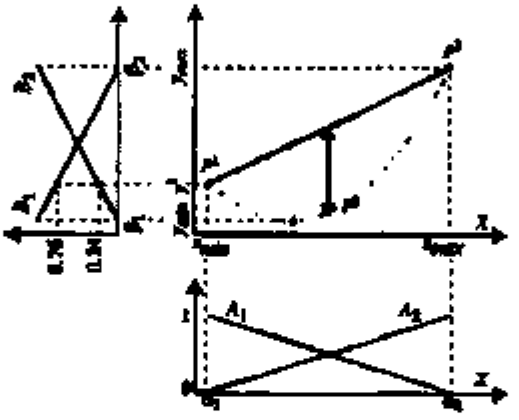


Figura 5.5 - Aproximação inicial de $y = f(x)$.

No primeiro passo da fase iterativa de refinamento, o ponto dos pares entrada-saída de aprendizagem que produz o maior erro segundo o critério (5.7), P_3, é escolhido para melhorar o modelo inicial. Dois novos valores lingüísticos A_3 e B_3 são criados sobre os espaços X e Y, respectivamente. Como resultado da aprendizagem de P_3, a estrutura do modelo cresce e um novo cálculo dos pesos das regras é feito de modo a acomodar esse novo vínculo, visualizado na figura 5.6 (NAKOULA, 1997).

Os valores modais de A_3 e B_3 são incluídos na base de regras, que sofre o acréscimo de uma regra e tem seus pesos recalculados. Assim, obtemos a base de regras

SE x é A_1 ENTÃO y é ($0.2/B_2 + 0.8/B_3$)

SE x é A_2 ENTÃO y é $1/B_2$

SE x é A_3 ENTÃO y é $1/B_3$ (5.17)

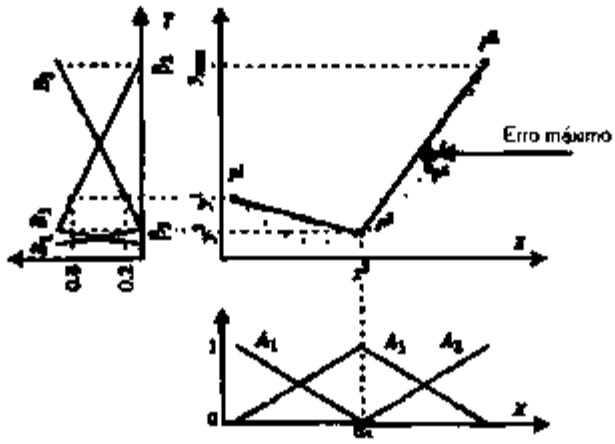


Figura 5.6 - Refinamento do modelo com o acréscimo de novo conjunto nebuloso.

Com efeito, a modificação na partição Y implicou em uma modificação na descrição das funções de pertinência nebulosas em X. A figura 5.6 ilustra as novas partições e o modelo correspondente obtido. Para os pontos aprendidos P_1, P_2 e P_3, o erro é nulo e o erro para todos os demais pontos é consideravelmente reduzido. Na próxima iteração, o ponto que produz o maior erro é P_4 na figura 5.7(a) (NAKOULA, 1997). Assim, ele é o escolhido para criar as variáveis A_4 e B_4. Esse acréscimo na estrutura do modelo é acompanhado pelo recálculo dos pesos das regras e a nova base de regras é

SE x é A_1 ENTÃO y é ($0,46/B_3 + 0,54/B_4$)

SE x é A_2 ENTÃO y é $1/B_2$

SE x é A_3 ENTÃO y é $1/B_3$

SE x é A_4 ENTÃO y é $1/B_4$ (5.18)

O modelo obtido e as partições nebulosas correspondentes a essa base de regras pode ser visto na figura 5.7(a) (NAKOULA, 1997). A figura 5.7(b) mostra o modelo obtido e as correspondentes partições nebulosas após a oitava iteração.

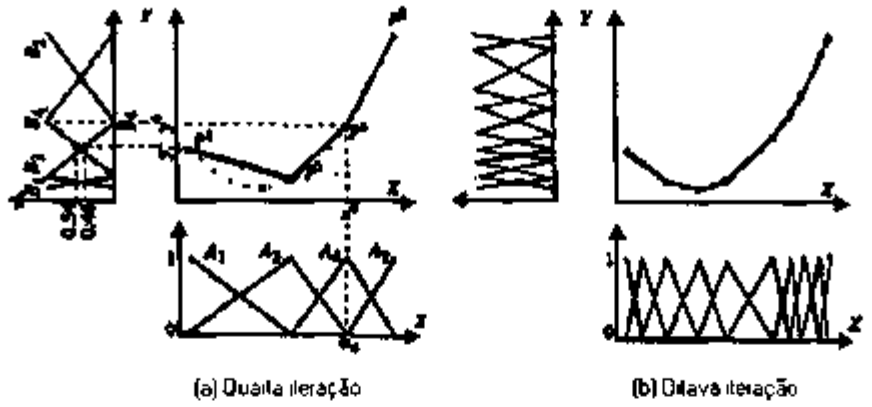


Figura 5.7 - Refinamentos sucessivos do modelo.

5.1.1.2 - Exemplo 2

Considere-se que tenhamos uma função $y = f(x)$ definida como na figura 5.8 (NAKOULA, 1997) onde:

$$\begin{aligned} y &= -9x^2 + 3x + 1/3 && x \in [0 \quad 1/3] \\ y &= 2x - 1/3 && x \in [1/3 \quad 1/2] \\ y &= 0{,}03/\left(0{,}03 + (4{,}5x - 3{,}85)^2\right) && x \in [1/2 \quad 1] \end{aligned} \tag{5.19}$$

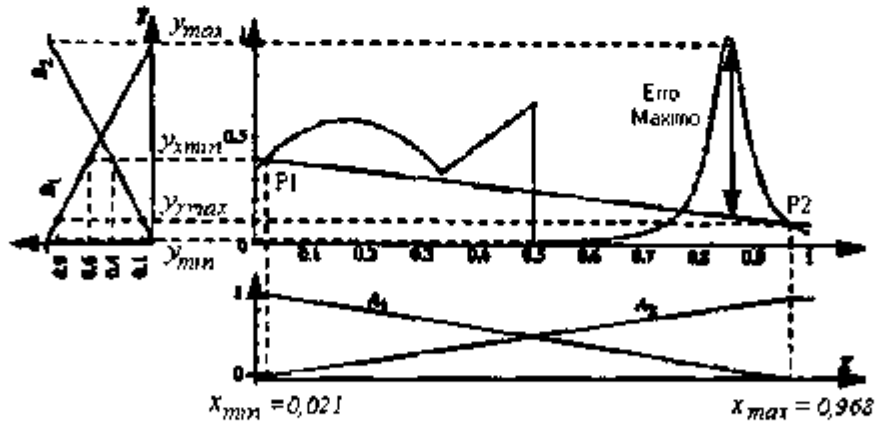


Figura 5.8 - Plotagem de $y = f(x)$ e partições de entrada e saída iniciais.

Gerando-se aleatoriamente 100 pontos de entrada-saída de treinamento na forma (x, $y = f(x)$), distribuidos no intervalo [0 , 1], pode-se avaliar o progresso do método.

Na fase de identificação, faz-se a extração dos valores extremos dos intervalos de entrada e saída. Segundo NAKOULA (1997), para seus dados de treinamento, x_{min} = 0.021 , x_{max} = 0,968 , y_{min} = 0.012 , y_{max} = 0,970. A esses valores, são associados os símbolos A_1 e A_2 (a x_{min} e x_{max} resp.) no espaço de entrada **X** e B_1 e B_2 (a y_{min} e y_{max} resp.) no espaço de saída **Y**. Esses símbolos formam as partições nebulosas iniciais, conforme mostra a figura 5.8. A derivação da base inicial de regras nebulosas do tipo de (5.16) nos dá

$$\begin{aligned} &\text{R1: } \underline{\text{SE}} \quad x \text{ é } A_1 \quad \underline{\text{ENTÃO}} \quad y \text{ é } (0{,}6/B_1 + 0{,}4/B_2) \\ &\text{R2: } \underline{\text{SE}} \quad x \text{ é } A_2 \quad \underline{\text{ENTÃO}} \quad y \text{ é } (0{,}9/B_1 + 0{,}1/B_2) \end{aligned} \tag{5.20}$$

com y_{min} = 0.393 e y_{max} = 0,105. Novamente, essa base com apenas duas regras leva a uma interpolação linear da função, interpolação essa sendo o segmento de reta entre P_1 e P_2 na figura 5.8. Na fase de refinamento, enquanto o critério de erro definido pela equação (5.2) se mantiver acima de uma tolerância especificada, novos protótipos são inseridos nos espaços X e Y nos pontos definidos segundo a equação (5.7). A figura 5.9 (NAKOULA, 1997) nos mostra 10 protótipos particionando o espaço **X** e a interpolação obtida para a curva (5.19). Afim de avaliar a capacidade de generalização do modelo obtido, NAKOULA (1997) gerou um novo conjunto de pontos aleatórios para teste. Com uma base de 20 regras (20 protótipos), o EMQ obtido foi de 0,0027 para o conjunto de dados de aprendizagem e de 0,0062 para o conjunto de dados de teste.

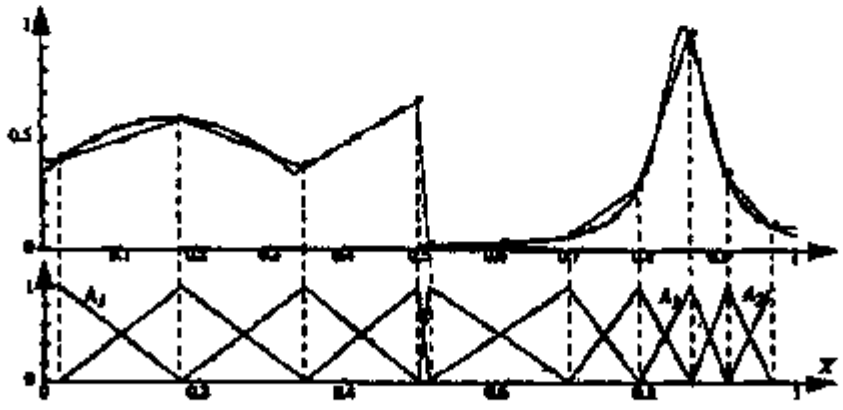


Figura 5.9 - Interpolação obtida com 10 protótipos.

5.1.2 - Uso do método em modelos MISO

A extensão do método para modelos MISO é conseguida considerando-se as variáveis de entrada como sendo as componentes de um vetor. Cada elemento $t \in T$ do conjunto de treinamento é composto do par $(\mathbf{x}, y)_t$, onde $\mathbf{x}_t = [x_{1t} \quad \cdots \quad x_{mt}] \in X_1 \times \ldots \times X_m$ e onde $y_t = f(\mathbf{x}_t)$. Da mesma maneira que no caso SISO, o algoritmo de identificação procura determinar os protótipos da partição nebulosa de cada variável X_i e o vetor θ de parâmetros de saída das regras. Considere-se, por exemplo, um modelo nebuloso com $m = 2$ variáveis de entrada; x_1 e x_2, onde essas variáveis são respectivamente descritas pelas partições nebulosas $\mathbf{A}_1 = \{A_{11}, A_{12}, A_{13}\}$ e $\mathbf{A}_2 = \{A_{21}, A_{22}, A_{23}\}$. As regras da base de regras são escritas como em (5.1):

$$\underline{\text{SE}} \quad x_1 \text{ é } A_{1i} \quad \underline{\text{E}} \quad x_2 \text{ é } A_{2i} \quad \underline{\text{ENTÃO}} \quad \hat{y} \text{ é } \theta_k \qquad (5.21)$$

As descrições nebulosas de cada entrada x_i são calculadas pela combinação dos termos nas premissas das regras que se expandem como os galhos de uma árvore (fig. 5.10). Como pode ser visto nessa figura, cada combinação de termos na premissa das regras pode ser vista como a definição de um novo termo do conjunto descritor $\mathbf{C} = \{C_1 \cdots C_r\}$. O resultado da fuzificação de x_i é a descrição $\tilde{x}_i$. No caso, para este exemplo:

$$\tilde{\mathbf{x}}_t = [\mu_{\tilde{x}}(C_1) \ldots \mu_{\tilde{x}}(C_4)] \qquad (5.22)$$

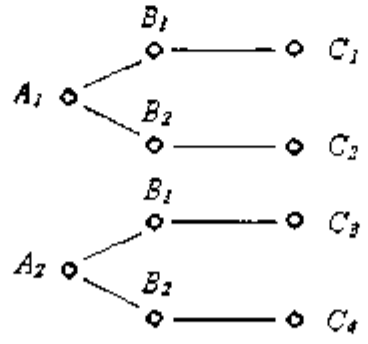


Figura 5.10 - Combinação de premissas que levam ao conjunto descritor **C**.

Cada componente da descrição $\tilde{x}_i$ é calculada a partir das descrições $\tilde{x}_{1i}$ e $\tilde{x}_{2i}$ por um operador t-norma (EVSUKOFF et al., 2000) r como:

$$\mu_{\tilde{x}_i}(C_i) = \mathrm{t}\left(\mu_{\tilde{x}_{1i}}(A_{1i}) \,.\, \mu_{\tilde{x}_{2i}}(A_{2i}) \right) \tag{5.23}$$

Aqui, usa-se o operador produto como operador t-norma, e as componentes da descrição e as componentes da descrição nebulosa $\tilde{x}_i$ são calculadas sendo:

$$\mu_{\tilde{x}_i}(C_i) = \left(\mu_{\tilde{x}_{1i}}(A_{1i}) \,.\, \mu_{\tilde{x}_{2i}}(A_{2i}) \right) \tag{5.24}$$

Claro que a equação (5.23) pode ser estendida a qualquer número de variáveis de entrada. Dessa forma, a base de dados para um modelo MISO nebuloso de qualquer dimensionalidade pode ser resumidamente escrita a partir do vetor de entrada $\mathbf{x}$ como

$$\underline{\text{SE}} \quad \mathbf{x} \text{ é } \mathrm{C}_i \quad \underline{\text{ENTÃO}} \quad \hat{y} = \theta_i \tag{5.25}$$

No caso, se a cardinalidade de $\mathbf{A}_1$ é n_1 e a cardinalidade de $\mathbf{A}_2$ é n_2, então a cardinalidade de $\mathbf{C}$ é M, onde $M = n_1 n_2$ é o tamanho da descrição $\tilde{\mathbf{x}}_i$. Quando são usadas muitas variáveis, isso leva à explosão combinatória conhecida como *"maldição da dimensionalidade"*. A saída do modelo nebuloso MISO descrito por regras do tipo de (5.25) é análoga ao do modelo SISO TSK descrito por regras do tipo de (5.3) e pode ser calculada por (EVSUKOFF et al., 2000):

$$\hat{y}_i = \boldsymbol{\pi}_i \,.\, \theta \tag{5.26}$$

onde $\theta = [\theta_1 \quad \ldots \quad \theta_M]^T$ é o vetor de parâmetros cujas componentes são as saídas das regras TSK como (5.25). Todas as combinações de termos nas premissas devem ser consideradas de tal forma que o modelo seja completo, *i.e.* que produza uma saída para qualquer valor admissível de entrada. Sem perda de generalidade, considere-se a identificação de um modelo de duas variáveis. O domínio do modelo é o

retângulo $\Omega = \{\omega_1 \ldots \omega_4\}$ no plano $V_1 \times V_2$, definido pelos limites de cada variável mostrado na figura 5.10 (EVSUKOFF et al . 2000):

$$\omega_1 = \begin{bmatrix} x_{1min} & x_{2min} \end{bmatrix} \quad \omega_2 = \begin{bmatrix} x_{1min} & x_{2max} \end{bmatrix}$$
$$\omega_3 = \begin{bmatrix} x_{1max} & x_{2min} \end{bmatrix} \quad \omega_4 = \begin{bmatrix} x_{1max} & x_{2max} \end{bmatrix} \tag{5.27}$$

Da mesma maneira que no caso SISO, na fase de inicialização (iteração $\kappa = 0$) os limites de cada variável definem os protótipos de dois conjuntos nebulosos (fig. 5.11):

$$\mathbf{a}_1 (k = 0) = [x_{1min} \quad x_{1max}]^t$$
$$\mathbf{a}_2 (k = 0) = [x_{2min} \quad x_{2max}]^t \tag{5.28}$$

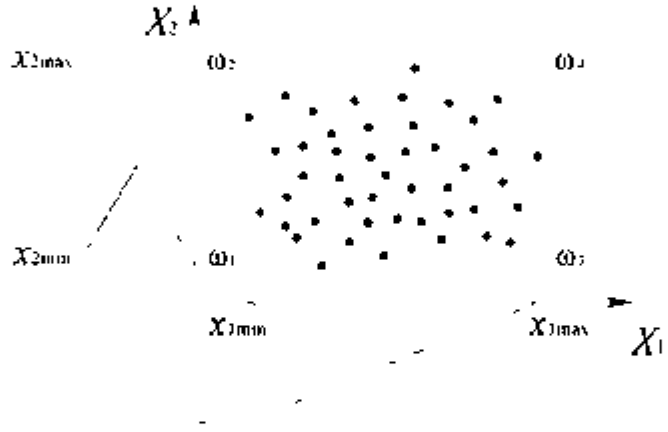


Figura 5.11 - Limites do domínio no caso 2D.

Na fase de inicialização, uma base de regras é obtida pela combinação dos termos na premissa da regras como em (5.1). Os pontos definidos pela combinação dos protótipos dos termos correspondentes são os limites do domínio $\omega_1 \ldots \omega_4$. Entretanto, esses pontos podem não possuir correspondentes no conjunto de treinamento T (como é o caso da figura 5.11) e o conjunto de pontos aprendidos pode continuar vazio após a primeira iteração. Nesse caso, a determinação do vetor de parâmetros $\theta(\kappa = 0)$ das primeiras quatro regras não pode ser feita do mesmo modo que no caso SISO (EVSUKOFF et al., 2000).

Para modelos MISO, o problema do preenchimento do vetor de saída das regras

também ocorre quando um novo conjunto nebuloso é incluído na partição nebulosa de uma dada variável de entrada. Por exemplo, considere-se a primeira iteração do procedimento de identificação e o conjunto de pontos aprendidos $P = \{(\alpha, \varphi)_l\}$, onde $\alpha_1 = [a_{12} \quad a_{22}]$ e a_{12} e a_{22} são os protótipos dos termos A_{12} e A_{22} que tenham acabado de ser incluídos nos conjuntos descritores $\mathbf{A}_1 = \{A_{11}, A_{12}, A_{13}\}$ e $\mathbf{A}_2 = \{A_{21}, A_{22}, A_{23}\}$ como mostrado na figura 5.12 (EVSUKOFF et al., 2000). O ponto $\alpha_1 = [a_{12} \quad a_{22}]$ é o único cuja saída $\varphi_1 = f(\alpha_1)$ está fixa no conjunto de pontos aprendidos. Todos os pontos definidos pela combinação dos protótipos dos termos das premissas das regras não possuem correspondentes no conjunto de pontos aprendidos. Neste exemplo, a saída $\varphi_1 = f(\alpha_1)$ é tomada como a componente θ_5 do vetor θ. As outras componentes do vetor devem ser calculadas (EVSUKOFF et al., 2000).

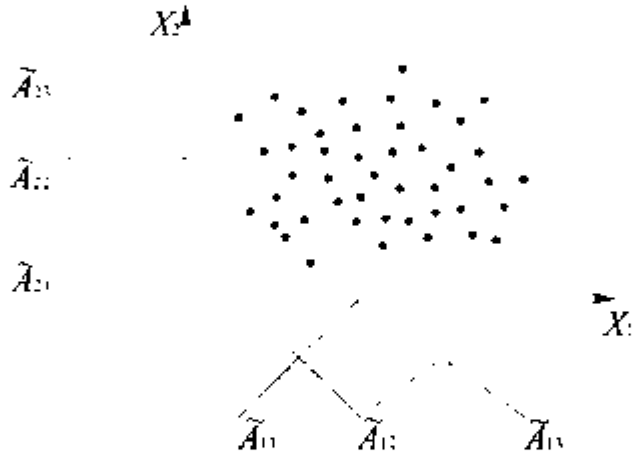


Figura 5.12- Novos conjuntos nebulosos no caso 2D.

NAKOULA et al. (1997) propõe que o preenchimento da base de regras seja calculado a partir dos pontos mais próximos do conjunto de treinamento em relação aos que seriam formados pela combinação dos protótipos dos termos das premissas das regras. Essa abordagem permite que apenas os pontos do conjunto de treinamento sejam usados para o preenchimento da base de regras. Entretanto, na ausência de pontos de aprendizagem na vizinhança dos pontos formados pela combinação dos protótipos, uma estimativa muito pobre é feita para as saídas das regras correspondentes. EVSUKOFF et al. (2000) propôs uma melhoria em relação ao método onde os parâmetros da base de regras são calculados como a solução de um problema de otimização de mínimos

quadrados, como é descrito a seguir.

5.2 - Um método aperfeiçoado

Em um trabalho recente, EVSUKOFF et al.(2000) considerou que o cálculo dos parâmetros θ em (5.26) poderia ser realizado através de um algoritmo que fizesse a minimização da norma quadrática de um vetor que contenha os resíduos $(\hat{y}_t - y_t)$ na equação em (5.2).

Considere-se a matriz $\mathbf{W} \in I^{N \times M}$ que armazena a descrição de todos os pontos no conjunto de treinamento, ou seja:

$$\mathbf{W} = \begin{bmatrix} \mathbf{w}_1 \\ \vdots \\ \mathbf{w}_N \end{bmatrix} = \begin{bmatrix} \mu_{A_1}(B_1) & \cdots & \mu_{A_M}(B_M) \\ \vdots & \ddots & \vdots \\ \mu_{A_1}(B_1) & \cdots & \mu_{A_M}(B_M) \end{bmatrix} \tag{5.29}$$

onde M é o tamanho da combinação de todos os termos nas premissas das regras do modelo MISO. A matriz $\mathbf{W}$ é comumente chamada, na literatura, de *matriz dos regressores* (YEN et WANG, 1999). Segundo o que vimos anteriormente, de acordo com (5.26), os dados de treinamento podem ser escritos como:

$$\mathbf{y} = \mathbf{W}\theta + \xi \tag{5.30}$$

onde $\mathbf{y} = [y_1 \quad \cdots \quad y_N]^t$ é o vetor cujas componentes são a saída de cada dado $t \in T$ e $\xi = [\xi_1 \quad \cdots \quad \xi_N]^t$ é o *vetor de resíduos*, que representa os erros de modelagem, ruídos aditivos e demais incertezas (EVSUKOFF et al., 2000). Os parâmetros θ em (5.30) podem ser determinados pela minimização de uma norma do vetor de resíduos ξ. A minimização da norma quadrática do vetor de resíduos caracteriza o *problema do erro mínimo quadrado* (GOLUB et al., 1989, LAWSON et al., 1974):

$$\min_{\theta} \|\mathbf{W}\theta - \mathbf{y}\|_2 \tag{5.31}$$

Como colocado nesse trabalho, a norma quadrática permite uma solução numérica robusta do vetor de parâmetros θ. A solução de (5.31) é equivalente à minimização do critério de erro quadrático (5.2), que pode ser escrito em forma vetorial como (EVSUKOFF et al., 2000):

$$J = \frac{1}{N}[\mathbf{W}\theta - \mathbf{y}]^t[\mathbf{W}\theta - \mathbf{y}] = \frac{1}{N}\|\mathbf{W}\theta - \mathbf{y}\|_2^2 \tag{5.32}$$

No caso em que o número de dados de treinamento N é maior do que o número de parâmetros M ($N > M$), a solução do problema de otimização (5.31) é única se a matriz de regressores $\mathbf{W}$ tem rank pleno (EVSUKOFF et al., 2000) e a solução é calculada pelo sistema linear

$$\mathbf{W}^t\mathbf{W}\theta = \mathbf{W}^t\mathbf{y} \tag{5.33}$$

sendo $\mathbf{W}^t\mathbf{W}$ simétrica e positiva definida. Em problemas nos quais o rank de $\mathbf{W}$ é deficiente ($rank(\mathbf{W}) < M$), existe um número infinito de soluções para (5.31) e o problema torna-se mal condicionado. A deficiência no rank de $\mathbf{W}$ afeta a estabilidade numérica da solução e mesmo pequenas perturbações em $\mathbf{W}$ ou $\mathbf{y}$ podem causar grandes variações na solução de θ. Algoritmos robustos para a solução do problema de mínimos quadrados podem ser obtidos pela *decomposição em valores singulares* (DVS) da matriz de regressores (EVSUKOFF et al., 2000).

5.2.1 - DVS - Decomposição em Valores Singulares

A DVS (YEN et WANG, 1999) é uma fatoração da matriz $\mathbf{W}$ em três matrizes na forma:

$$\mathbf{W} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^t \tag{5.34}$$

onde $\mathbf{U} \in R^{N \times N}$ e $\mathbf{V} \in R^{M \times M}$ são matrizes ortogonais e $\boldsymbol{\Sigma} = \text{diag}(\sigma_1, \sigma_2, \ldots, \sigma_M) \in R^{N \times M}$ é

uma matriz diagonal onde $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_M \geq 0$ são os valores singulares de **W**. A DVS é uma decomposição adequada pois permite calcular uma solução para (5.31) com norma quadrática mínima para problemas nos quais o rank é deficiente e ainda informa o número de condição da matriz **W** (EVSUKOFF et al., 2000). Escrevendo-se $\mathbf{U} = [\mathbf{u}_1 \quad \ldots \quad \mathbf{u}_N]$ e $\mathbf{V} = [\mathbf{v}_1 \quad \ldots \quad \mathbf{v}_M]$, onde $\mathbf{u}_i$ e $\mathbf{v}_i$ são as colunas de **U** e **V**, a solução de (5.31) é calculada como sendo

$$\boldsymbol{\theta} = \sum_{i=1}^{r} \frac{\mathbf{u}_i' \mathbf{y}}{\sigma_i} \mathbf{v}_i \tag{5.35}$$

onde $r = rank(\mathbf{W})$. O número de condicionamento da matriz **W** pode também ser calculado diretamente a partir dos valores singulares como:

$$\kappa_2(\mathbf{W}) = \frac{\sigma_1}{\sigma_M} \tag{5.36}$$

O número de condicionamento κ_2 mede, pela norma quadrática, a sensibilidade da solução para pequenas variações em **W** ou **y**. No caso de problemas mal condicionados, temos (EVSUKOFF et al., 2000)

$$\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_r \geq \sigma_{r+1} \cong \ldots \cong \sigma_M \cong 0 \tag{5.37}$$

e o rank de **W** não pode ser calculado exatamente devendo ser estimado numericamente. Na prática, uma dada tolerância $\delta > 0$, compatível com a precisão de máquina é escolhida de tal forma que:

$$\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_r \geq \delta \geq \sigma_{r+1} \geq \ldots \geq \sigma_M \tag{5.38}$$

onde $\hat{r}$ é uma estimativa numérica do rank de **W**, a qual é usada para calcular a solução de (5.31) como (EVSUKOFF et al., 2000)

$$\boldsymbol{\theta} = \sum_{i=1}^{\hat{r}} \frac{\mathbf{u}_i' \mathbf{y}}{\sigma_i} \mathbf{v}_i \tag{5.39}$$

Para a identificação do modelo nebuloso, (5.39) calcula todas as componentes do vetor de parâmetros θ, que é o caso quando o conjunto de pontos aprendidos está vazio. Contudo, a idéia básica do método, da mesma forma que em NAKOULA (1997), é que o modelo aproximativo tenha erro nulo sobre os pontos aprendidos. Assim, é necessário se impor que a saída dos pontos aprendidos estejam dentro do vetor de parâmetros θ, o que implica em uma redução do problema original. Esse problema reduzido e sua solução são descritos a seguir.

5.2.2 - O problema reduzido

Considere-se que, em uma determinada iteração do processo de identificação, existam p pontos aprendidos formando o conjunto de ponto aprendidos $P = \{(\alpha,\varphi)_1 \ldots (\alpha,\varphi)_p\}$, onde α_i representa as coordenadas do i-ésimo ponto aprendido e $\varphi_i = f(\alpha_i)$ é a saída correspondente. Chega-se ao problema reduzido quando as componentes correspondentes às saídas dos pontos aprendidos φ_i são extraídas do vetor de parâmetros θ. A solução desse problema reduzido consiste no cálculo do vetor de parâmetros reduzido θ', o qual contém apenas os valores de θ não sujeitos a vínculos (chamados *valores livres*) (EVSUKOFF et al., 2000). Tal solução deve ser calculada a partir da eliminação das colunas na equação (5.31) que correspondem às regras cujos valores de saída devem ser mantidos em φ_i, o que nos leva a procurar a solução de

$$\min_{\theta} \left\| \mathbf{W}'\theta' - \mathbf{y}' \right\|_2 \tag{5.40}$$

onde

$$\mathbf{W}' = \mathbf{W}\mathbf{A}$$

$$\mathbf{y}' = \mathbf{y} - \sum_{1 \le i \le p} \mathbf{W}\lambda_i\varphi_i \tag{5.41}$$

A matriz $\mathbf{A}$ é uma matriz de transformação que elimina as colunas de $\mathbf{W}$ relacionadas às regras correspondentes aos pontos aprendidos, cujas saídas devem ser fixadas em φ_i. A matriz $\mathbf{A}$ pode ser construída com facilidade a partir da matriz identidade de ordem M (a ordem da quantidade de regras) pela extração das colunas λ_i

cujas posições correspondam às regras geradas pelos pontos aprendidos (EVSUKOFF et al., 2000). Por exemplo, suponha que tenhamos a matriz de regressores

$$\mathbf{W} = \begin{bmatrix} w_{11} & w_{12} & w_{13} & w_{14} \\ w_{21} & w_{22} & w_{23} & w_{24} \\ \vdots & \vdots & \vdots & \vdots \\ w_{N1} & w_{N2} & w_{N3} & w_{N4} \end{bmatrix} \tag{5.42}$$

na qual a terceira coluna corresponde a uma regra gerada por um ponto aprendido. A matriz Λ será uma matriz identidade 4×4 da qual é retirada a terceira coluna, correspondente à regra do ponto aprendido. Assim, obtemos:

$$\Lambda = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \text{e} \quad \lambda_3 = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} \tag{5.43}$$

A solução θ' para (5.40) contém apenas os valores livres do vetor de parâmetros original e pode ser calculada a partir da decomposição em valores singulares da matriz $\mathbf{W}' = \mathbf{U}'\mathbf{\Sigma}'\mathbf{V}'^{t}$. Assim, obtem-se (EVSUKOFF et al., 2000)

$$\theta' = \sum_{j=1}^{r'} \mathbf{u}_j'^{t} \frac{\mathbf{y}'}{\sigma_j'} \mathbf{v}_j' \tag{5.44}$$

onde r' e σ_j' são o rank estimado e os valores singulares de $\mathbf{W}'$; $\mathbf{u}_j'$ e $\mathbf{v}_j'$ são as colunas das matrizes $\mathbf{U}'$ e $\mathbf{V}'$ e $\mathbf{y}'$ é dado por (5.41). A solução completa, com as saídas φ_i dos pontos aprendidos colocadas nas posições corretas é calculada por

$$\theta = \Lambda\theta' + \sum_{1 \le i \le p} \lambda_i \varphi_i \tag{5.45}$$

A saída do modelo nebuloso obtido com os parâmetros de saída calculados pela equação (5.45) tem erro nulo nos pontos aprendidos.

5.2.3 - Uso do método em problemas SISO

Da mesma maneira que no método anterior, inicia-se o processo de identificação com a definição do domínio do problema. Define-se esse domínio, como vimos, como $\Omega = [w_1, w_2]$, sendo $w_1 = x_{min}$ e $w_2 = x_{max}$ dentro dos limites da variável. Quando de sua inicialização, o algoritmo atribui esses limites do domínio aos protótipos dos conjuntos nebulosos associados às duas primeiras regras do modelo (fig. 5.1) de modo que (5.4) seja satisfeita. A seguir, esses pontos associados aos limites do domínio são incluídos no conjunto de pontos aprendidos fazendo-se (cf. (5.5))

$$P(k=1) = \{ (\alpha,\varphi)_1 ; (\alpha,\varphi)_2 \} \tag{5.46}$$

onde $\alpha_1 = w_1$ e $\alpha_2 = w_2$. Aqui também, os valores das saídas correspondentes são atribuídos como saída das regras (cf. (5.6)) de forma que

$$\theta(k=1) = [\varphi_1 \varphi_2] = [f(w_1)\ f(w_2)] \tag{5.47}$$

O processo iterativo se inicia com o algoritmo buscando novos pontos para serem incluídos no conjunto P de pontos aprendidos enquanto uma dada tolerância ζ não tiver sido atingida segundo o critério definido em (5.2). Novamente, em uma iteração genérica k em que o sistema tenha p pontos aprendidos, o novo ponto $(\alpha,\varphi)_{p+1}$ a ser incluído no conjunto de pontos aprendidos é o de maior distância da aproximação segundo o critério (cf. (5.7)).

$$(\alpha,\varphi)_{p+1} = \underset{x \in \mathcal{A}}{arg\,max}\left(\xi_1(x,y)^2\right) \tag{5.48}$$

Como em NAKOULA (1997), esse novo ponto $(\alpha,\varphi)_{p+1}$ calculado em (5.48) e selecionado para ser aprendido, define uma nova regra do tipo (5.3) e redefine toda a partição dos protótipos em $\mathcal{A}$ como mostra a figura 5.2 (EVSUKOFF et al., 2000). O vetor de protótipos é atualizado de forma a incluir o novo ponto aprendido segundo (5.8). Aqui, esse vetor precisa ser reordenado de forma que os protótipos estejam em

ordem estritamente crescente (fig. 5.13) e o vetor de parâmetros de saída das regras também é atualizado (cf. (5.9)), nele incluindo-se o valor de saída correspondente ao novo ponto aprendido:

$$\theta(k) = \theta(k\text{-}1) \cup \{\varphi_{p+1}\} \tag{5.49}$$

O erro total (5.2) é então calculado para o novo modelo. O processo iterativo continua até que esse erro seja menor que uma tolerância ζ arbitrariamente definida.

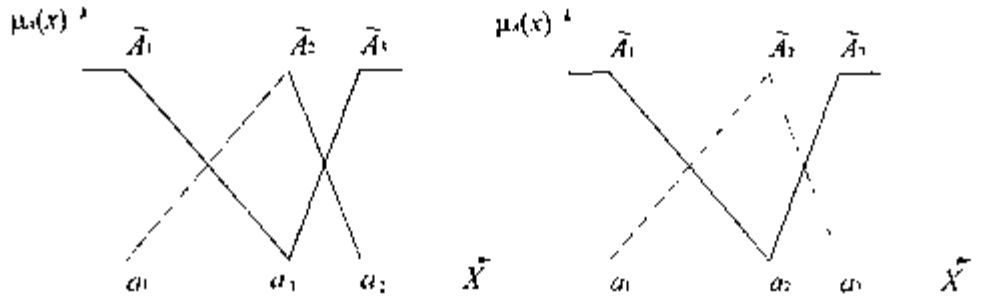


Figura 5.13 - Protótipos: (a) Antes da reordenação. (b) Depois da reordenação.

5.2.4 - Uso do método em modelos MISO

A extensão para modelos MISO do método proposto por EVSUKOFF et al.. (2000) é imediata, obedecendo aos mesmos princípios elaborados por NAKOULA et al.. (1997) já que as dimensões permanecem compatíveis em (5.24) e o domínio é o mesmo de (5.27).

5.2.4.1 - Exemplo 3

Considere-se a curva *sinc* definida no exemplo do item 4.3.4 onde os dados de treinamento são também tomados em grade regular. Usamos aqui um intervalo menor, [-7, 7] para cada variável, diminuindo a quantidade de pontos de dados, o que não afeta nosso resultado. Usando-se o método aqui proposto, obtemos um particionamento das

variáveis de entrada x e y conforme visualizado na figura 5.14. Nesta figura, os pontos da função são marcados discretizadamente, os pontos aprendidos são marcados com círculos e a aproximação obtida para a função é a linha mais fina. Observe-se que embora a base de regras seja composta de 7 x 7 = 49 regras contra 16 modeladas pelo ANFIS, as funções de pertinência tem uma interpretação lingüística muito mais apurada, havendo uma função central e as demais simetricamente afastadas desta. Também aqui, necessita-se proceder ao ajuste de apenas 49 parâmetros (os valores dos θ, que são os parâmetros conseqüentes) contra 72 parâmetros (48 conseqüentes e 24 antecedentes) no ANFIS. O EMQ obtido foi de 0.0008.

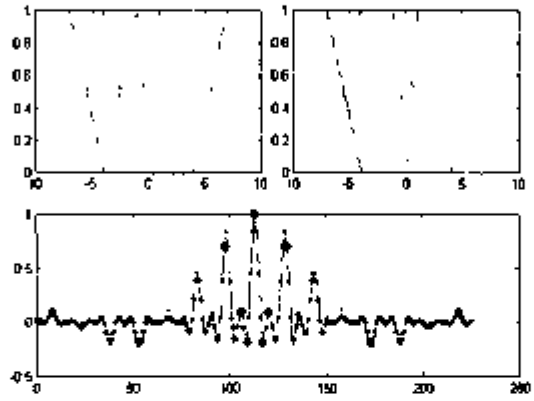


Figura 5.14 - Modelo obtido para a curva *sinc*.

5.2.4.2 - Exemplo 4

Seja o problema de modelar a curva unidimensional definida pela superfície

$$z = 4sin(10x) + cos(3y) \qquad x, y = [3, 5] \tag{5.50}$$

num conjunto de pontos desde (x_{min}, y_{min}) até (x_{max}, y_{max}) de forma a ser atingida uma tolerância para o EMQ menor que 0,01. Usaram-se 201 grupos de pontos de treinamento tomados igualmente espaçados nos domínios de cada variável de forma a modelar a curva em grade regular. O modelo obtido com o método apresentado aqui modela esta função da forma como pode ser vista na figura 5.15. Obtemos 10 protótipos em cada

variável e o modelo é então composto de 100 regras. A dimensão do vetor de parâmetros θ é 1 x 100 onde 80 θ's são livres e 20 são fixos. O EMQ é de 0.0090.

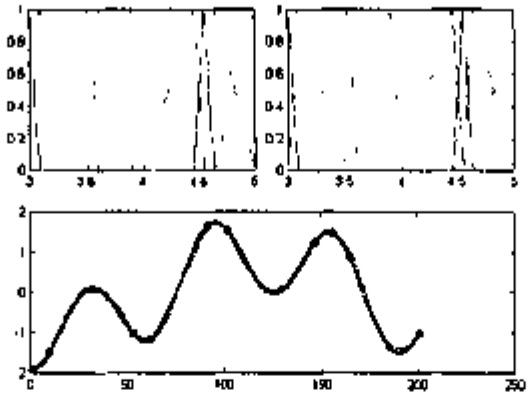


Figura 5.15 - Modelo obtido para a curva definida por (5.50).

As posições dos protótipos em x e y são iguais e dadas por

X = Y : 3,0000 3,0900 3,5200 3,6100 3,9100 4,0100 4,4600 4,5500 4,6400 5,0000

5.2.4.3 - Exemplo 5

Considere-se a modelagem de um sistema de um forno a gás no qual ar e metano são combinados de modo a formar uma mistura cujo resíduo é dióxido de carbono (CO_2) e cuja concentração é medida na saída do forno. A entrada de ar é mantida constante e a taxa de metano é tomada como variável de controle. Esse sistema é conhecido como *"benchmark" Box & Jenkis* (BOX et JENKIS, 1970) e tem sido utilizado para diversos métodos de identificação de modelos. Para fins de comparação, EVSUKOFF et al. (2000) utilizou a formulação discreta usada por NAKOULA (1997) na qual o conjunto de dados original de uma única variável de entrada foi convertido no conjunto de treinamento T de duas variáveis de entrada no formato $(\mathbf{x}(t), y(t))$, onde $\mathbf{x}(t) = [u(t-3) \quad y(t-1)]$.

Com o algoritmo desenvolvido, pode ser obtido um modelo composto de 36 regras (M = 36) onde foi feito o aprendizado de $p = 4$ pontos que se vê na figura 5.16

(EVSUKOFF et al., 2000). O particionamento nebuloso das funções de pertinência obtidas pode ser visualizado para cada variável na parte superior da figura 5.16 (respectivamente $x_1(t) = u(t-3)$ e $x_2(t) = y(t-1)$). Os valores da variável de saída tanto no conjunto de dados como na aproximação do modelo também são mostrados plotados contra o tempo na parte inferior da figura. Observe-se no detalhe os pontos aprendidos marcados com círculos.

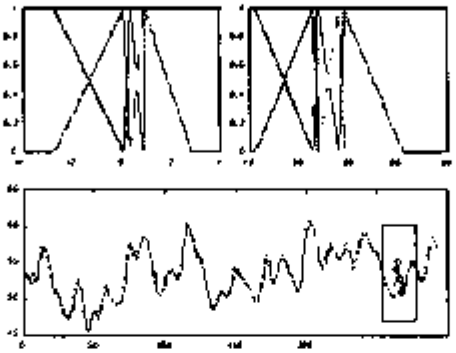


Figura 5.16 - Resultados para 36 regras.

De forma a permitir uma comparação mais direta com os resultados apresentados por NAKOULA et al. (1997), foi gerado um modelo composto por 90 regras (M = 90) onde foram aprendidos p = 8 pontos, como pode ser visto na figura 5.17.

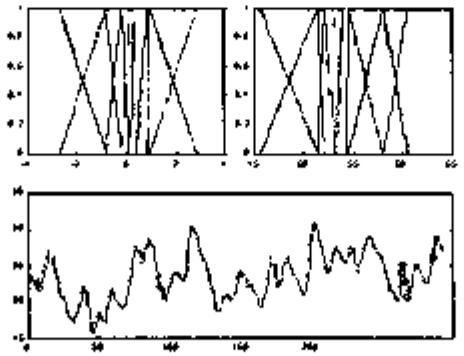


Figura 5.17 - Resultados para 90 regras.

Tabela 5.1 - Modelagem de Box & Jenkis com diversos métodos.

Método	Ano	Regras	EMQ
Box & Jenkis	(1970)	-	0,710
Tong	(1980)	19	0,469
Xu & Lu	(1987)	25	0,328
Pedrycz	(1984)	81	0,320
Nakoula et al.	(1997)	90	0,175
Evsukoff et al.	(1998)	36	0,153

A tabela 5.1 apresenta uma comparação entre os resultados obtidos por este modelo e aqueles obtidos por NAKOULA et al., (1997) e outros autores. Pode-se ver que os resultados obtidos por EVSUKOFFf et al. (2000) são bem melhores do que os obtidos anteriormente.. Mesmo o modelo mais simples, de 36 regras, obteve um desempenho melhor do que o apresentado por NAKOULA et al., (1997) com 90 regras.

5.3 - Considerações

O método usado por NAKOULA et al. (2000) foi bem sucedido em termos de modelar funções unidimensionais de variadas complexidades. Sua implementação, contudo, é algo confusa em vista da necessidade de repetidos recálculos dos pesos das regras do modelo. A base de regras dos modelos é de fácil visualização mas, algumas vezes, de difícil interpretação, como resultado da inferência nebulosa de tipo 2. Ao passarmos para duas dimensões, a visualização da base de regras é muito mais prejudicada, em vista das combinações dos termos das premissas levando as regras a terem pesos de difícil interpretação. Apesar de sua robustez, o método é pouco eficiente, pela fixação de todos valores de saída das regras. Assim, apesar de não termos regras redundantes como nos modelos adaptáveis estudados, a base de regras pode crescer a níveis muito elevados.

Com a abordagem desenvolvida por EVSUKOFF et al. (2000), os modelos de sistemas bidimensionais podem ter sua base de regras sensivelmente reduzida, pelo recálculo de alguns valores de saída das regras, os chamados θ *livres*. Esses valores são

eficientemente calculados por uma técnica robusta, a DVS, e os modelos obtidos, mesmo com bases de regras mais compactas, conseguem apresentar melhores aproximações. Apesar do grande avanço, essa abordagem ainda apresenta duas limitações importantes: em modelos unidimensionais ela reduz-se ao método de NAKOULA et al. (1997), pois não existem, então, valores livres. Já em modelos com mais de 3 dimensões, como as posições dos protótipos são fixas, não adaptáveis, a quantidade de partições cresce rapidamente e o algoritmo proposto exige o cálculo de um número muito grande de valores livres.

6

Um modelo expansível e auto-adaptável

6.1 - Objetivo

Como visto nos capítulos precedentes, tanto os sistemas adaptáveis (ou ajustáveis) como os expansíveis possuem vantagens e desvantagens que estão descritas resumidamente na tabela 6.1. O objetivo desta tese foi o de, mantendo e unindo as vantagens de tais sistemas, eliminar ou diminuir tanto quanto possível as desvantagens que apresentam. Assim, segundo o critério de seleção de modelos de minimização do erro médio quadrático apresentado no capítulo 3, devemos ser capazes de construir um modelo neuro-nebuloso que consiga manter um equilíbrio adequado entre suas capacidades de acuidade e de generalização e que também evite a explosão combinatória causada pela *maldição da dimensionalidade*.

Tabela 6.1 - Comparação entre os modelos ajustáveis e expansíveis

	Vantagens	Desvantagens
Modelos adaptáveis	adaptável sem expansão	baixa interpretabilidade
	bases de regras compactas	funções não normalizadas
Modelos expansíveis	alta interpretabilidade	não adaptável sem expansão
	funções simétricas	bases de regras extensas

Uma clara vantagem dos modelos ajustáveis reside na sua capacidade de adaptação dos parâmetros das regras nebulosas em função das características do sistema sob estudo e das precisões requeridas. Devido à sua própria característica de construção,

contudo, essa adaptação tem tanto limitações como alguns efeitos indesejados, por exemplo, a perda da capacidade de interpretação lingüística do modelo. Prosseguindo na busca de uma modelagem não-paramétrica de sistemas (item 3.2.2), nossa abordagem deve ser, assim, construir modelos adaptáveis de sistemas não-lineraes que não só mantenham uma boa interpretação lingüística de suas funções de pertinência como levem ao modelo mais compacto possível.

As condições de convergência também devem ser asseguradas para tal modelo. Se garantirmos que dentro de uma tolerância máxima ζ possuímos funções de base com as quais conseguimos uma minimização consistente para (3.3), o teorema de Stone-Weierstrass garante tal convergência. Nesta tese, utilizamos como funções de base tanto triângulos (que podem ser consideradas B-splines de ordem 2) como curvas Splines polinomiais.

A adaptabilidade dos modelos é conseguida com o uso da regra delta tal como no Anfis, com a diferença de que aqui mantemos o particionamento da unidade entre curvas adjacentes, assegurando uma normalização constante, o que evita dificuldades de interpretação lingüística. Como no trabalho de EVSUKOFF et al. (2000), utiliza-se a fatoração DVS para o cálculo dos θ na parte conseqüente das regras de sistemas TSK de ordem zero como em (5.1) ou (5.3).

6.2 - A regra delta

O modelo procurado é suposto ter tanto uma estrutura expansível como os parâmetros das funções de pertinência adaptáveis. Seja Ep o critério de erro para um determinado ponto definido como

$$Ep = (\hat{y} - y)^2 \tag{6.1}$$

onde, segundo nossa terminologia, $\hat{y}$ é a resposta do modelo e y é a saída desejada. O valor da saída $\hat{y}$ é resultado de (3.28) onde f é definida segundo um ou mais parâmetros

e o denominador é sempre igual a 1, pois asseguramos o particionamento da unidade entre funções adjacentes. Seja α um parâmetro de f e portanto de Ep a ser ajustado de forma a diminuir o próprio Ep. Sua contribuição intrínseca ao valor de Ep é tal que

$$\frac{\partial Ep}{\partial \alpha} = \frac{\partial Ep}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial \alpha} \tag{6.2}$$

Um conjunto de dados de treinamento é composto de um número arbitrário de pontos. Para N pontos de treinamento, o erro total é

$$E_t = \sum_{p=1}^{N} Ep \tag{6.3}$$

Assim, podemos considerar que a contribuição de α ao erro total é

$$\frac{\partial E_t}{\partial \alpha} = \sum_{p=1}^{N} \frac{\partial Ep}{\partial \hat{y}_p} \frac{\partial \hat{y}_p}{\partial \alpha} \tag{6.4}$$

Procuramos minimizar o erro total alterando o valor de α. Segundo o capítulo 4, o ajuste de α, $\Delta\alpha$, é dado por (4.18), ou seja:

$$\Delta\alpha = -\eta \frac{\partial E_t}{\partial \alpha} = -\eta \sum_{p=1}^{N} \frac{\partial Ep}{\partial \hat{y}_p} \frac{\partial \hat{y}_p}{\partial \alpha} \tag{6.5}$$

sendo η o passo de aprendizagem.

A expansão da estrutura de um dado modelo ocorre aqui segundo o mesmo critério proposto por NAKOULA (1997) e adotado também por EVSUKOFF et al. (2000), ou seja, para o refinamento das partições dos espaços de entrada, na fase iterativa do processo, novos conjuntos nebulosos são criados tendo seus pontos modais nas posições dos protótipos definidos segundo (5.7).

No caso da modelagem aqui apresentada, os diversos α_i que devem ser ajustados são os pontos c_i que definem as posições dos protótipos dos conjuntos nebulosos das C variáveis de entrada (a parte antecedente) das regras. Esses protótipos nada mais são, como sabemos, que as posições modais das funções de pertinência das regras. Como na fase iterativa em um passo k qualquer podemos ter i protótipos, devemos ajustar i-2 deles por variável, excluindo-se os protótipos que definem os extremos do intervalo. Se definirmos as posições dos protótipos de uma variável C qualquer como $c_1, c_2, \ldots, c_{i-1}, c_i$, tendo como c_i um protótipo genérico (fig. 6.1), então, $\alpha_1 = c_i$ e

$$\Delta c_i = -\eta \frac{\partial E_i}{\partial c_i} = -\eta \sum_{p=1}^{N} \frac{\partial Ep}{\partial \hat{y}_p} \frac{\partial \hat{y}_p}{\partial c_i} = -\eta \sum_{p=1}^{N} 2(\hat{y}_p - y_p) \frac{\partial \hat{y}_p}{\partial c_i} \tag{6.6}$$

com η, o passo de aprendizagem, sendo definido tal qual em (4.19) como

$$\eta = \frac{\lambda}{\sqrt{\sum_{c_i} \left(\frac{\partial E_i}{\partial c_i} \right)^2}} \tag{6.7}$$

onde λ é a constante de aprendizagem, dependente do problema em foco e, a partir de (6.1)

$$\frac{\partial Ep}{\partial \hat{y}_p} = 2(\hat{y}_p - y_p) \tag{6.8}$$

Substituindo η na equação (6.6) por seu valor explícito em (6.7), obtemos a expressão que nos importa, o ajuste da posição de um protótipo c_i qualquer:

$$\Delta c_i = -\frac{\lambda}{\sqrt{\sum_{c_i} \left(\frac{\partial E_i}{\partial c_i} \right)^2}} \sum_{p=1}^{N} 2(\hat{y}_p - y_p) \frac{\partial \hat{y}_p}{\partial c_i} \tag{6.9}$$

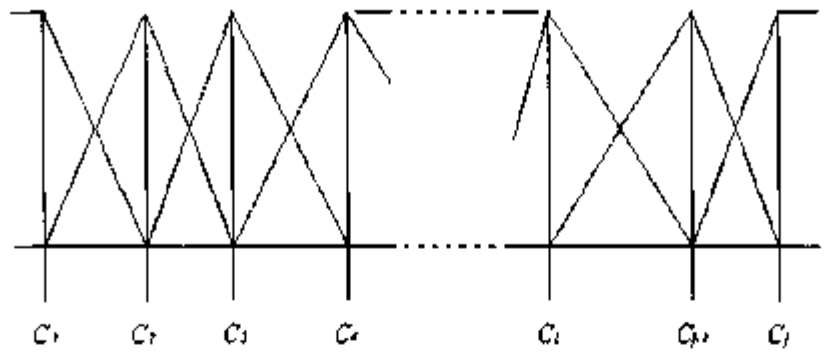


Figura 6.1 - Protótipos definindo os centros das funções de pertinência.

Isto estabelecido, devemos então determinar, para todos os N pontos e para c_{i+2} protótipos de cada variável de entrada, os valores de $\frac{\partial \hat{y}_p}{\partial c_i}$. Para isso, preenchemos, para cada variável de entrada, o Jacobiano

$$\begin{bmatrix} \frac{\partial \hat{y}_1}{\partial c_1} & \frac{\partial \hat{y}_1}{\partial c_2} & \cdots & \frac{\partial \hat{y}_1}{\partial c_i} \\ \frac{\partial \hat{y}_2}{\partial c_1} & \frac{\partial \hat{y}_2}{\partial c_2} & \cdots & \frac{\partial \hat{y}_2}{\partial c_i} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial \hat{y}_N}{\partial c_1} & \frac{\partial \hat{y}_N}{\partial c_2} & \cdots & \frac{\partial \hat{y}_N}{\partial c_i} \end{bmatrix} \tag{6.10}$$

6.3 - Aplicação a modelos SISO

No caso de um modelo SISO, um ponto genérico p possui pertinências em dois conjuntos nebulosos adjacentes A_i e A_{i+1} dadas por μA_i e μA_{i+1} ou, simplificadamente, A_i e A_{i+1}. Cada um desses conjuntos define uma regra do tipo

$$\underline{\text{SE}} \quad x \text{ é } A_i \quad \underline{\text{ENTÃO}} \quad \hat{y} = \theta_i \tag{6.11}$$

Assim, a saída do modelo para esse ponto p é dada por

$$\hat{y}_p = A_i^r \theta_i + A_{i-1}^r \theta_{i+1} \tag{6.12}$$

Para os N pontos do conjunto de treinamento e incluindo-se todos os conjuntos nebulosos, temos

$$\begin{bmatrix} \hat{y}_1 \\ \vdots \\ \hat{y}_N \end{bmatrix} = \begin{bmatrix} A_1^1 & A_2^1 & \cdots & A_i^1 & A_{i+1}^1 & \cdots & A_n^1 \\ & & & \vdots & & & \\ A_1^N & A_2^N & \cdots & A_i^N & A_{i+1}^N & \cdots & A_n^N \end{bmatrix} \begin{bmatrix} \theta_1 \\ \vdots \\ \theta_i \\ \vdots \\ \theta_n \end{bmatrix} \tag{6.13}$$

Recordando (5.29), consideramos que a matriz $\mathbf{W} \in I^{N \times M}$ armazena a descrição dos pontos do conjunto de treinamento na forma matricial

$$\mathbf{W} = \begin{bmatrix} \mathbf{w}_1 \\ \vdots \\ \mathbf{w}_N \end{bmatrix} = \begin{bmatrix} \mu_{w_1}(Z_1) & \cdots & \mu_{w_1}(Z_M) \\ \vdots & \ddots & \vdots \\ \mu_{w_N}(Z_1) & \cdots & \mu_{w_N}(Z_M) \end{bmatrix} \tag{6.14}$$

Como os elementos da matriz $\mathbf{W}$ podem ser identificados como W_{ij}, então

$$\begin{bmatrix} \hat{y}_1 \\ \vdots \\ \hat{y}_N \end{bmatrix}_{N \times 1} = \begin{bmatrix} W_{11} & W_{12} & \cdots & W_{1M} \\ \vdots & \vdots & \cdots & \vdots \\ W_{N1} & W_{N2} & \cdots & W_{NM} \end{bmatrix}_{N \times M} \begin{bmatrix} \theta_1 \\ \vdots \\ 0_M \end{bmatrix}_{M \times 1} \tag{6.15}$$

ou em forma mais compacta,

$$\hat{\mathbf{y}} = \mathbf{W}\,\theta \tag{6.16}$$

Para um ponto genérico p, a derivada parcial em relação a um protótipo genérico a_i é dada por

$$\frac{\partial \hat{y}_p}{\partial a_i} = \frac{\partial A_i}{\partial a_i}\theta_i + \frac{\partial A_{i+1}}{\partial a_i}\theta_{i+1} \tag{6.17}$$

O cálculo dessas derivadas é facilitado caso consideremos as funções de pertinência como compostas de duas funções independentes e simétricas, uma, f , à esquerda (f_L) e outra, g , à direita (f_R). No caso de serem usados conjuntos nebulosos triangulares e considerando-se os protótipos c_i de uma variável de entrada C qualquer, temos (fig. 6.2):

$$f(x) = f_L(x) = \frac{x - c_{i+1}}{c_i - c_{i+1}} \tag{6.18a}$$

$$g(x) = f_R(x) = \frac{x - c_i}{c_{i+1} - c_i} \tag{6.18b}$$

ambas com derivadas contínuas em c_i e c_{i+1} dadas por

$$\frac{\partial f}{\partial c_i} = \frac{c_{i+1} - x}{(c_i - c_{i+1})^2} \quad , \quad \frac{\partial f}{\partial c_{i+1}} = \frac{x - c_i}{(c_i - c_{i+1})^2} \tag{6.19a}$$

$$\frac{\partial g}{\partial c_i} = \frac{x - c_{i+1}}{(c_{i+1} - c_i)^2} \quad , \quad \frac{\partial g}{\partial c_{i+1}} = \frac{c_i - x}{(c_{i+1} - c_i)^2} \tag{6.19b}$$

Os valores de pertinência de um ponto x qualquer são, assim,

$$\mu_i(x) = f(x) \quad \text{e} \quad \mu_R(x) = g(x) \tag{6.20}$$

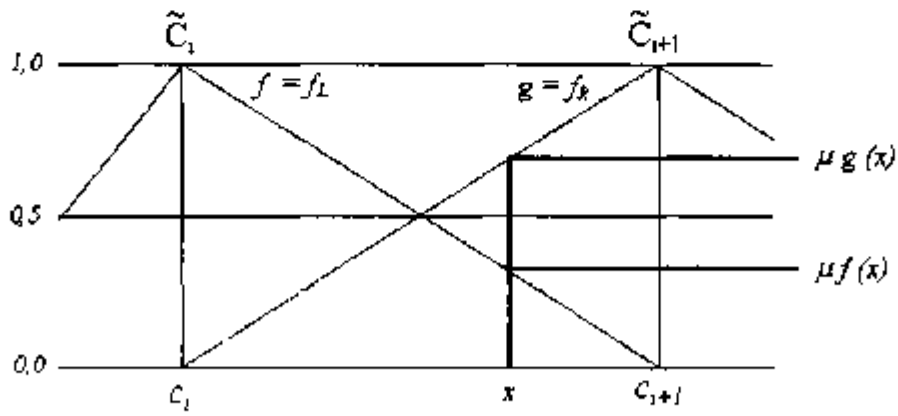


Figura 6.2 - Funções de pertinência simétricas à esquerda (f_L) e à direita (f_R).

São essas derivadas que preenchem o Jacobiano (6.10) e que são usadas no somatório (6.9).

6.4 - Aplicação a modelos MISO

No caso de um modelo MISO, o cálculo é mais complexo e depende da quantidade de variáveis de entrada. Consideremos, sem perda de generalidade, que nosso modelo tipo MISO possui duas variáveis independentes x_1 e x_2 cada uma com m e n protótipos, respectivamente, a_1, ..., a_m e b_1, ..., b_n, definidos em seus universos de suporte numa dada iteração k. Considere-se que as formas dos conjuntos nebulosos sejam funções convexas quaisquer definidas no plano cartesiano por $f(x_1) = A(x_1)$ e $g(x_2) = B(x_2)$. Como, em cada variável de entrada, um ponto p possui , no máximo, pertinências em dois conjuntos nebulosos, apenas quatro regras (no máximo) podem ser ativadas simultâneamente. Então, por simplicidade,

$$\hat{y}_p = A_i B_j \theta_{\quad} + A_{i+1} B_j \theta_{(i+1)\quad} + A_i B_{j+1} \theta_{\quad(j+1)} + A_{i+1} B_{j+1} \theta_{(i+1)(j+1)} \tag{6.21}$$

Contudo, não sabemos de antemão quais conjuntos nebulosos terão as pertinências de p. Incluindo-se todas as combinações possíveis, a saída do modelo para um ponto p é dada por

$$\begin{aligned} \hat{y}_p = {} & A_1 B_1 \theta_1 + A_1 B_2 \theta_2 + \ldots + A_1 B_n \theta_n + \\ & A_2 B_1 \theta_{n+1} + A_2 B_2 \theta_{n+2} + \ldots + A_2 B_n \theta_{2n} + \\ & \vdots \\ & A_m B_1 \theta_{((m-1)n)+1} + A_m B_2 \theta_{((m-1)n)+2} + \ldots + A_m B_n \theta_{((m-1)n)+n=mn} \end{aligned} \tag{6.22}$$

Em forma matricial, para os N pontos do conjunto de treinamento e incluindo-se todos os termos, (6.22) se torna

$$\begin{bmatrix} \hat{y}_1 \\ \vdots \\ \hat{y}_N \end{bmatrix} = \begin{bmatrix} A_1^1 B_1^1 & A_1^1 B_2^1 & A_1^1 B_3^1 & \cdots & A_3^1 B_1^1 & \cdots & A_4^1 B_1^1 & \cdots & A_*^1 B_*^1 \\ & & & & \vdots & & & & \\ & & & & A_3^N B_*^N & & & & \end{bmatrix} \begin{bmatrix} \theta_1 \\ \theta_2 \\ \vdots \\ \theta_n \end{bmatrix} \tag{6.23}$$

Para o cálculo de (6.9) devemos achar:

$$\text{Em } x_1: \quad \frac{\partial A_1}{\partial a_1}, \frac{\partial A_1}{\partial a_2}, \cdots, \frac{\partial A_1}{\partial a_m}, \frac{\partial A_2}{\partial a_1}, \frac{\partial A_2}{\partial a_2}, \cdots, \frac{\partial A_m}{\partial a_m} \tag{6.24a}$$

$$\text{Em } x_2: \quad \frac{\partial B_1}{\partial b_1}, \frac{\partial B_1}{\partial b_2}, \cdots, \frac{\partial B_1}{\partial b_n}, \frac{\partial B_2}{\partial b_1}, \frac{\partial B_2}{\partial b_2}, \cdots, \frac{\partial B_n}{\partial b_n} \tag{6.24b}$$

Claro que muitas das derivadas acimas serão nulas porém, para facilitar a implementação em algoritmo, é conveniente manter-se o cômputo, em vista da característica de expansibilidade do modelo. As derivadas parciais em relação aos protótipos em x_1 são

$$\begin{aligned}\frac{\partial \hat{y}_r}{\partial a_1} = \quad & \frac{\partial A_1}{\partial a_1}(B_1\theta_1 + B_2\theta_2 + \cdots + B_n\theta_n) + \\ & \frac{\partial A_2}{\partial a_1}(B_1\theta_{n+1} + B_2\theta_{n+2} + \cdots + B_n\theta_{2n}) + \\ & \vdots \\ & \frac{\partial A_m}{\partial a_1}(B_1\theta_{(m-1)n+1} + B_2\theta_{(m-1)n+2} + B_n\theta_{mn})\end{aligned} \tag{6.25}$$

e, similarmente para qualquer $\frac{\partial \hat{y}_r}{\partial a_i}$.

Frisamos que apenas duas das derivadas parciais dos conjuntos nebulosos acima são não-nulas, contudo, pela característica de expansibilidade do modelo, algoritmicamente, é conveniente deixar todos os termos da soma. Os protótipos em x_2 tem as derivadas parciais dadas por

$$\begin{aligned}\frac{\partial \hat{y}_r}{\partial b_1} = \quad & \frac{\partial B_1}{\partial b_1}(A_1\theta_1 + A_2\theta_{n+1} + \cdots + A_m\theta_{(m-1)n+1}) + \\ & \frac{\partial B_2}{\partial b_1}(A_1\theta_2 + A_2\theta_{n+2} + \cdots + A_m\theta_{(m-1)n+2}) + \\ & \vdots \\ & \frac{\partial B_n}{\partial b_1}(A_1\theta_n + A_2\theta_{2n} + \cdots + A_m\theta_{mn})\end{aligned} \tag{6.26}$$

e similarmente para qualquer $\frac{\partial \hat{y}_{ip}}{\partial b_j}$.

O problema então consiste em utilizar as formas explícitas das derivadas em (6.24). No caso do uso de triângulos como funções de pertinência, elas são dadas por (6.19). A expansão para qualquer número de variáveis ocorre naturalmente.

Mais especificamente, sem qualquer perda de generalidade, consideremos um modelo MISO a duas entradas e uma saída onde as regras sejam do tipo de (5.1), ou seja,

$$\underline{\text{SE}} \quad x_1 \text{ é } A_i \quad \underline{\text{E}} \quad x_2 \text{ é } B_j \quad \underline{\text{ENTÃO}} \quad \hat{y} = \theta_k \tag{6.27}$$

e que em uma determinada iteração k tal modelo tenha particionado os espaços das variáveis de forma que tenhamos 4 conjuntos nebulosos em x_1 com as posições dos protótipos em a_1, a_2, a_3 e a_4 e 3 conjuntos nebulosos em x_2 com os protótipos em b_1, b_2 e b_3 (fig. 6.3). A base de regras de tal modelo possui, nesse instante, 4 x 3 = 12 regras do tipo de (6.27) por uma combinação em árvore similar à da figura 5.10. Um ponto p genérico, como por exemplo, o mostrado na figura 6.3 possui, em relação aos conjuntos nebulosos definidos nesse instante, as pertinências

$$A_1^p, A_2^p, A_3^p, A_4^p, B_1^p, B_2^p, B_3^p$$

e a saída do modelo para esse ponto p é dada a partir de (6.21) por

$$\hat{y}_p = A_i B_j \theta_{i,j} + A_{i+1} B_j \theta_{(i+1),j} + A_i B_{j+1} \theta_{i,(j+1)} + A_{i+1} B_{j+1} \theta_{(i+1)(j+1)} \tag{6.28}$$

Para um ponto genérico, não sabemos quais conjuntos nebulosos lhe terão pertinências. Dessa forma, a saída $\hat{y}_k$ fornecida por esse modelo de 12 regras a uma entrada qualquer $\mathbf{x}_k$ é

$$\begin{aligned} \hat{y}_k = {} & A_1 B_1 \theta_1 + A_1 B_2 \theta_2 + A_1 B_3 \theta_3 + A_2 B_1 \theta_4 + A_2 B_2 \theta_5 + A_2 B_3 \theta_6 + \\ & A_3 B_1 \theta_7 + A_3 B_2 \theta_8 + A_3 B_3 \theta_9 + A_4 B_1 \theta_{10} + A_4 B_2 \theta_{11} + A_4 B_3 \theta_{12} \end{aligned} \tag{6.29}$$

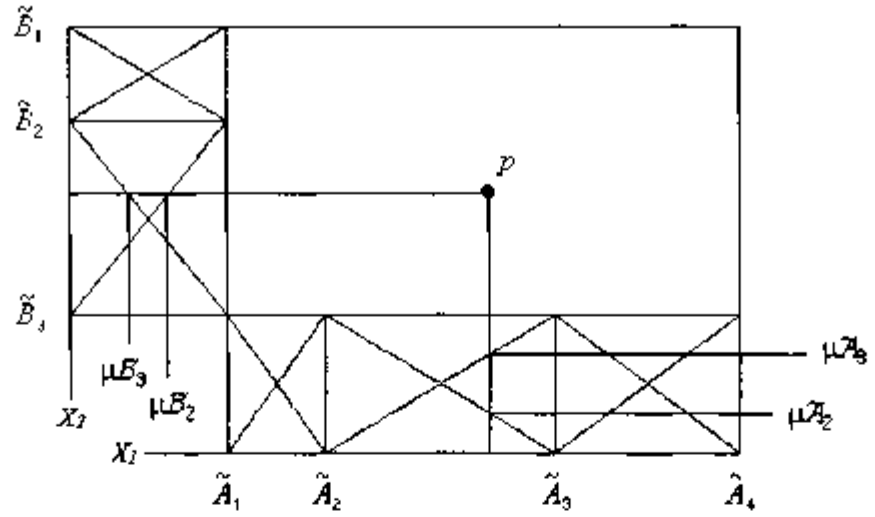


Figura 6.3 - Pertinências de um ponto "p" numa iteração hipotética k.

E as derivadas parciais em relação aos protótipos em x_1 são

$$\begin{aligned}\frac{\partial \hat{y}_k}{\partial a_1} = \quad & \frac{\partial A_1}{\partial a_1}(B_1\theta_1 + B_2\theta_2 + B_3\theta_3) + \\ & \frac{\partial A_2}{\partial a_1}(B_1\theta_4 + B_2\theta_5 + B_3\theta_6) + \\ & \frac{\partial A_3}{\partial a_1}(B_1\theta_7 + B_2\theta_8 + B_3\theta_9) + \\ & \frac{\partial A_4}{\partial a_1}(B_1\theta_{10} + B_2\theta_{11} + B_3\theta_{12})\end{aligned} \tag{6.30}$$

e, similarmente para $\frac{\partial \hat{y}_k}{\partial a_2}$, $\frac{\partial \hat{y}_k}{\partial a_3}$ e $\frac{\partial \hat{y}_k}{\partial a_4}$. Os protótipos em x_2 tem as derivadas parciais dadas por

$$\begin{aligned}\frac{\partial \hat{y}_k}{\partial b_1} = \quad & \frac{\partial B_1}{\partial b_1}(A_1\theta_1 + A_2\theta_4 + A_3\theta_7 + A_4\theta_{10}) + \\ & \frac{\partial B_2}{\partial b_1}(A_1\theta_2 + A_2\theta_5 + A_3\theta_7 + A_4\theta_{11}) + \\ & \frac{\partial B_3}{\partial b_1}(A_1\theta_3 + A_2\theta_6 + A_3\theta_9 + A_4\theta_{12})\end{aligned} \tag{6.31}$$

e similarmente para $\frac{\partial \hat{y}_k}{\partial b_2}$ e $\frac{\partial \hat{y}_k}{\partial b_3}$.

Essas derivadas são as usadas na regra delta na equação (6.9) no lugar dos termos $\frac{\partial \hat{y}_L}{\partial c_i}$.

O modelo neuro-fuzzy a ser construído é visualizado na topologia da rede neuronal da figura 6.4.

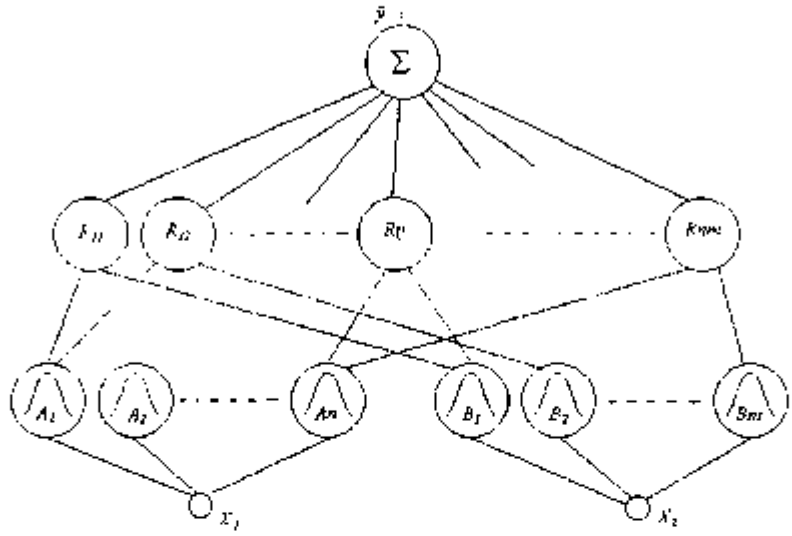


Figura 6.4 - Rede neuronal dos modelos neuro-fuzzy desta tese.

Nessa figura, o nível mais inferior é o da entrada das variáveis do problema. A figura mostra duas variáveis, x_1 e x_2. O nível acima é o das funções de base das funções de pertinência. Cada nó desse nível associa uma partição nebulosa para as variáveis de entrada. É um nível expansível horizontalmente, à medida que as partições aumentam. O nível seguinte é o das regras que combinam as partições das variáveis. Ele também expande-se horizontalmente para acomodar novas regras. A unidade do último nível faz a associação das saídas das regras e fornece a saída $\hat{y}$ calculada pelo modelo.

6.5 - Algoritmo de aprendizagem

Na inicialização, a base de regras possui apenas as regras correspondentes aos limites do domínio do problema, tal como em NAKOULA (1997) e EVSUKOFF et al.

(2000). Os θ iniciais são calculados por DVS. A evolução do algoritmo é feita em duas fases. Definida uma tolerância ζ, a cada iteração é calculado um novo ponto do espaço de entrada segundo (5.7) para referenciar novos protótipos nas partições das variáveis de entrada, aumentando a estrutura do problema e novos θ são calculados por DVS para todas as regras, inclusive as que definem os extremos dos intervalos. Em seguida, uma adaptação dessa estrutura é feita, modificando-se as posições dos protótipos "livres" (os protótipos dos limites do domínio são "fixos") com o uso da regra delta da equação (6.9). Cada vez que a(s) posição(ões) do(s) protótipo(s) é(são) modificada(s), todos os θ são recalculados por DVS.

Aqui, notam-se outras diferenças em relação a EVSUKOFF et al. (2000). Com exceção dos limites do domínio, não existem "pontos aprendidos". Da mesma forma, como os protótipos livres podem mover-se livremente entre os extremos dos intervalos das variáveis de entrada, todos os θ podem e devem ser recalculados no sentido de diminuir ainda mais o erro médio quadrático (EMQ) definido por (5.2). Também não existe mais a necessidade de ser resolvido o problema reduzido. Enquanto o erro diminuir, os protótipos livres são continuamente modificados e todos os θ são recalculados de acordo.

Caso a tolerância especificada seja atingida, o algoritmo pára e o modelo é considerado completo. Se durante o ajuste dos protótipos o EMQ subir, um novo protótipo é determinado por (5.7), novos θ são calculados por DVS e o processo de ajuste recomeça. Da mesma maneira que nos métodos do capítulo 5, a convergência do algoritmo é assegurada pelo fato de o erro ser nulo quando existirem protótipos suficientes para cobrir todos os pontos de treinamento do intervalo.

6.5.1 - Exemplo 1

Vejamos a evolução passo a passo de um processo de identificação. Seja o exemplo 4 do item 5.2.4.2 do capítulo 5. A tolerância especificada permanece em 0,01. Neste exemplo chega-se à convergência em 10 passos, aqui marcados de (a) até (j).

(a) Após a definição dos limites do domínio. O EMQ para todos os pontos de treinamento é 0.5135 e temos os protótipos fixos localizados em (fig. 6.5a):

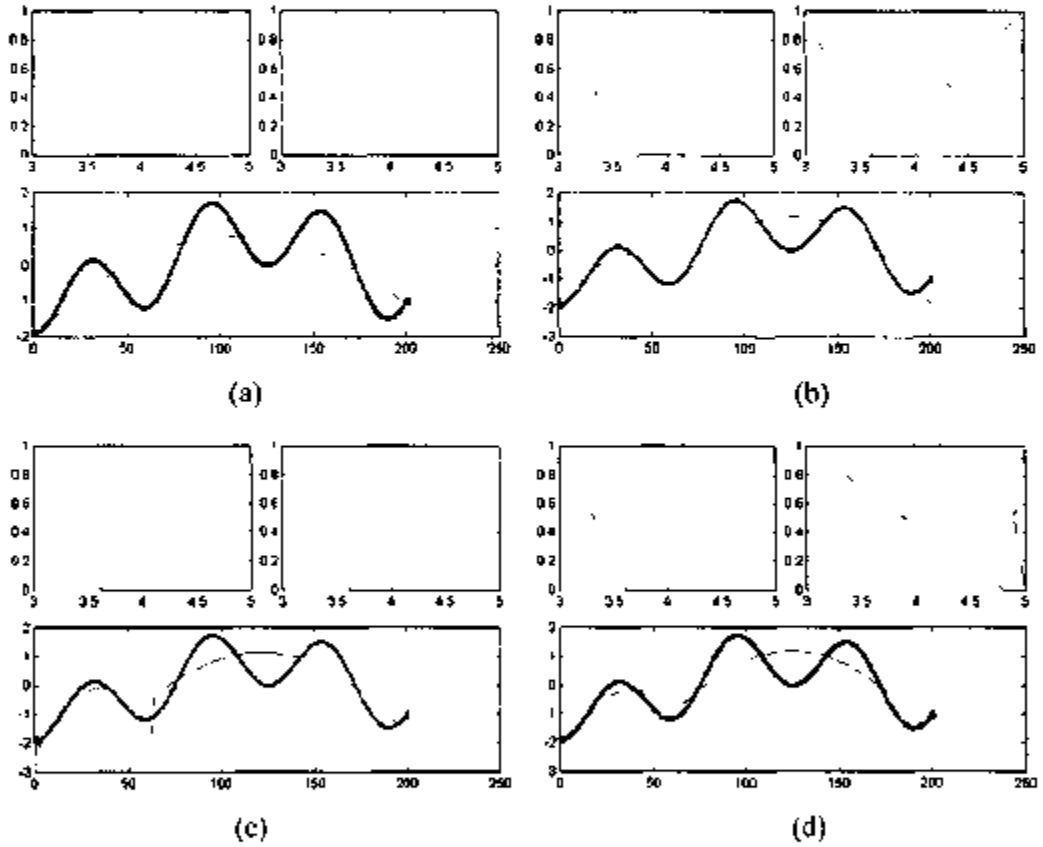
x_1 : 3,0000 5,0000

x_2 : 3,0000 5,0000

(b) Um novo protótipo (livre) é selecionado segundo (5.7). A estrutura do modelo aumenta e o EMQ cai para 0,3224. As posições dos protótipos é (fig. 6.5b)

x_1 : 3,0000 3,6100 5,0000

x_2 : 3,0000 3,6100 5,0000



(a) (b)

(c) (d)

Figura 6.5 - Evolução do modelo desde (a) partições iniciais até (d) aproximação com 9 regras.

(c) Os protótipos livres são ajustados segundo (6.9). O EMQ cai para 0,2930 e os protótipos (fig. 6.5c) ocupam as posições

x_1: 3,0000 3,6216 5,0000

x_2: 3,0000 3,6274 5,0000

(d) Os protótipos livres são novamente ajustados (fig. 6.5d) segundo (6.9) e o EMQ sobe para 0,3099. Os protótipos agora ocupam as posições

x_1: 3,0000 3,6196 5,0000

x_2: 3,0000 4,8105 5,0000

(e) Como o EMQ subiu, os protótipos são recolocados na posição do passo (c) anterior e novos protótipos são inseridos de acordo com (5.7). O EMQ cai para 0,1253 e os protótipos ocupam as posições (fig. 6.6a)

x_1: 3,0000 3,6216 4,2500 5,0000

x_2: 3,0000 3,6274 4,2500 5,0000

(f) O EMQ cai para 0,1083 quando os protótipos são ajustados (fig. 6.6b) para

x_1: 3,0000 3,6259 4,3144 5,0000

x_2: 3,0000 3,6286 4,2322 5,0000

(g) Novo ajuste é feito (fig. 6.6c) e o erro cai para 0,0194. Os protótipos ocupam as posições

x_1: 3,0000 3,2353 4,2332 5,0000

x_2: 3,0000 3,6266 4,6109 5,0000

(h) O algoritmo procede novo ajuste aos protótipos livres (fig. 6.6d) e o erro sobe para 0,1579 quando as posições são

x_1: 3,0000 3,5977 3,7336 5,0000

x_2: 3,0000 3,6246 4,6089 5,0000

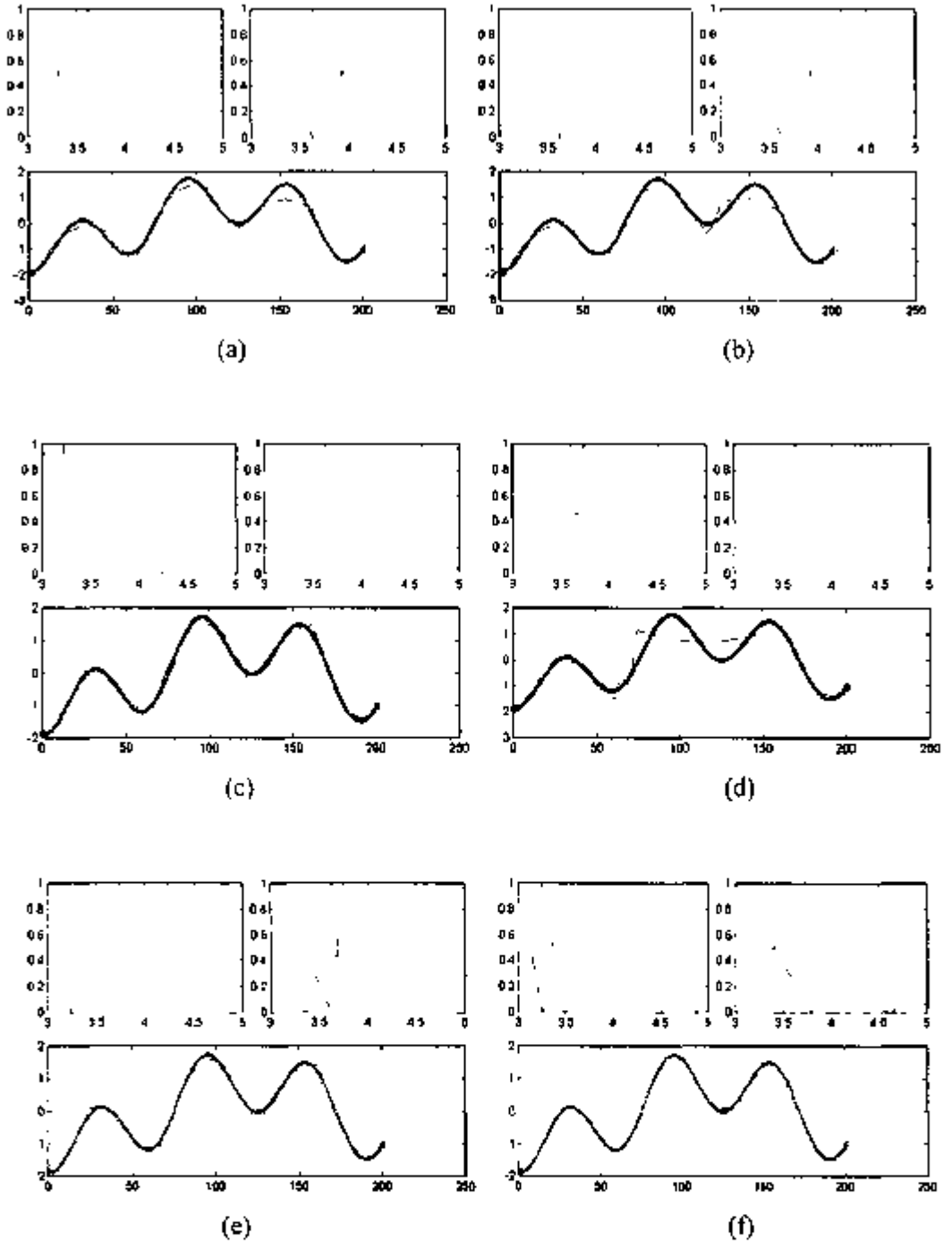


Figura 6.6 - Evolução do modelo desde (a) 16 regras até (f) convergência em 25 regras.

(i) Como o erro subiu, os protótipos livres retornam às posições do passo (g) , novos protótipos são calculados por (5 7) e o EMQ cai para 0,0180. As posições dos protótipos são (fig. 6.6e)

x_1 . 3,0000 3.2356 3.7400 4.2332 5,0000

x_2 : 3,0000 3,6266 3,7400 4,6109 5,0000

(j) Faz-se o ajuste dos protótipos livres. O EMQ cai para 0,0037, abaixo da tolerância especificada. O algoritmo termina e o modelo nebuloso final (fig. 6.6f) tem os protótipos nas posições

x_1 : 3,0000 3,2514 3,4690 4,0499 5,0000
x_2 : 3,0000 3,7986 4,4974 4,6553 5,0000

com 25 posições no vetor de parâmetros θ.

A constante de aprendizagem utilizado na regra delta (6.9) foi diferente para cada variável de entrada. Usou-se $\lambda = 2$ para x_1 e $\lambda = 3$ para x_2. O algoritmo precisou de 9 iterações para gerar um modelo com 25 regras na base de regras e que aproximam a função com um EMQ = 0.0037 para os dados de treinamento. O método anterior (EVSUKOFF et al., 2000) necessitava das mesmas 9 iterações para gerar um modelo com 100 regras na base de regras e cuja aproximação da função era de 0,0090. Os resultados estão na tabela 6.2.

Tabela 6.2 - Teste e validação dos resultados

Método	Tolerância	Regras	EMQ
Evsukoff et al.	< 0,01	100	0,0090
Este trabalho	< 0,01	25	0,0037

6.5.2 - Algoritmo

O algoritmo pode ser usado para qualquer número de variáveis de entrada com a expansão horizontal da rede da figura 6.4. Definida uma tolerância ζ, para m variáveis de entrada, definido um conjunto de dados de treinamento $T = \{(x, y)_1, \ldots, (x, y)_N\}$,

$x_i \in R^n$, $i \in R$, $i_i = f(x_i)$... definido um valor para λ, tal algoritmo pode ser resumido pelos passos que se seguem.

INICIO

1: Calcular os limites do domínio $\Omega = \{\omega_1, \ldots, \omega_{2^m}\}$

2: SE $\omega_k \in T$, $k = 1 \ldots 2^m$, então $P \leftarrow P \cup (\omega_k, f(\omega_k))$

3: Gerar as partições iniciais $\mathbf{A}_i(x_i)$, $i = 1 \ldots m$ com os protótipos sendo definidos pelos limites do domínio.

4: Calcular todos os θ por DVS

5: Para dados = 1, ... , N calcular o EMQ J segundo (5.2)

6: REPETIR

7: SE $J \geq \zeta$

8: Gerar novos protótipos em cada variável nos pontos definidos por (5.7)

9: Calcular todos os θ por DVS

10: Para dados = 1, ... , N calcular o EMQ J segundo (5.2)

11: SE $J \geq \zeta$ FAZER

12: ENQUANTO $J_{novo} \leq J_{antigo}$ e $J \geq \zeta$

13: Para todas as variáveis, ajustar os picos livres por (6.9) usando (em duas variáveis) (6.25) e (6.26)

14: Recalcular todos os θ por DVS

15: Para dados = 1, ... , N calcular o EMQ J segundo (5.2)

16: FIM - ENQUANTO

17: FIM - SE

18: SENÃO **FIM**

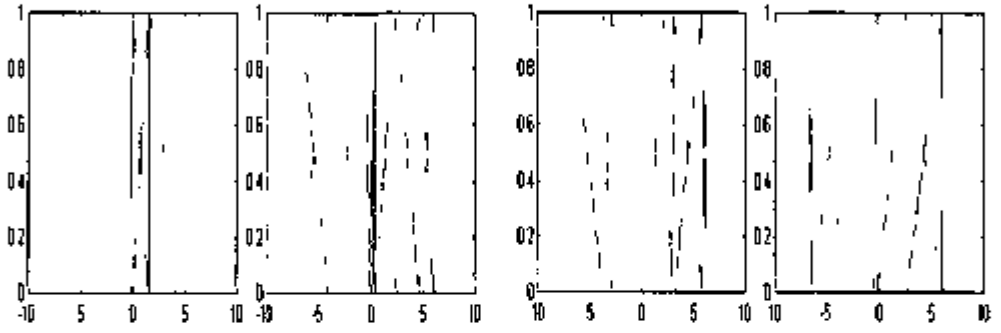
19: FIM - SE

20: FIM-REPETIR

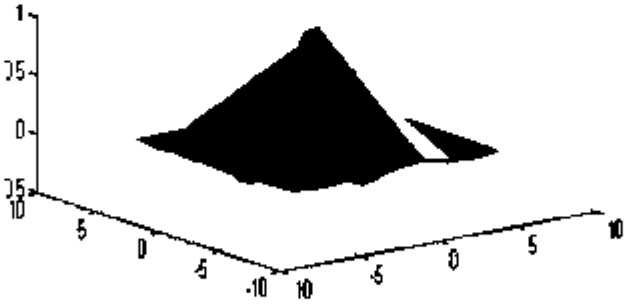
6.6 - Exemplos

6.6.1 - Exemplo 2

No exemplo 3 do capitulo 5, item 5.2.4.1, pode-se verificar como a abordagem proposta por EVSUKOFF et al. (2000) modelou a superficie *sinc* com apenas 49 regras. Naquele exemplo, os dados estavam distribuidos em grade regular no intervalo de -7 a 7 para as variáveis x e y. No mundo real temos, em geral, os dados distribuidos de uma forma completamente aleatória. Neste exemplo, foram gerados 225 pontos aleatórios de teste em grade completamente irregular para modelar a curva *sinc*



(a) (b)



(c)

Figura 6.7 - Modelos obtidos para a curva *sinc* com dados totalmente aleatórios. (a) Evsukoff et al. (b) Presente. (c) Superficie reconstruida.

Mantida a tolerância daquele exemplo, a modelagem de EVSUKOFF et al. (2000), neste caso, precisou de 144 regras para modelar a curva com um EMQ = 0.0009.

A abordagem de construção de modelos neuro-fuzzy desenvolvida nesta tese construiu um modelo com apenas 81 regras que aproxima a superfície com um EMQ = 0.0008. Os modelos neuro-fuzzy das duas técnicas podem ser vistos na figura 6.7 (a) e (b) e a superfície reconstruida com a técnica aqui desenvolvida está mostrada na figura 6.7(c).

6.6.2 - Exemplo 3

Considere-se a modelagem da superfície formada pela curva do exemplo 1. A tolerância especificada permanece em 0,01. Para este teste foram gerados 196 pontos de treinamento totalmente aleatórios em grade irregular para as modelagens (uma grade de 196 pontos em x_1 e 196 pontos em x_2 onde apenas 196 pares das interseções foram considerados). O modelo gerado pelo método de Evsukoff et al é constituido de 64 regras. A abordagem aqui colocada construiu um modelo com apenas 49 regras. Tais modelos e a superfície obtida estão na figura 6.8.

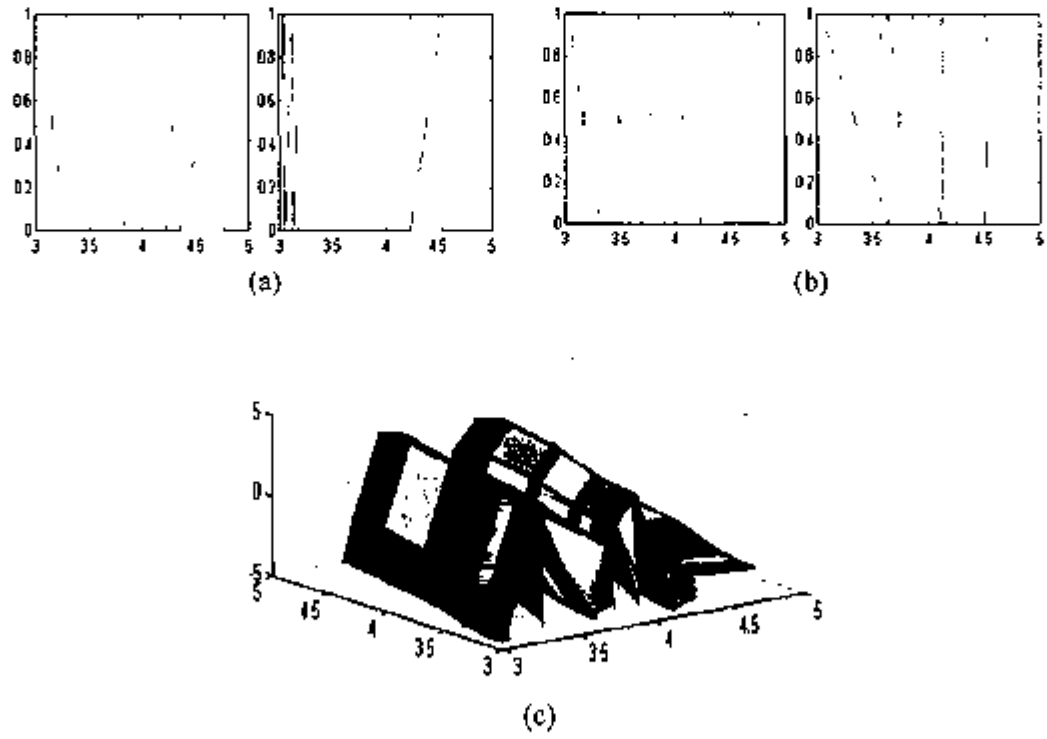


Figura 6.8 - Modelos obtidos com (a) Evsukoff et al. (b) Presente.
(c) Superfície reconstruída.

6.6.3 - Exemplo 4

Tomemos a curva (5.19) apresentada no exemplo 2 do capítulo 5. Utilizando-se a técnica apresentada neste capítulo, obteve-se o modelo exibido na figura 6.9. O modelo gerado é composto de 16 regras e aproxima os dados do conjunto de treinamento com um EMQ de 0,0005. Para fins de comparação, segundo o item 5.1.1.2, o método de NAKOULA (1997) gera um modelo composto de 20 regras cujo EMQ em relação aos dados do conjunto de treinamento é de 0,0027. Para teste da aproximação, 10 conjuntos de 100 pontos aleatórios foram gerados e a média do EMQ foi de 0.0011. Naquele método, essa média foi de 0,0062.

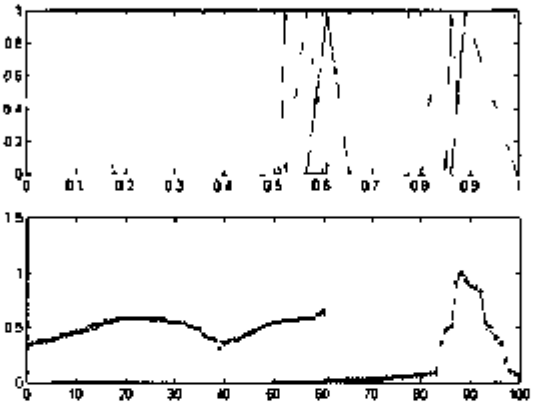


Figura 6.9 - Modelo obtido para a curva (5.19)

6.6.4 - Exemplo 5

KOSKO (1996) apresenta um conjunto de curvas adequado ao teste de funções de pertinência. Tomamos para este exemplo de teste as curvas unidimensionais geradas desde (x_{min}, y_{min}) até (x_{max}, y_{max}) por um conjunto de pontos aleatórios pertencentes às superfícies $g_1(x,y)$, $g_2(x,y)$ e $g_3(x,y)$ definidas como:

$$a)\ g_1(x,y) = f_2(x) \times 10\left[sen(4y + 0,1) + sen(11y - 0,2) + sen(14y) + sen(17y + 0,3)\right]$$

$$\text{com} \quad f_2(x) = 10\,arctg\left(\frac{(x - 0,2)(x - 0,7)(x + 0,8)}{x + 1,4}\right) \qquad x, y = [\ -1\ ,\ 1\ ]$$

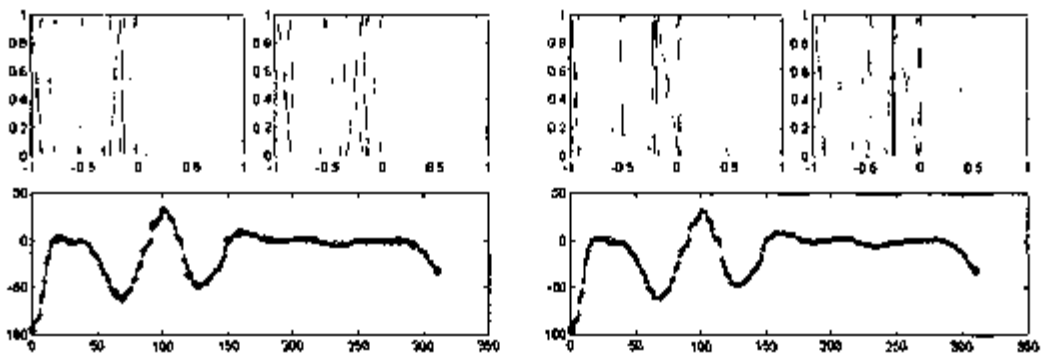
$$\text{b) } g_2(x,y) = f_4(x) \times 2\left[exp-\left(\frac{y-0,1}{0,25}\right)^2 - 0,8\, exp-\left(\frac{y+0,75}{0,15}\right)^2 - 0,4\, exp-\left(\frac{y-0,8}{0,1}\right)^2 \right]$$

$$\text{com} \quad f_4(x) = 8\, sen(10x^2 + 5x + 1) \qquad x, y = [\ -1\ ,\ 1\ ]$$

$$\text{c) } g_3(x,y) = f_1(x) \times sen(y)$$

$$\text{com} \quad f_1(x) = 3x(x-1)(x-1,9)(x-0,7)(x+1,8) \quad x, y = [\ -2\ ,\ 2\ ]$$

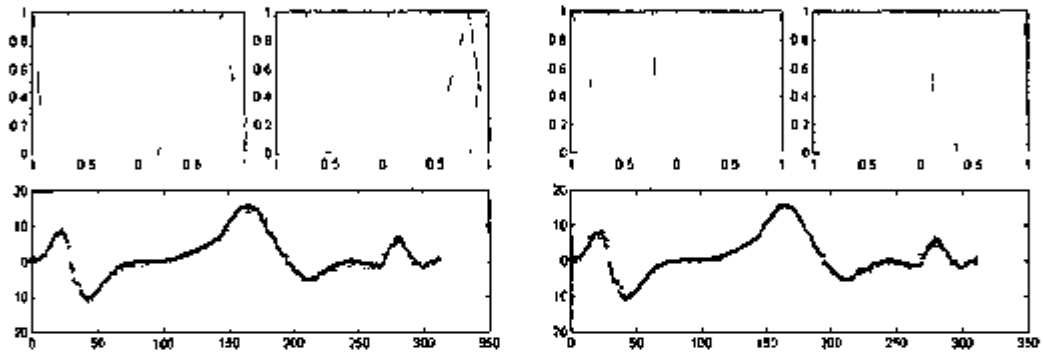
Nosso objetivo é o de modelar as curvas ao longo de pares de pontos gerados aleatoriamente. Foram gerados 310 conjuntos de pontos aleatórios para treinamento das curvas $g_1(x,y)$ e $g_2(x,y)$ (310 pontos em x e idem para y). Para a curva $g_3(x,y)$ foram gerados 250 conjuntos de pontos de treinamento aleatórios. Também foram gerados, para cada curva, 10 grupos de conjuntos de pontos aleatórios para teste. A tabela 6.3 sumariza os resultados obtidos com os modelos de EVSUKOFF et al. (2000) e com o apresentado neste trabalho. As colunas de EMQ de dados de teste apresentam as médias obtidas para os 10 conjuntos de teste em cada curva. Pode ser visto que, embora a quantidade de funções de pertinência (e, conseqüentemente, de regras) seja normalmente bastante reduzida com a modelagem aqui proposta, não existe perda de precisão ou de generalidade do modelo. As figuras 6.10, 6.11 e 6.12 comparam os modelos neuro-nebulosos obtidos com a técnica aqui apresentada e a do método anterior.



(a) (b)

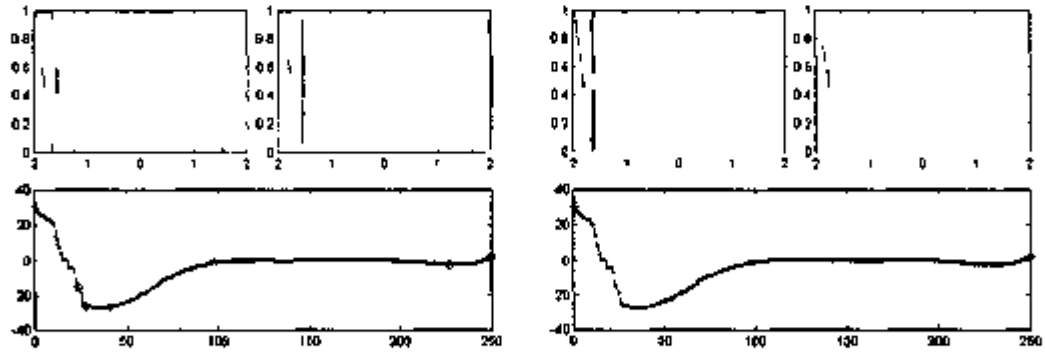
Figura 6.10 - $g_1(x,y)$. Modelos obtidos com (a) Evsukoff et al. (b) Esta tese.

A curva $g_1(x,y)$ apresenta uma alta amplitude (aprox. 125 unidades) no seu espaço de saída pois os valores absolutos de y variam de -95 a 31 com um comportamento bastante oscilatorio. Esse fato levou ao estabelecimento da tolerância em 5 unidades. Valores menores de tolerância, aumentariam muito a complexidade dos modelos. A curva $g_2(x,y)$ tem uma amplitude dos dados na saída bem menor, na faixa de 26 unidades, o que possibilitou uma tolerância quanto ao EMQ menor (1 unidade). Apesar da curva $g_3(x,y)$ ter uma amplitude de saída em torno de 57 unidades, seu comportamento quase linear em grande parte de seu espectro possibilitou o estabelecimento da tolerância do EMQ em 0,5 unidades.



(a) (b)

Figura 6.11 - $g_2(x,y)$. Modelos obtidos com (a) Evsukoff et al. (b) Esta tese.



(a) (b)

Figura 6.12 - $g_3(x,y)$. Modelos obtidos com (a) Evsukoff et al. (b) Esta tese.

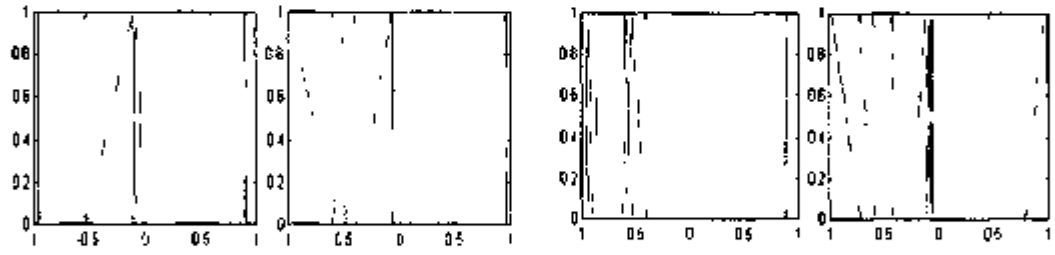
Os valores de λ usados para a modelagem das curvas foram fixados, respectivamente, para g_1, g_2 e g_3 como $\lambda_1 = \lambda_2 = 10$, $\lambda_1 = \lambda_2 = 1$ e $\lambda_1 = \lambda_2 = 1$.

Tabela 6.3 - Quantidade de regras e EMQ dos modelos obtidos

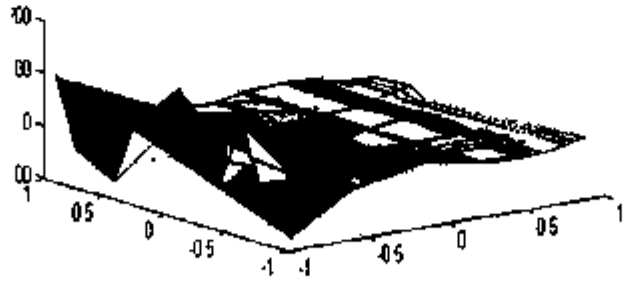
Curva	Regras		Tolerância	EMQ treinamento		EMQ teste	
	Anterior	Presente	(unidades)	Anterior	Presente	Anterior	Presente
$g_1(x,y)$	169	169	< 5	4,6025	3,1650	5,7382	3,5344
$g_2(x,y)$	81	49	< 1	0,8073	0,9693	1,1203	0,9825
$g_3(x,y)$	49	36	< 0,5	0,1684	0.1433	0,1729	0,1528

6.6.5 - Exemplo 6

Considere-se a modelagem das superfícies definidas pelas curvas do exemplo anterior. Para este teste, como no exemplo 2, também foram gerados 196 pontos de treinamento totalmente aleatórios em grade irregular para as modelagens de todas as curvas, respeitados os intervalos definidos para as variáveis em cada problema.



(a) (b)



(c)

Figura 6.13 - Modelos obtidos com (a) Evsukoff et al. (b) Presente.
(c) Superfície reconstruída.

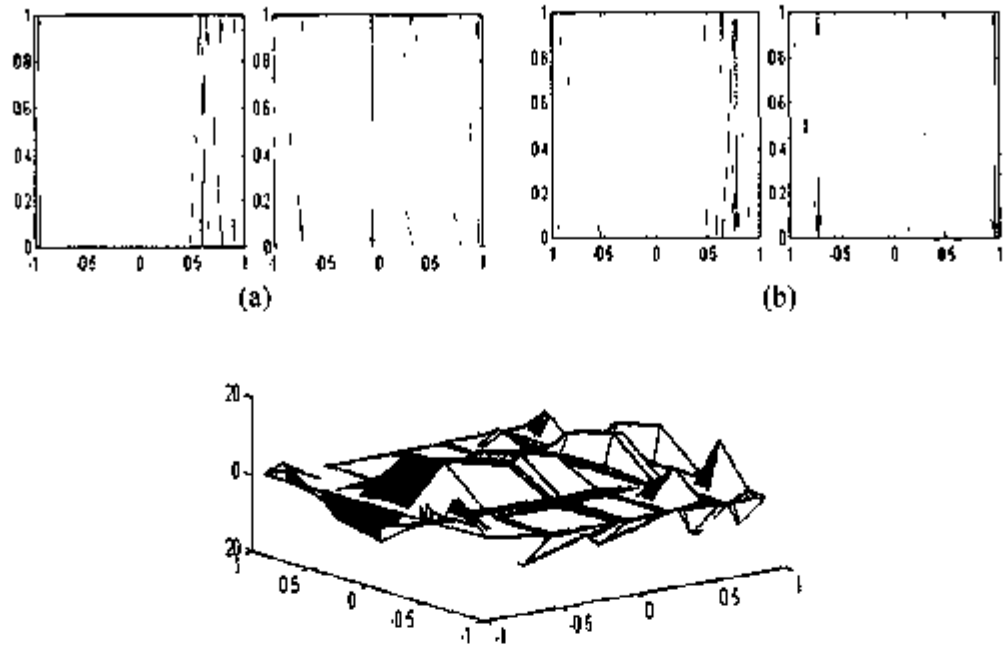


Figura 6.14 - Modelos para $g_2(x,y)$ obtidos com (a) Evsukoff et al. (b) Presente. (c) Superfície reconstruída com a presente abordagem.

Para a superfície $g_1(x,y)$, considerada uma tolerância menor do que 2 unidades, a abordagem de EVSUKOFF et al. (2000) construiu um modelo de 256 regras com um EMQ = 1.5062. A modelagem apresentada aqui construiu um modelo com 196 regras com um EMQ = 1,8216. Os modelos obtidos e a superfície reconstruída podem ser vistos na figura 6.13.

Na modelagem da superfície definida por $g_2(x,y)$ foi definida uma tolerância menor do que 1 unidade. A presente abordagem gerou um modelo composto de 100 regras com um EMQ = 0,8945. O método de EVSUKOFF et al. (2000) necessitou de 156 regras para gerar um modelo abaixo da tolerância, quando o EMQ atingiu 0,4510. Os modelos gerados e a superfície reconstruída pela presente abordagem estão na figura 6.14.

Modelou-se a superfície definida por $g_3(x,y)$ com uma tolerância menor do que 0,5 unidade para o EMQ. A abordagem de EVSUKOFF et al. (2000) gerou um modelo com 144 regras para atingir um EMQ = 0,2064. O método aqui apresentado construiu um modelo com 64 regras que atingiu um EMQ = 0.2505. Os modelos obtidos com as

duas abordagens e a superfície construída estão na figura 6.15.

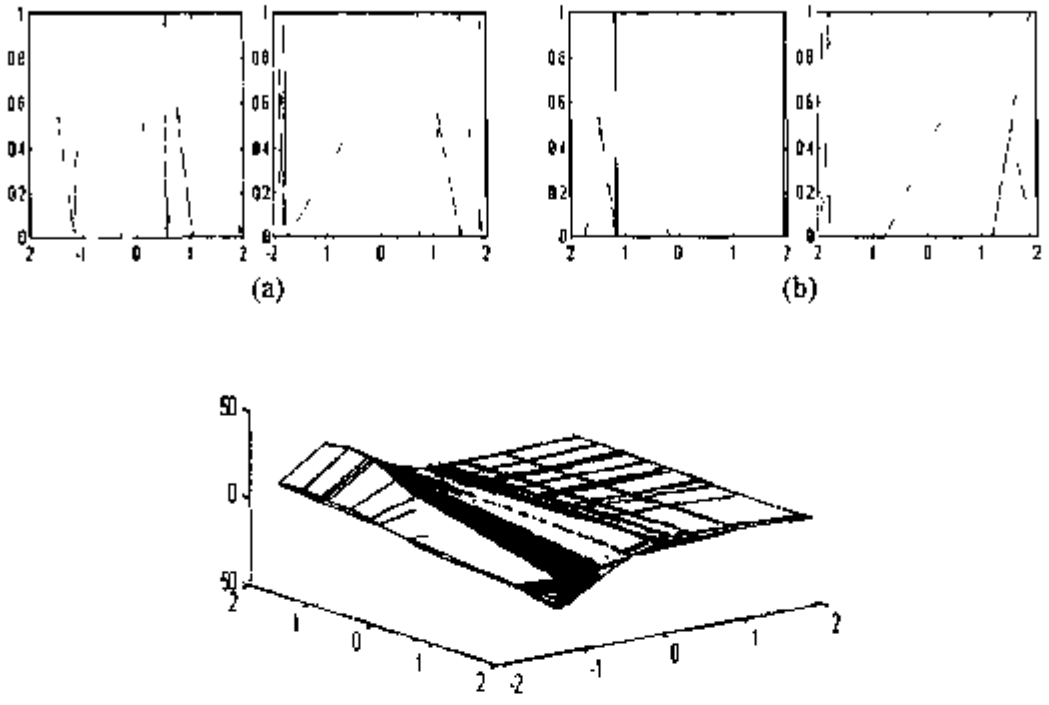


Figura 6.15 - Modelos para $g_3(x,y)$ obtidos com (a) Evsukoff et al. (b) Presente. (c) Superfície reconstruída com a presente abordagem.

6.6.6 - Exemplo 7 - Teste de Box & Jenkis

O benchmark de Box & Jenkis também foi utilizado para validar a modelagem aqui proposta. Para este teste, o valor de λ foi de 0,1. A figura 6.16 mostra o modelo obtido e a tabela 6.4 compara o desempenho do modelo com o de outras técnicas.

Tabela 6.4 - Resultados de Box & Jenkis para diversos métodos.

Método	Regras	Erro Med. Quad.
Nakoula et al. (1997)	90	0,175
Evsukoff et al. (1998)	90	0,090
presente	49	0,096
presente	90	0,042

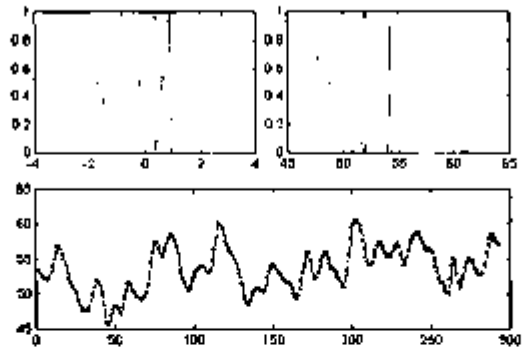


Figura 6.16 - Modelo obtido para o benchmark de Box & Jenkis.

6.7 - Uso de Splines

As Splines já foram apresentadas no capítulo 3 desta tese. Vinhamos até agora usando como funções de pertinência os triângulos, que podem ser vistos, obedecidas certas condições, como B-Splines de ordem dois. Consideremos agora que as nossas funções à esquerda, f_L e à direita f_R , respectivamente, f e g (fig. 6.17) sejam definidas segundo as equações

$$f(x) = a_0 + a_1x + a_2x^2 + a_3x^3 \tag{6.32}$$

$$g(x) = b_0 + b_1x + b_2x^2 + b_3x^3 \tag{6.33}$$

cujas derivadas são

$$f'(x) = a_1 + 2a_2x + 3a_3x^2 \tag{6.34}$$

$$g'(x) = b_1 + 2b_2x + 3b_3x^2 \tag{6.35}$$

Se forem obedecidas as condições de contorno

$$f(C_i) = 1 \;;\; f(P_{i,i}) = 0.5 \;;\; f(C_{i-1}) = 0 \;;\; f'(C_i) = 0 \;;\; f'(C_{i-1}) = 0 \tag{6.36a}$$

$$g(C_i) = 0 \;;\; g(P_{i,i}) = 0.5 \;;\; g(C_{i-1}) = 1 \;;\; g'(C_i) = 0 \;;\; g'(C_{i+1}) = 0 \tag{6.36b}$$

as curvas (6.32) e (6 33) são consideradas Splines cúbicas.

O objetivo é verificar se as Splines podem ser adequadas à metodologia de aproximação de sistemas aqui proposta. Já foi visto que a regra delta necessita de funções explícitas para o preenchimento do Jacobiano (6.10) cujos termos são usados em (6.9). Assim, tal como com os triângulos, necessita-se explicitar as derivadas (6.34) e (6.35) em termos de c_i e c_{i+1} para os cálculos das derivadas parciais (6.24). O problema passa a ser o de determinar os valores das constantes (não confundir com protótipos!) a_0 , a_1 , a_2 , a_3 , b_0 , b_1 , b_2 e b_3 em (6.32) e (6.33). Esse cálculo não apresenta maiores dificuldades e basta que calculemos um dos dois conjuntos de constantes a ou b pois, como as curvas f e g são simétricas,

$$b_0 = 1 - a_0 \;\; ; \;\; b_1 = - a_1 \;\; ; \;\; b_2 = - a_2 \;\; ; \;\; b_3 = - a_3 \tag{6.37}$$

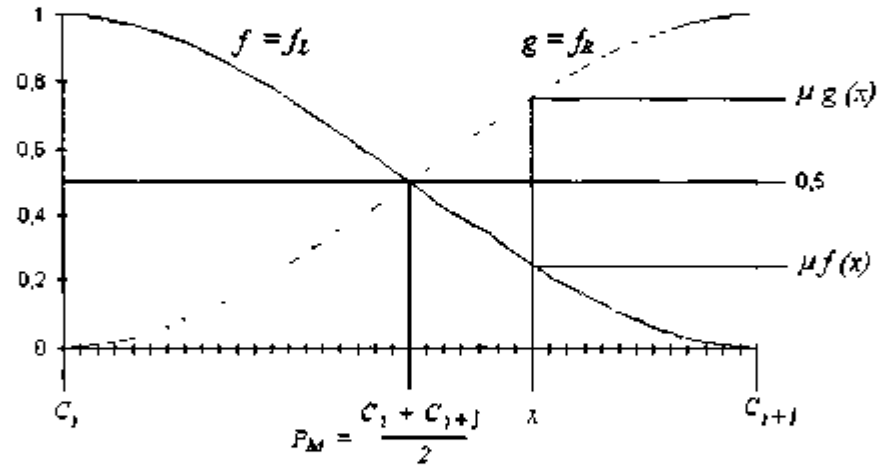


Figura 6.17 - Splines usadas como funções de pertinência

Em termos de c_1 e c_2, os valores explícitos das constantes a_{0-3} é

$$a_3 = \frac{2}{-3(c_i^2 - c_{i-1}^2)(c_i - c_{i-1}) + 4c_{i-1}^3 - 6c_i c_{i-1}^2 + 2c_i^3} \tag{6.38}$$

$$a_2 = \frac{-3a_3(c_i^2 - c_{i-1}^2)}{2(c_i - c_{i-1})} \tag{6.39}$$

$$a_1 = 3a_3c_{i-1}\left[\frac{(c_i^2 - c_{i-1}^2)}{c_i - c_{i-1}} - c_{i-1}\right] \tag{6.40}$$

$$a_0 = a_3c_{i-1}^2\left[2c_{i-1} - \frac{3(c_i^2 - c_{i-1}^2)}{2(c_i - c_{i-1})}\right] \tag{6.41}$$

As derivadas parciais destes termos em relação a c_i e c_{i+1} devem ser calculadas para o preenchimento do Jacobiano (6.10) para o uso de seus termos nas equações (6.30) e (6.31) já que, com Splines,

$$\frac{\partial f_i}{\partial c_i} = \frac{\partial}{\partial c_i}(a_0 + a_1x + a_2x^2 + a_3x^3) \tag{6.42a}$$

e

$$\frac{\partial f_{i+1}}{\partial c_i} = \frac{\partial}{\partial c_i}(b_0 + b_1x + b_2x^2 + b_3x^3) \tag{6.42b}$$

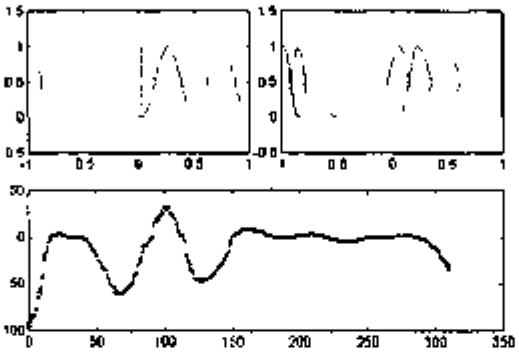
6.7.1 - Exemplo 8

Usou-se as Splines como definidas em (6.32) e (6.33) para modelar aproximações para funções de duas variáveis de entrada e para efeito de comparação foram usadas as curvas $g_1(x,y)$; $g_2(x,y)$ e $g_3(x,y)$ do item 6.6.4. Os resultados obtidos encontram-se comparados com os de triângulos na tabela 6.5. A figura 6.18 ilustra a aproximação e as funções de pertinência obtidas para (a) $g_1(x,y)$, (b) $g_2(x,y)$ e (c) $g_3(x,y)$.

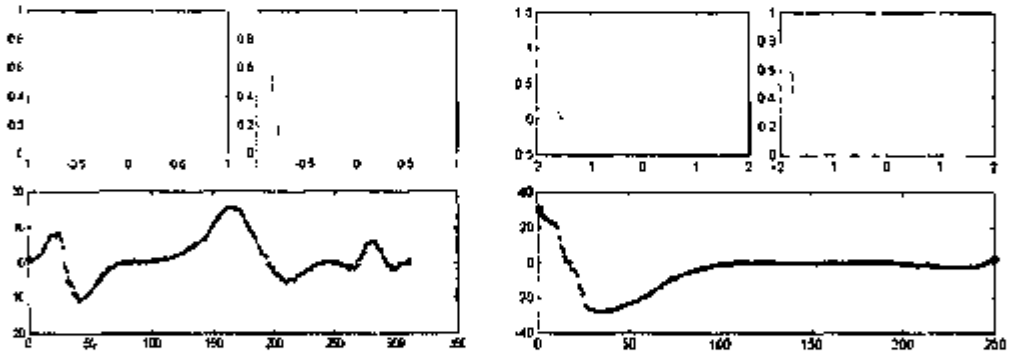
Observa-se que os modelos obtidos com Splines apresentam uma boa redução da base de regras e/ou do EMQ tanto para os conjuntos de treinamento como para os conjuntos de dados de teste. Os valores de λ foram: 200 para a curva $g_1(x,y)$; 1 para $g_2(x,y)$ e $g_3(x,y)$.

Tabela 6.5 - Comparação Triângulos x Splines.

Curva	Regras		Tolerância EMQ	EMQ treinamento		EMQ teste	
	Triângulos	Splines	(unidades)	Triângulos	Splines	Triângulos	Splines
$g_1(x,y)$	169	121	< 5	3,1650	2,8302	3,5344	2,9572
$g_2(x,y)$	49	36	< 1	0,9693	0,3819	0,9825	0,3923
$g_3(x,y)$	36	25	< 0,5	0,1433	0,2041	0,1528	0,2113



(a)



(b) (c)

Figura 6.18 - (a) $g_1(x,y)$, (b) $g_2(x,y)$ e (c) $g_3(x,y)$. Modelos obtidos com Splines.

6.7.2 - Exemplo 9

Usemos a curva do item 5.1.1.2 (exemplo 2 do capítulo 5) para verificar a validade da aproximação de modelos construídos com Splines em sistemas SISO. O valor de λ neste caso foi de 0,1 e o modelo obtido para a curva é composto de 14 regras

que levam a um EMQ de 0,0009 para o conjunto de treinamento e de 0,0012 para o conjunto de dados de teste, composto de 10 grupos de dados aleatórios. Esse resultado é melhor do que o modelo obtido com triângulos, que era composto de 16 regras na base de conhecimento. O modelo obtido pode ser visto na figura 6.19.

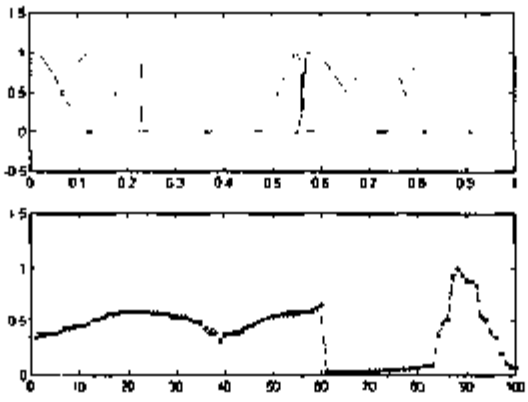


Figura 6.19 - Modelo com Splines da curva (5.19).

6.7.3 - Exemplo 10

O benchmark de Box & Jenkis quando aproximado com Splines leva ao modelo observado na figura 6.20. A comparação com o modelo obtido com triângulos é feita na tabela 6.6. Os resultados aqui não podem ser considerados como significativamente diferentes. Na verdade, a aproximação com triângulos como funções de base levou a um modelo com melhores características de acuidade do que o modelo obtido com as splines. Neste teste, o valor de λ foi fixado em 0,001. Valores maiores, quando testados, levaram a modelos mais complexos.

Tabela 6.6 - Comparação de Box & Jenkis com triângulos e Splines.

Funções	Regras	Erro Med. Quad.
Triângulos	90	0,042
Splines	90	0,040
Triângulos	49	0,096
Splines	49	0,098

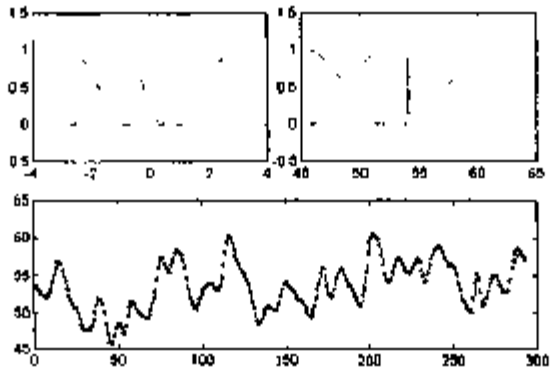
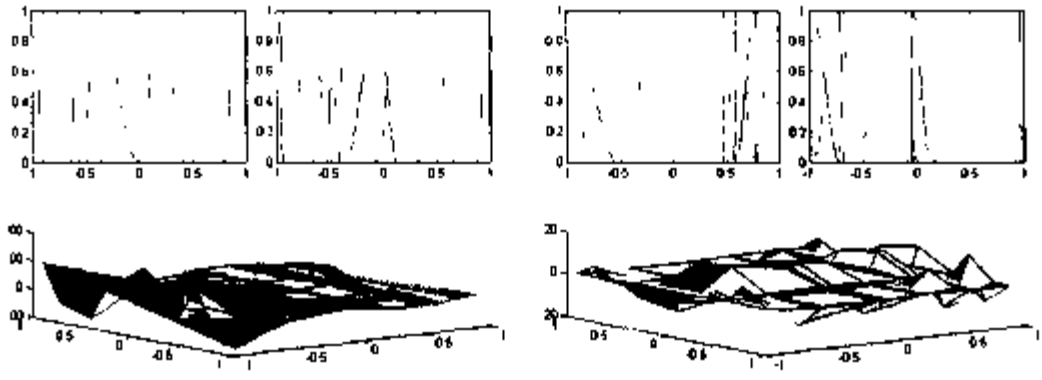


Figura 6.20 - O *benchmark* de Box & Jenkis modelado com Splines.

6.7.4 - Exemplo 11

Neste exemplo foram construídos os modelos neuro-fuzzy que modelam as superfícies definidas pelas equações $g_1(x,y)$, $g_2(x,y)$ e $g_3(x,y)$ Os modelos obtidos e as superfícies construídas estão esquematizados nas figuras 6.21 (a), (b) e 6.22, respectivamente. As tolerâncias especificadas são as mesmas do exemplo 5 deste capítulo. Para $g_1(x,y)$ foi construído um modelo de 196 regras cujo EMQ é de 0,5782. A superfície $g_2(x,y)$ foi aproximada por um modelo de 121 regras com EMQ = 0,8154. O modelo para a curva $g_3(x,y)$ é também composto de 121 regras que aproximam a superfície com um EMQ de 0,1757. As Splines modelaram melhor $g_1(x,y)$ mas tiveram um desempenho inferior para as duas outras superfícies.



(a) (b)

Figura 6.21 - Modelos obtidos e superfícies construídas para

(a) $g_1(x,y)$ (b) $g_2(x,y)$

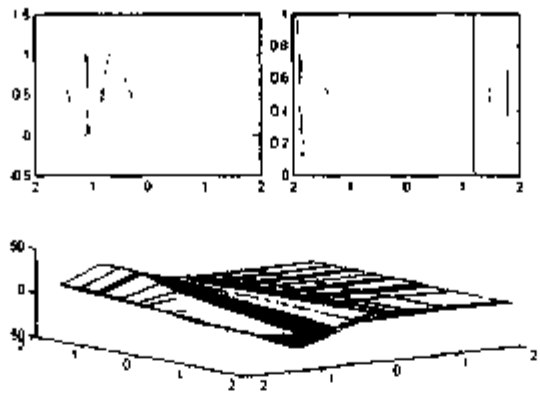


Figura 6.22 – Modelo obtido para $g_3(x,y)$

6.8 - Considerações

Neste capítulo apresentamos um método de ajuste das funções de pertinência como uma tentativa de minimizar a base de regras de modelos neuro-nebulosos. Os testes com triângulos e sua comparação com um método de controle (EVSUKOFF et al., 2000) mostram a validade da proposta apresentada.

A DVS provou mais uma vez ser um método robusto para o cálculo dos parâmetros conseqüentes das regras. Os ajustes nas partes antecedentes, mantendo-se o particionamento da unidade entre as funções de pertinência, levou a uma boa interpretabilidade dos modelos atingidos. As Splines cúbicas mostraram-se perfeitamente compatíveis com a metodologia de identificação proposta, levando a bases de regras ainda mais compactas em vários casos.

No próximo capítulo, aplica-se a metodologia ao controle do pressurizador da usina de Angra-1 , apontando-se os resultados obtidos. Também faz-se a modelagem de concentrações de elementos no núcleo de um reator nuclear. A seguir, as conclusões sobre o trabalho desenvolvido e sugestões para futuras pesquisas são apresentadas.

Resultados e conclusões

7.1 - Aplicação no controle do pressurizador de uma usina PWR

O pressurizador do circuito primário de uma usina PWR desempenha papel fundamental no funcionamento de toda a usina pois controla a pressão dentro do reator mantendo-a dentro dos limites de funcionamento especificados durante a operação normal e limita as variações de pressão durante os transientes que possam ocorrer na usina (ALVES, 1993). A ligação do pressurizador ao sistema de água primário de uma usina PWR é mostrada na figura 7.1.

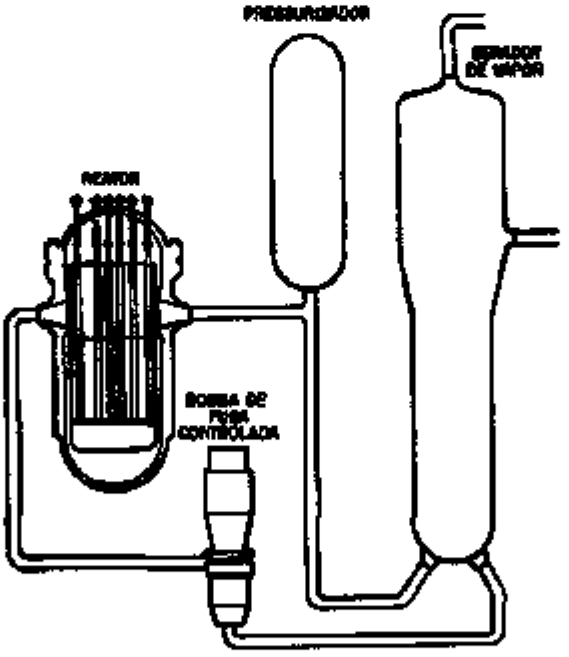


Figura 7.1 - Pressurizador no circuito primário de uma usina PWR.

Fundamentalmente, o equipamento consiste de um vaso com uma cabeça superior hemisférica e a parte inferior plana. Existem no pressurizador, em sua parte

inferior, aquecedores de imersão cambiáveis e, na parte superior, um bocal aspergidor de água fria. O controle da pressão do circuito primário é feito sobre a pressão do volume de vapor dentro do pressurizador aumentando-a ou diminuindo-a conforme seja necessário. Um diagrama esquemático do pressurizador e sua ligação ao circuito primário são mostrados na figura 7.2.

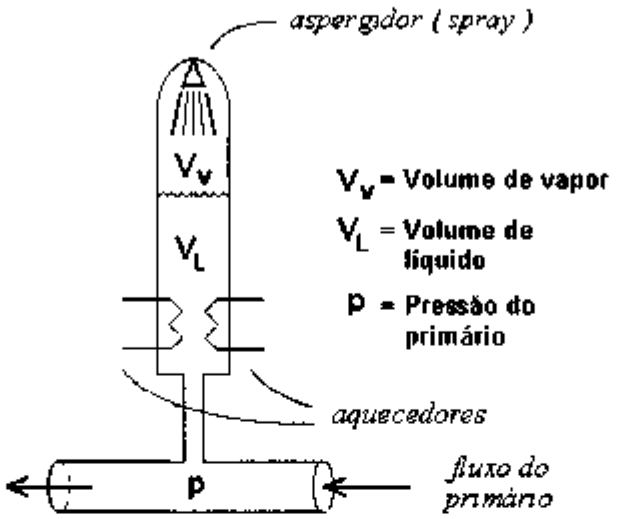


Figura 7.2 - Diagrama esquemático de um pressurizador de usina PWR.

Durante as condições normais de operação, cerca de 60% do volume do pressurizador é ocupado por água e os restantes 40% por vapor. Os aquecedores de imersão mantêm a água na temperatura de saturação mantendo-se, assim, constante a pressão no sistema. Para a usina de Angra-1, a pressão nominal do circuito primário é de 157,2 Kg/cm^2 , com os aquecedores atuando a aproximadamente 47,6% de sua capacidade. Se ocorrer um aumento na demanda de carga no sistema, isso significa maior demanda de vapor na turbina. Esse fato vai implicar em maior troca de calor com o refrigerante do circuito primário que, por conseqüência, tem sua temperatura média diminuída e, também, o seu volume. O refrigerante tende a escoar do pressurizador para o circuito reduzindo assim, o nível e a pressão dentro do equipamento. Contudo, essa redução de pressão é auto-limitante, pois provoca a formação de vapor, que limita a redução da pressão.

Em todo caso, essa redução da pressão provoca o fechamento de contatos nos aquecedores e a conseqüente elevação de temperatura tende a aumentar a pressão do sistema, equilibrando-o. No caso contrário, quando há uma diminuição na demanda de carga, o refrigerante tem a sua temperatura média aumentada e, também, o seu volume. Isso aumenta a pressão do sistema e faz atuar válvulas na linha do aspergidor que, jogando água da perna fria na região de vapor, provoca uma condensação do mesmo, diminuindo a pressão do circuito.

Vemos que também o próprio pressurizador necessita ser controlado. Caso a pressão do circuito primário aumente além dos níveis normais de funcionamento, o aspergidor entra em ação aumentando gradativamente sua atuação até 100 % de sua capacidade. No outro extremo, quando a pressão cai muito, são os aquecedores que começam a ser solicitados, também aumentando sua atuação conforme a demanda. A atuação de um e de outro dispositivo é feita conforme é esquematizada na figura 7.3.

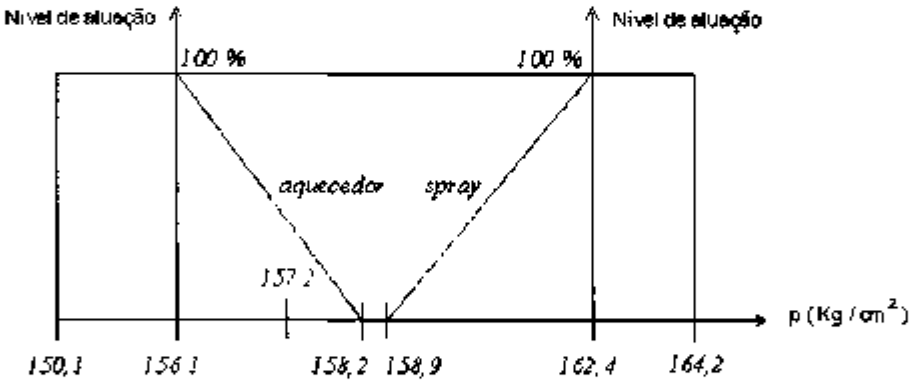


Figura 7.3 - Níveis de atuação, em função da pressão, do aspergidor (spray) e dos aquecedores.

Ainda que essa atuação seja feita de forma automática, ela não é normalmente suave, uma vez que tanto o funcionamento dos aquecedores como das válvulas do aspergidor são controlados pela própria pressão dentro do circuito primário do reator. Uma atuação por controle externo pode ter várias vantagens sobre a convencional e um sistema que proceda a essa atuação de forma precisa vem a ser de inegável utilidade para a operação da usina.

Um especialista poderia implementar um modelo neuro-nebuloso do controle do pressurizador com regras do tipo de (5.1), que nos levaria (provavelmente) à base de regras

$$
\begin{array}{llll}
\underline{\text{SE}} & p \text{ é baixa} & \underline{\text{ENTÃO}} & \hat{y} = \theta_1 \\
\underline{\text{SE}} & p \text{ é normal} & \underline{\text{ENTÃO}} & \hat{y} = \theta_2 \\
\underline{\text{SE}} & p \text{ é alta} & \underline{\text{ENTÃO}} & \hat{y} = \theta_3 \\
\underline{\text{SE}} & p \text{ é muito alta} & \underline{\text{ENTÃO}} & \hat{y} = \theta_4
\end{array} \qquad (7.1)
$$

que está esquematizada na figura 7.4.

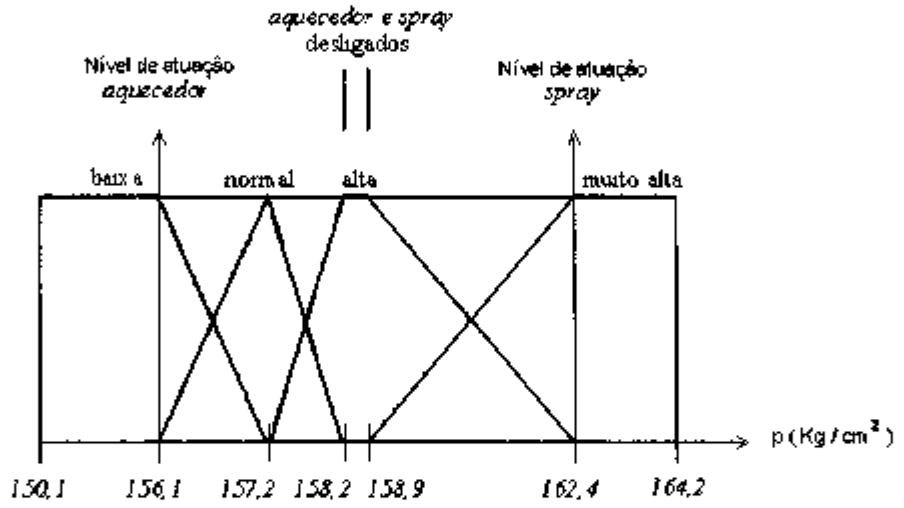


Figura 7.4 - Modelo possível de ser implementado por um especialista.

Essa base não só é válida como também representa exatamente o comportamento do sistema. Entretanto, ela tem algumas desvantagens. A mais evidente é o uso de um trapézio para representar a partição nebulosa "pressão alta". Apesar de lingüisticamente correta, a partição definida pelo trapézio não mantém a homogeneidade entre o formato das funções de pertinência (triângulos e trapézios misturados). Outra desvantagem é que as posições das partições nebulosas definidas pelo especialista podem não ser as mais adequadas para modelar o sistema.

Podemos obter uma base de regras do tipo de (7.1) com o uso apenas de

triângulos como funções de pertinência (fig. 7.5). Essa base de regras é mais homogênea, pois agora temos apenas triângulos definindo os conjuntos nebulosos. Infelizmente, nesse caso, a base de regras sofre a inclusão de mais uma regra, aumentando a estrutura do modelo.

Um outro senão é que ainda não temos certeza se as partições nebulosas estritamente triangulares da figura 7.5 definidas pelo especialista são as mais adequadas para a modelagem do sistema. Por exemplo, as partições "baixa" e "normal" parecem inadequadas, pois a "baixa" é, na verdade, a partição de operação nominal do pressurizador. Por outro lado, para rebatiza-la de "normal", teriamos então, a segui-la, as partições rebatizadas de "alta", "muito alta" e "extremamente alta" ; uma classificação, pelo menos, inadequada.

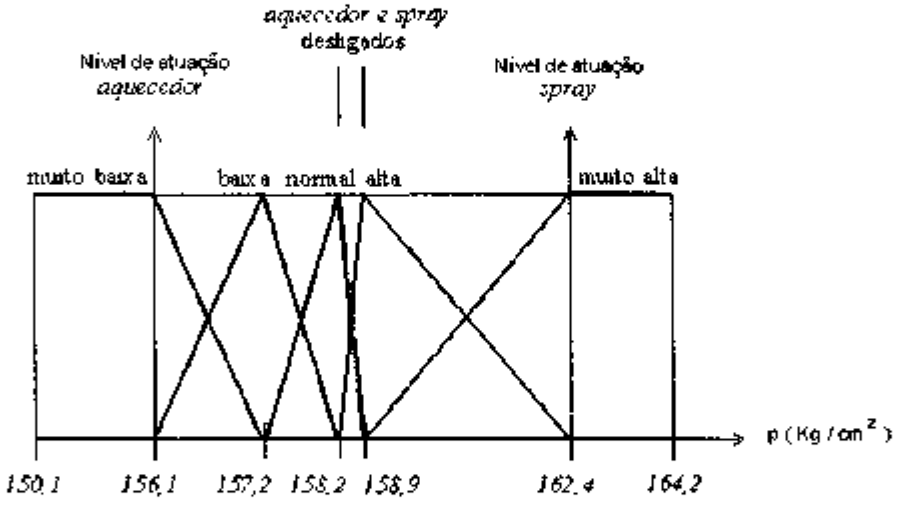


Figura 7.5 - Modelo implementado por um especialista apenas com triângulos.

Naturalmente, o especialista poderia eliminar algum dos conjuntos nebulosos e redefinir as partições, mas isso seria segundo um critério particular, que talvez não fosse o mais adequado à perfeita modelagem do problema.

Essa dificuldade na identificação de um modelo neuro-nebuloso adequado e de base mínima, lingüisticamente interpretável e de funções de base homogêneas foi a motivação para esta tese. Usando-se o método de um modelo expansível auto-ajustável

desenvolvido no capitulo anterior procedeu-se à modelagem de um sistema de controle neuro-nebuloso para o pressurizador de Angra-1. Usando-se os triângulos como funções de base, obteve-se o modelo apresentado na figura 7.6. O EMQ para o modelo obtido foi de 0,0001 (uma parte em 10.000). Para o treinamento, foram usados 101 pontos aleatórios tomados no intervalo [156,1 ; 162,4]. Por sua natureza aleatória, o extremo inferior acabou não sendo incluído e o intervalo usado foi [156,1371 ; 162,4].

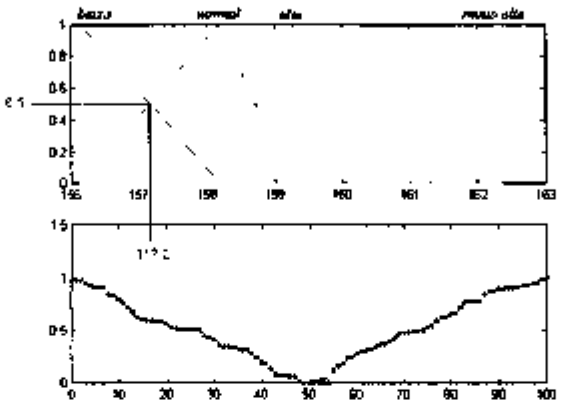


Figura 7.6 - Modelo obtido para o pressurizador.

Pode-se verificar que apenas 4 regras foram necessárias para a obtenção de um modelo plenamente adequado, com ótima interpretação lingüística e excelente acuidade. Os pontos modais correspondem às posições 156,1371 ; 158,1781 ; 159,3992 e 162,4 ; correspondentes, respectivamente, aos conjuntos nebulosos "baixa" , "normal", "alta" e "muito alta".

A base de regras final obtida foi:

$$\begin{array}{llll} \underline{\text{SE}} & p \text{ é baixa} & \underline{\text{ENTÃO}} & \hat{y} = 0{,}9884 \\ \underline{\text{SE}} & p \text{ é normal} & \underline{\text{ENTÃO}} & \hat{y} = -0{,}0017 \\ \underline{\text{SE}} & p \text{ é alta} & \underline{\text{ENTÃO}} & \hat{y} = 0{,}1363 \\ \underline{\text{SE}} & p \text{ é muito alta} & \underline{\text{ENTÃO}} & \hat{y} = 1{,}0024 \end{array} \qquad (7.2)$$

O ponto de operação normal com pressão do primário igual a 157.2 Kg/cm^2 corta as curvas de "pressão baixa" e de "pressão normal" aproximadamente na metade de suas alturas, segundo a figura 7.6. Isso dá uma atuação, no visual, de 49,3% para os aquecedores, um desvio de menos de 4%. Aplicando-se verdadeiramente o ponto de 157,2 Kg/cm^2 às funções de pertinência do modelo, obtemos as pertinências de 0,4792 e 0,5208 para "pressão baixa" e "pressão normal", respectivamente. Usando-se (6.12) para calcular a saída, obtemos 47,7% de atuação para os aquecedores, uma acuidade de 99,8%.

A modelagem obtida com as Splines é apresentada na figura 7.7. Nesse caso, foi necessária a inclusão de mais uma partição nebulosa, um modelo inferior ao conseguido com os triângulos. Isso foi devido à alta linearidade (por partes) do sistema sob modelagem, sendo, então, inadequado à modelagem por curvas não lineares como as Splines.

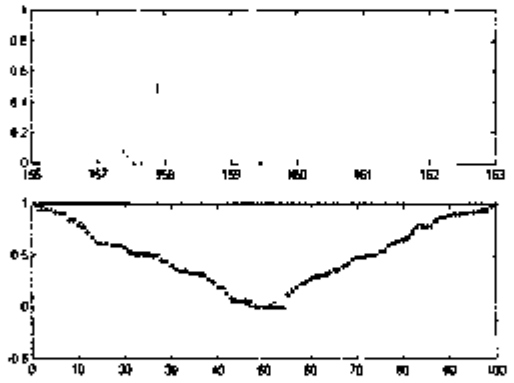


Figura 7.7 - Modelo do controle do pressurizador com Splines.

7.2 - Modelagem da oscilação espacial do fluxo de nêutrons

A modelagem do fluxo espacial de nêutrons no interior de um reator nuclear como uma função das concentrações de xenônio e iodo pode ser interessante no sentido de se construir um sistema de controle para a potência do reator.

Usou-se aqui os dados obtidos com uma abordagem simplificada (DA RUAN,

2000) que podem ser visualizados na figura 7.8, onde se empregam as equações de balanço não lineares para o xenônio e o iodo e a equação de difusão de nêutrons unidimensional a um grupo e as distribuições do fluxo, do xenônio e do iodo são assumidas serem séries harmônicas espaciais a dois termos. O modelo do reator é estimado estar tão crítico quanto possível pelo uso de uma estimativa variacional do autovalor da equação de difusão. A potência total do reator é suposta constante ainda que a densidade de potência varie em função da posição e do tempo. Nessa modelagem, o reator é dividido em duas partes iguais, a superior e a inferior.

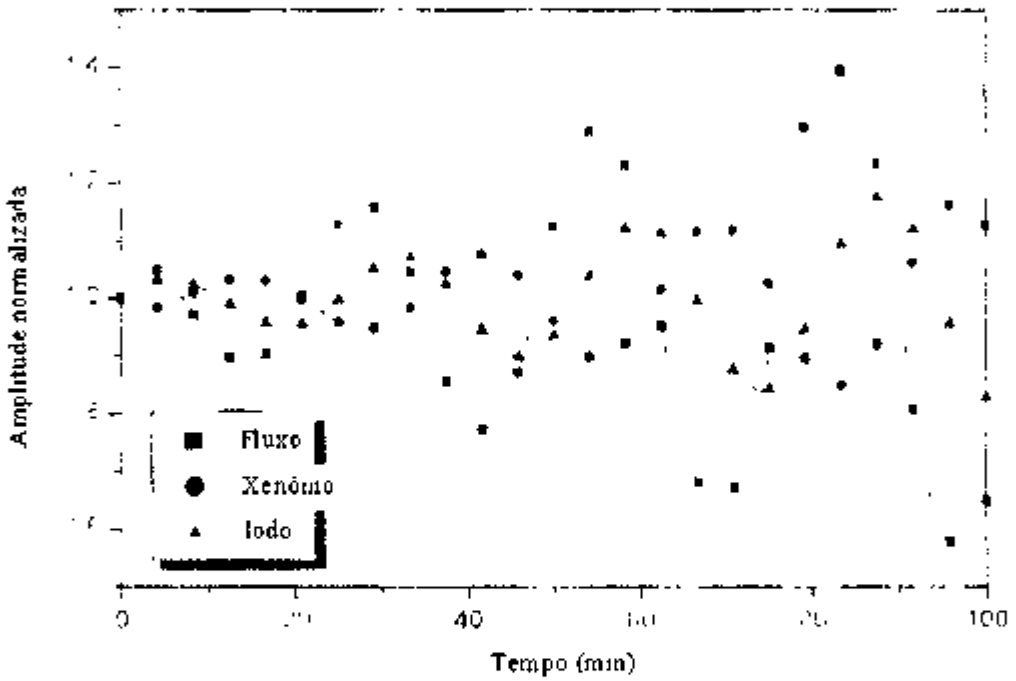
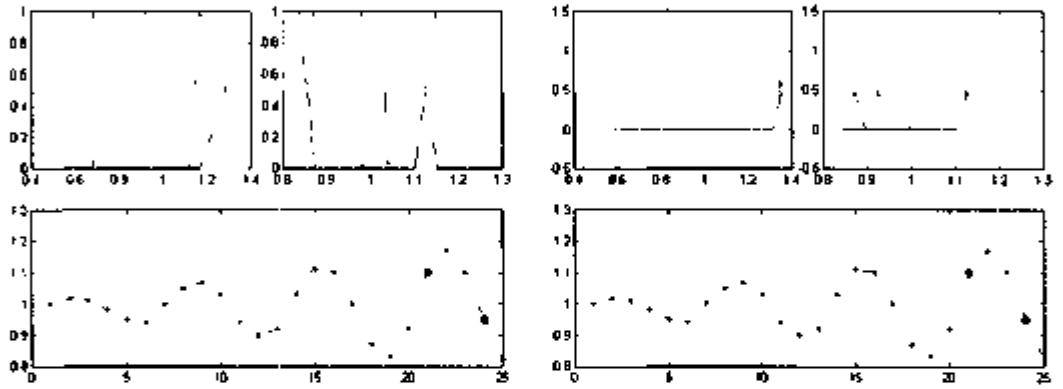


Figura 7.8 - Oscilações do fluxo de nêutrons, xenônio e iodo devido a uma mudança na reatividade.

O modelo simula o caso em que o reator está a 100% de potência com uma concentração de xenônio em estado estacionário antes que uma perturbação seja aplicada. Essa perturbação é uma mudança na combinação das seções macroscópicas de absorção do combustível, do moderador e do veneno queimável e é suposta durar 2,5 h. As oscilações normalizadas do fluxo e das concentrações de xenônio e iodo são mostradas na figura 7.8, de onde foram retirados os dados para o modelo neuro-fuzzy.

O objetivo é apresentar como a abordagem desenvolvida nesta tese pode ser usada para modelar as concentrações de elementos no interior de um reator, sem haver preocupação com o controle em si do fluxo de nêutrons. Os modelos obtidos pelo método do capítulo 6 com triângulos e Splines são visualizados nas figuras 7.9 (a) e (b). Na parte superior das figuras estão as partições nebulosas que modelam o fluxo (à esquerda) e o xenônio. A parte inferior é o comportamento da concentração de iodo em vista tanto do comportamento do fluxo de nêutrons como da concentração de xenônio. Ambos os modelos são compostos de 64 regras com um EMQ menor que 0,0001. Apenas para referência, o modelo de EVSUKOFF et al. (2000) necessita de 90 regras para atingir a mesma tolerância.



(a) (b)

Figura 7.9 - Modelagem da concentração de iodo (a) Triângulos (b) Splines.

7.3 - O valor de λ

A constante de aprendizagem, λ, é normalmente referenciada na literatura (JANG, 1992, MAEDA et al., 1991) como sendo, preferencialmente, um valor "pequeno", na faixa de 0,01 - 1,00. Caso o progresso do algoritmo de identificação (qualquer que seja o algoritmo) seja muito lento, o que se faz é aumentar esse valor, aumentando-se, assim, a velocidade de convergência. Caso se deseje um ajuste mais fino, faz-se reduzir o valor de λ para permitir uma pesquisa mais apurada no espaço de busca.

O espaço de busca é a superfície de erro definida por (3.3), uma função completamente desconhecida. Apesar de ser inidentificável de antemão, sabe-se que essa função é repleta de "picos" e "vales", que representam os pontos locais de erros máximos e mínimos. Acredita-se que, em geral, essas funções possuem pontos de máximo e mínimo globais e a busca por esse ponto de mínimo global é o grande objetivo de qualquer processo de identificação. Contudo, como não se conhece a localização desse ponto, é praxe aceitar-se a identificação de um ponto de mínimo local, desde que ele realmente apresente características de mínimo global. A característica mais visível é o aumento do gradiente de erro em qualquer direção mesmo para incrementos muito pequenos na caminhada sobre o espaço de busca.

O aumento arbitrário de λ tanto pode levar o algoritmo a "oscilar" em torno de um ponto de baixo erro como faze-lo "pular" um ponto de erro mínimo, enquanto que valores muito pequenos podem levar o algoritmo a ficar preso em algum ponto de mínimo local inadequado, próximo a pontos de mínimo locais (pelo menos um) de valores mais baixos.

Durante o trabalho de pesquisa desta tese, foi verificado experimentalmente que determinados valores para λ permitiam à modelagem uma convergência mais rápida, consistente e precisa do que outros. A investigação desses valores mostrou que, de uma maneira geral, eles poderiam ser relacionados, numa dada iteração k, ao EMQ dos N pontos de treinamento e à quantidade de partições nebulosas p segundo

$$\lambda_k = \frac{J_{\lambda_k}}{p_{m_k}} \tag{7.3}$$

onde:

J_{λ_k} = Valor do EMQ (J) do conjunto de pontos N na iteração k

p_{m_k} = número de partições nebulosas (p) da dimensão m na iteração k

Sendo vinculado às partições nebulosas da dimensão considerada, λ é, em geral, normalmente diferente para cada dimensão do problema. Quanto maior o EMQ dos

dados de treinamento, maior o valor de λ, aumentando a velocidade de convergência do modelo. À medida que o número de partições nebulosas cresce e o EMQ do modelo diminui, λ decresce proporcionalmente, melhorando o ajuste fino da busca dos pontos de mínimo erro.

Foi experimentada a relação empírica (7.3) em alguns problemas. Por exemplo, sua aplicação à modelagem da função $g_1(x,y)$ aumentou bastante a velocidade de convergência do algoritmo e o modelo obtido foi praticamente idêntico ao original. Essa curva em particular tem uma alta amplitude de saída, que leva a valores elevados do EMQ nas primeiras iterações. Ajustes muito pequenos nos c_i (6.9) praticamente não diminuem o EMQ. À medida que mais e mais partições eram acrescidas e o EMQ diminuía, o valor de λ caía proporcionalmente, refinando a busca dos mínimos na superfície de erros.

7.4 - Conclusões

A necessidade de melhores métodos de controle foi a motivação para a busca de uma modelagem em forma de sistemas neuro-nebulosos expansíveis e auto-ajustáveis. Tomando como exemplo o controle de um pressurizador de usina PWR, demonstrou-se como a identificação por um especialista pode ser, apesar de correta, inadequada.

Foi apresentado um novo método para a identificação da estrutura e parâmetros de modelos nebulosos de sistemas lineares e não-lineares. Este método baseia-se no método de descida do gradiente e muda as posições dos protótipos dos conjuntos nebulosos de entrada de forma a minimizar uma função de erro médio quadrático. Pode ser visto como uma evolução tanto dos modelos expansíveis do capítulo 5 como do Anfis. Nossa preferência recai nessa última visão, em vista da abordagem neuro-fuzzy que motivou e permeou o desenvolvimento desta tese. Os testes em várias funções uni e bidimensionais provaram que o método é adequado no sentido de levar a modelos mais compactos em termos de estrutura e com boas capacidades de generalização e precisão.

Com isso, não apenas o objetivo primordial de um sistema de controle foi atingido, mas também verificou-se que a metodologia aqui apresentada pode ser perfeitamente adequada a problemas de diagnóstico e modelagem de sistemas complexos.

Deve-se anotar que em modelos de sistemas onde a limitação de conjuntos nebulosos nas partições não seja uma imposição, a abordagem de EVSUKOFF et al. (2000) pode trazer benefícios. Por exemplo, em pontos de dados definidos em grade regular, viu-se que ela pode ser bem eficiente, pois ela atribui conjuntos nebulosos exatamente nos pontos de maior erro da aproximação e não os move. Isso é benéfico no caso em que os pontos mais contribuintes para o EMQ também estão regularmente distribuídos na grade.

Verificou-se como as formas das funções de pertinência podem afetar a interpretabilidade dos modelos construídos. É conveniente que mais de uma parametrização seja testada em um determinado problema. Por exemplo, em sistemas com altas taxas de não linearidades, as Splines podem ser uma opção melhor do que os triângulos. Elas não são, contudo, uma panacéia e em sistemas com altos graus de linearidades (ainda que por partes), os triângulos podem ser as funções de base de maior eficiência.

Apresentou-se uma relação empírica para a definição da *constante de aprendizagem* λ , como uma função dos dados de treinamento envolvidos. Foi mostrado que essa relação pode ser usada para a definição e ajuste dos valores de λ, ao invés da forma mais comumente usada de tentativa-e-erro.

7.5 - **Futuras pesquisas**

As bases de regras obtidas para os diversos modelos não podem ser asseguradas como sendo as bases mínimas que podem ser obtidas. Da mesma forma, não se garante que as parametrizações aqui usadas (triângulos e splines cúbicas) sejam as melhores. KOSKO (1996) faz uma interessante consideração sobre várias parametrizações

utilizáveis. Segundo esse estudo, a curva *sinc* pode aproximar várias funções com boa confiabilidade. Uma investigação maior sobre essa e outras parametrizações pode ser interessante.

Por exemplo, no início deste trabalho de tese, cogitou-se sobre o uso de gaussianas cujos expoentes fossem modificados e ajustados dinamicamente, fazendo variar a variância das funções. Algum trabalho foi feito mas as Splines pareceram ser tão promissoras que atraíram nosso interesse. Um estudo sobre as gaussianas de variância modificável pode ser de interesse.

No capítulo 3, falou-se, com alguns poucos detalhes, das B-Splines. Uma pesquisa que poderia ser de bastante valor seria a investigação do ajuste de tais funções para implementar os modelos neuro-nebulosos. Esse ajuste pode ser feito de duas formas, a nosso ver: pelo ajuste e adaptação das posições dos nós das B-Splines e pela própria mudança na ordem das funções, aumentando-se, algoritmicamente, o grau das curvas para a melhor modelagem.

A redução da base de regras também resultaria num estudo bem atual. Após a convergência dos modelos, métodos estatísticos podem ser empregados tanto para compactar a base de regras atingida como para suprimir eventuais partições nebulosas que contribuam pouco para a acuidade e a generalização do modelo. Uma outra linha de pesquisa é o uso de técnicas evolucionárias na expansão da estrutura do modelo. Novos algoritmos de busca vem sendo desenvolvidos e seu uso na pesquisa dos pontos de inserção de novos conjuntos nebulosos é uma abordagem a ser investigada.

A aplicação da metodologia de ajuste aqui apresentada não chegou a ser testada em modelos TSK de ordem 1. Um bom trabalho seria essa aplicação e sua comparação com os resultados aqui obtidos.

A expressão empírica apresentada para a escolha de λ, longe de ser apresentada como verdade indiscutível, é proposta mais como um guia, devendo ser testada e, se necessário, aperfeiçoada.

Referências bibliográficas

[1] Alves, A.C.P.D., *Um Sistema de Análise de "Trip" em Reatores PWR usando Redes Neuronais*. Tese de Mestrado, COPPE / UFRJ, Rio de Janeiro, 1993.

[2] Anonimous, *Final Safety Analisys Report for Angra-1 Nuclear Power Plant*. 1 ed. Westinghouse Eletric Corporation, Rio de Janeiro, 1979.

[3] Bossley, K.M., *Neurofuzzy Modelling Approaches in System Identification*. Ph. D. dissertation, University of Southampton, Southampton, England, 1997.

[4] Box, G.E.P., Jenkis,G.M., *Time Series analysis: Forecasting and Control*. 1 ed. San Francisco, Holden Day, 1970.

[5] Chapra, S.C., Canale, R.P., *Numerical Methods for Engineers*. 2 ed. New York, McGraw-Hill, 1988.

[6] Combs, W.E., Andrews, J.E., "Combinatorial explosion eliminated by a fuzzy rule configuration", *IEEE Transactions on Fuzzy Systems* v. 6, n. 1, pp.1-11, 1998.

[7] Da Ruan, "Fuzzy Systems and Soft Computing in Nuclear Engineering". In: *Studies in Fuzziness and Soft Computing*, v. 38. New York, Physica-Verlag, 2000.

[8] Dubois, D., Prade, H., *Fuzzy Sets and Systems*. 1 ed. New York, Academic Press, 1980.

[9] Evsukoff, A., *Approximate Reasoning for Process Supervision*. Ph.D. dissertation, Institut National Polytechnique de Grenoble, Grenoble, Switzerland, 1998.

[10] Evsukoff, A., Branco, A.C.S., Galichet, S. *Structure Identification and Parameter Optimization for Non-Linear Fuzzy Modelling*. (inédito) Submetido à revista Fuzzy Sets and Systems, 2000.

[11] Eyckhoff, P., *System Identification Parameters and State Estimation* 1 ed New York, John Willey & Sons, 1974

[12] Foulloy, L., Galichet, S., "Typology of fuzzy controlers" In *Theoretical Aspects of Fuzzy Control* v 1, New York, John Willey Publishers, pp 174-186, 1995

[13] Golub, G H., Van Loan, C F., *Matrix Computations* 1 ed Baltimore, John Hopkins, 1989

[14] Haykin, S., *Neural Networks A comprehensive foundation* 1 ed New York, Macmillan, 1994

[15] Jang, J S R., *Neuro-Fuzzy Modeling Architectures Analyses and Applications* Ph D dissertation, University of California at Berkeley, Berkeley, California, 1992

[16] Jang J S R Sun, C T., Mizutani, E., *Neuro-fuzzy and Soft Computing - A Computational Approach to Learning and Machine Intelligence* 1 ed New York, Prentice-Hall, 1997

[17] Juditsky, A., Zhang Q., Delyon, B et al *Wavelets in Identification* In INRIA Research Report 2315, INRA, 1994

[18] Kosko, B., *Neural Networks and Fuzzy Systems* 1 ed New Jersey, Prentice Hall, 1992

[19] Kosko, B., *Fuzzy Engineering* 1 ed New Jersey, Prentice Hall, 1997

[20] Kosko, B., Mitaim, S., "What is the Best Shape for a Fuzzy Set in Function Approximation?", *IEEE Transactions on Fuzzy Systems* v 6, n 1, pp 1237-1243, 1996

[21] Lawson, C F . Hanson, R J . *Solving Least Squares Problems* 1 ed. New Jersey, Prentice Hall. 1974

[22] Ljung, L., *System Identification - Theory for the user* 1 ed. New Jersey, Prentice Hall, 1987.

[23] Maeda, A., Funabashi, M., "Fuzzy and Neural Hybrid Expert Systems: Synergetic AI". In: *IEEE Expert*, pp. 32-40, California, IEEE Computer Society Press, 1995.

[24] Maeda, A., Ichimori, T., Funabashi, M., "Flip-net: A Network Representation of Fuzzy Inference Procedure and its Application to Fuzzy Rule Structure Analysis". In: *Proceedings of the Second IEEE International Conference on Fuzzy Systems*, pp.391-395. California, IEEE Computer Society Press, 1993.

[25] Maeda, A., Someya, R. Funabashi, M., "A Self-Tuning Algorithm for Fuzzy Membership Functions using Computational Flow Network". In: *Proceedings of IFSA'91*, pp. 129-132. Brussels, 1991.

[26] Maeda, A., Someya, R et al. "A Fuzzy-based Expert System Building Tool with Self-Tuning Capability for Membership Functions". In: *Proceedings of the World Congress on Expert Systems*, pp. 639-647, New York, Pergamon Press, 1991.

[27] Nakoula, Y., *Apprentissage des Modeles Linguistiques flous, par Jeu de Regles Ponderées* Ph.D. dissertation, Université de Savoie, França, 1997.

[28] Nakoula, Y., Galichet, S., Foulloy, L., "Simultaneous learning of rules and linguistic terms". In: *Proceedings of the International Conference on Fuzzy Systems*, pp. 237-253, New Orleans, 1996.

[29] Nakoula, Y., Galichet, S., Foulloy, L., "A learning method for structure and parameter identification of fuzzy linguistic models". In: *Selected Approaches to Fuzzy Model Identification*, pp. 281-319, 1997.

[30] Narendra, K.S., Parthasarathy, K., "Identification and control of dynamical systems using neural networks", *IEEE Transactions on Neural Networks* v. 1, n. 1, pp. 4-26, 1990.

[31] Passino, K.M., Yurkovich, S., *Fuzzy Control*. 1ed. Menlo Park, California, Addison Wesley Longman, Inc., 1998.

[32] Pedricz, W., "An identification algorithm in fuzzy relational systems". In: *Fuzzy Sets and Systems*, v. 13, pp. 156-167, 1983.

[33] Pedrycz, W., *Fuzzy Control and Fuzzy Systems*. 2 ed. New York, John Willey & Sons, 1993.

[34] Powell, M.J.D., *Approximation theory and methods*. 1 ed. New York, Press Syndicate of the University of Cambridge, 1981.

[35] Rice, J. R., *Numerical Methods, Software and Analysis*. 1 ed. New York, Mc Graw-Hill, 1983.

[36] Sugeno, M., Kang G.T., "Structure identification of fuzzy models". In: *Fuzzy Sets and Systems*, v. 42, pp. 15-33, 1988.

[37] Sugeno, M., Asai, K., Terano, T., *Fuzzy Systems Theory and its applications*. 1 ed. California, Academic Press, 1992.

[38] Sugeno, M., Yasukawa, T., "A fuzzy-logic-based approach to qualitative modeling", *IEEE Transactions on Fuzzy Systems*, v. 1, n. 1, pp. 7-31, 1993.

[39] Takagi, T., Sugeno, M., "Fuzzy identification of systems and its application to modeling and control", *IEEE Transactions on Systems, Man and Cybernetics* v. 15, n. 1, pp. 116-132, 1985.

[40] Tong, R.M., "The evaluation of fuzzy models derived from experimental data". In: *Fuzzy Sets and Systems*, v. 4, pp. 1-12, 1980.

[41] Wang, L.X., *Adaptive Fuzzy Systems and Control: Design and Stability Analysis*. 1 ed. New Jersey, Prentice-Hall, 1994.

[42] Wang, L.X., Langari, R., "Complex systems modeling via fuzzy logic", *IEEE Transactions on Systems, Man and Cybernetics*, v. 26, n. 1, pp. 100-106, 1996.

[43] Willaeys, D., Malvache, N., "The use of fuzzy sets for the treatment of fuzzy information by computer". In: *Fuzzy Sets and Systems*, v. 5, pp. 323-327, 1981.

[44] Xu, C.W., Lu, Y.Z., "Fuzzy model identification and self learning for dynamic systems", *IEEE Transactions on Systems, Man and Cybernetics*, v. 17, n. 4, pp. 683-689, 1987.

[45] Yager, R.R., Filev, D.P., "Unified structure and parameters identification of fuzzy models", *IEEE Transactions on Systems, Man and Cybernetics*, v. 23, n. 4, pp.1198-1205, 1993.

[46] Yager, R.R., Filev, D.P., *Essentials of Fuzzy Modeling and Control*. 1 ed. New York, John Willey & Sons, 1994.

[47] Yager, R.R., "On the construction of hierarchical fuzzy systems models", *IEEE Transactions on Systems, Man and Cybernetics*, v. 28, n. 1, part C, pp. 55-66, 1998.

[48] Yen, J., Wang, L., "Simplifying fuzzy rule-based models using orthogonal transformation methods", *IEEE Transactions on Systems, Man and Cybernetics*, v 23, n 1, part B, pp. 13-24, 1999.

[49] Yi, S.Y., Chung, M.J., "Identification of fuzzy relational model and its application to control". In: *Fuzzy Sets ans Systems*, v. 59, pp.25-33, 1993.

[50] Zadeh, L., "Fuzzy Sets", *Information and Control*, v.8, pp. 338-353, 1965.

[51] Zadeh, L., "Fuzzy logic = computing with words", *IEEE Transactions on Fuzzy Systems*, v. 4, n. 2, pp. 103-111, 1996.

[52] Zimmermann, H J., *Fuzzy Set Theory and its applications* 2 ed Massachussets, Kluwer Academic Publishers, 1994.