UNIVERSIDADE ESTADUAL DE CAMPINAS

FACULDADE DE ENGENHARIA ELÉTRICA

DEPARTAMENTO DE ENGENHARIA DE COMPUTAÇÃO E AUTOMAÇÃO INDUSTRIAL

COMPRESSÃO E DESCOMPRESSÃO DE DADOS ATRAVÉS DE REDES NEURAIS E LÓGICA NEBULOSA COM APLICAÇÃO EM CURVAS PLANAS

Myriam Regattieri De Biase da Silva

Orientador : Prof. Dr. Márcio Luiz de Andrade Netto

Co-orientador : Prof. Dr. Armando Freitas da Rocha

Tese submetida à Faculdade de Engenharia Elétrica da Universidade Estadual de Campinas, para preenchimento dos pré-requisitos para obtenção do Título de Mestre em Engenharia Elétrica.

Agradeço :

Ao prof. Márcio pela oportunidade, orientação, paciência e calma nos momentos decisivos.

Ao prof. Armando pela co-orientação, pela força e coragem transmitidas ao se lançar constantemente aos novos desafios.

Ao prof. Petrônio Pulino do IMECC pelos empréstimos dos livros e por tanta boa vontade.

Ao amigo Fernando Von Zuben pelas lições diárias de competência e humildade; pelas discussões, excelentes sugestões e enorme contribuição.

Aos amigos Sérgio e Celeste por todo carinho com que me receberam e apoio até o final.

À amiga Olga pela grande amizade, fortalecida por uma convivência que resultou em crescimento, preocupação e respeito mútuo.

Às amigas Rita e Kathya que apesar da distância souberam se fazer muito presentes.

À amiga Blanca que dividiu muito mais que um espaço destinado à sala de estudos. Agradeço pela força nas horas difíceis e alegria compartilhada quando dos sucessos obtidos.

Aos amigos Ely, Humberto, Taninha, Fofi e Juju pelo apoio, carinho e constatação de que a busca pelo crescimento profissional requer uma realização pessoal que envolve, necessariamente, a descoberta de novas amizades.

Dedico este trabalho aos meus pais Walter e Anastasia, irmãos Waltinho e Wania e ao meu namorado Armando por todo amor, carinho e compreensão.

Resumo

Este trabalho propõe um método de compressão-descompressão de dados aplicado a curvas no espaço 2D. Novas técnicas como Redes Neurais e Lógica Nebulosa são usadas para comprimir e descomprimir os pontos de uma curva plana. No processo de compressão, estruturas de redes neurais são desenvolvidas para operarem conjuntos de pontos seqüenciais (x,y). O objetivo é extrair o mínimo de dados possíveis de modo a permitir a recuperação da curva original. Um método de interpolação baseado em regras nebulosas é proposto, introduzindo-se modificações no algoritmo original apresentado por Uchino *et al.*. Este algoritmo é utilizado para recuperar a informação, sempre que necessário. O algoritmo original (INL) - Interpolação Nebulosa Linear - funciona somente quando os pares de entrada saída (x,y) representam funções. O algoritmo proposto (INNL) - Interpolação Nebulosa Não Linear - introduz não linearidades no cálculo das funções de pertinência associadas às regras nebulosas, visando à obtenção de curvas interpoladas mais suaves. Uma outra modificação é introduzida no sentido de se generalizar a aplicação do método a curvas genéricas (curvas fechadas e não funções).

Abstract

This work proposes a method for compression-restoring data applied to bi-dimensional curves. New techniques as Neural Networks and Fuzzy Logic are used on compressing and restoring the points of the curve. On data compression, neural-like structures are developed to operate on a set of sequential points (x,y) in order to extract the minimal amount of data that can still accurately represent the entire original set. A method of interpolation based on fuzzy rules is proposed introducing modifications on the original algorithm presented by Uchino *et al.*, to regenerate the whole original set of data whenever necessary. The original algorithm based on linear fuzzy rules works only when the input-output relation (x,y) represent a function. The modified algorithm (INNL) makes use of non linear fuzzy rules introducing the nonlinearities to calculate the membership functions associated to the rules, whose purpose is to obtain smoother interpolated curves. Another modification is introduced to generalize the algorithm to any kind of curve (closed and non-functions).

Conteúdo

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

Sistemas Conexionistas e Métodos Seqüenciais para manipulação simbólica se originaram dos estudos de esquemas computacionais com inspiração biológica. A pouca eficiência dos sistemas de engenharia no tratamento de problemas de manipulação simbólica resultou num maior interesse em sistemas mais flexíveis, onde redes neurais e lógica nebulosa são alguns exemplos. Atualmente, estes sistemas de origem comum representam áreas distintas. As Redes Neurais podem ser vistas como um elo de ligação entre os sistemas baseados em Lógica Nebulosa e Métodos de Engenharia [Bar89]. No entanto, os avanços obtidos tanto na área dos sistemas conexionistas quanto dos métodos seqüenciais têm motivado a aplicação conjunta de redes neurais e lógica nebulosa em estudos de problemas de inferência [Tak90].

A determinação de relações entrada-saída de sistemas, cuja representação analítica não é conhecida, figura como uma área de grande interesse em sistemas de informação. Questões relativas à dimensão da base de dados são fundamentais na escolha do método de representação destas relações. Outro aspecto importante se refere à questão do erro cometido ao se recuperar a informação. O compromisso a ser mantido requer a recuperação dos dados originais, com um erro aceitável dependendo do tipo de aplicação, tendo como entrada o mínimo de informação possível.

Neste contexto, a idéia de se utilizarem métodos de compressão e descompressão de dados, baseados em técnicas de redes neurais e lógica nebulosa aparece como uma alternativa bastante interessante para o problema da representação dos sistemas de entrada-saída. Particularmente, a utilização destas técnicas na representação de curvas planas se mostra como uma solução viável e eficiente. Os pares de entrada-saída (x, y) podem ser comprimidos, obtendo-se assim, um conjunto de pontos significativos. A curva original pode então ser recuperada, na etapa de descompressão, através de um algoritmo de interpolação.

O processo de compressão consiste basicamente da definição de uma classe de padrões de entrada (seqüência de cinco pontos adjacentes) onde o segundo ou terceiro ponto, dependendo da classe, representa um ponto significativo.

Na etapa de descompressão é necessária a definição do método de interpolação a ser utilizado. O problema da interpolação estabelece uma infinidade de diferentes funções passando por um conjunto de pontos dados, onde a escolha de uma destas funções depende de alguns critérios como simplicidade e suavidade. A maioria das funções de interpolação é formada por combinações lineares de funções elementares. No caso da interpolação nebulosa, a função interpolante é obtida ponderando-se os valores das retas que unem os pontos dados, pelos valores das funções de pertinência dos conjuntos nebulosos associados.

A grande capacidade da teoria nebulosa em tratar problemas complexos viabiliza a aplicação desta técnica na estruturação da base de regras utilizada pelo método de interpolação. Já a alta flexibilidade das redes neurais em processos de classificação de padrões coloca estas últimas como uma alternativa para se solucionar o problema da determinação dos pontos significativos.

Foram propostos dois modelos para a solução do problema da compressão: modelo de rede neural para processamento simbólico e modelo de rede neural artificial para processamento numérico. Com base no método de interpolação proposto por Uchino *et al.* [UYY90], onde se utilizam regras lineares, desenvolveu-se uma nova base de regras nebulosas não lineares [RvZR93], para o algoritmo de interpolação.

O presente trabalho tem por objetivo propor soluções para os problemas de compressão (extração de pontos significativos) e descompressão (interpolação) dos dados correspondentes aos pares (x, y) de uma curva plana. Na etapa de interpolação, mostra-se que a simples introdução da não linearidade garante uma curva interpolada mais suave, com primeira derivada contínua. Uma outra modificação proposta é a alteração da saída interpolada, de forma que seja possível interpolar tanto funções, como curvas genéricas (não funções). Deste modo, deseja-se comprovar a eficiência do método comparado à proposta original, assim como, mostrar que a simplificação obtida ao se estruturar o método de interpolação por meio de regras nebulosas não implica em menor precisão, quando comparado aos métodos tradicionais. A etapa de compressão visa à definição da classe de padrões que representam os pontos significiativos e traz ainda soluções para a extração destes pontos a partir da seqüência de entrada dos pares (x, y).

Este trabalho visa à solução do problema da compressão e descompressão dos dados, com aplicação em curvas planas, e está estruturado do seguinte modo:

O capítulo 2 traz um embasamento teórico onde são apresentados, de forma sucinta, os conceitos básicos da teoria de lógica nebulosa e redes neurais. São abordados aspectos diversos da teoria de conjuntos nebulosos e lógica nebulosa, voltados para o tipo de raciocínio nebuloso que define a base de regras utilizadas no processo de interpolação. A teoria de redes neurais é dividida em duas partes: redes neurais artificiais e redes neurais para processamento simbólico. Alguns aspectos como estrutura, treinamento, capacidade de generalização são abordados na parte relativa à teoria de redes neurais artificiais. A teoria referente à rede neural para processsamento simbólico concentra-se no modelo do neurônio e tipo de processamento realizado por este.

No capítulo 3 são apresentados os métodos tradicionais de interpolação. A teoria de interpolação polinomial e interpolação polinomial por partes é aqui apresentada, com ênfase para os polinômios splines cúbicos por partes. Faz-se uma análise de erro para cada método descrito, uma vez que a análise de convergência é fundamental na teoria da interpolação.

O capítulo 4 descreve o método de interpolação nebulosa. Inicialmente são apresentados os conceitos básicos que definem o método original proposto por Uchino *et al.*, e a seguir, são mostradas as modificações propostas no sentido de se suavizar a curva interpolada. Justifica-se teoricamente a introdução da não linearidade como forma de se garantir a continuidade da primeira derivada. Posteriormente, é feita uma análise de convergência, com o objetivo de se garantir a eficiência do método quando novos pontos são inseridos. Por último, mostra-se a alteração que possibilita a interpolação de curvas genéricas.

No capítulo 5 são definidas as classes de padrões que apresentam ponto significativo, assim como, os modelos de rede propostos para a obtenção destes pontos. As etapas de processamento realizadas pela rede simbólica são apresentadas, com o objetivo de se explicar o processo realizado na detecção do pontos significativos. O modelo de rede neural artificial e alguns aspectos relativos ao treinamento desta, visando à classificação dos padrões de entrada (seqüência de pontos da curva plana), são então apresentados.

O capítulo 6 traz uma análise dos resultados obtidos nas etapas de interpolação e extração de pontos significativos. Na etapa de interpolação comparam-se os resultados obtidos pelo método proposto e pelo método original. São comparados ainda os resultados da interpolação com os resultados obtidos pelos métodos tradicionais de interpolação por partes. A parte de extração de pontos significativos compara, para situações diferentes da situação de treinamento, os pontos extraídos pelos dois modelos propostos; e também o resultado da interpolação passando por estes pontos é comparado com a curva original. Este

capítulo mostra ainda a alta capacidade de compressão dos métodos quando comparada a um sistema ideal, onde o conjunto de pontos significativos leva à interpolação com erro nulo.

Finalmente, o capítulo 7 traz as conclusões gerais sobre o trabalho, assim como, propõe alternativas a serem desenvolvidas em etapas futuras.

Capítulo 2

Fundamentos Teóricos

2.1 Introdução

Sistemas conexionistas e métodos seqüenciais para manipulação simbólica têm origens comuns com maior ênfase para o período denominado "período romântico" [MP69]. Nesta época, iniciaram-se os estudos de esquemas computacionais com inspiração biológica. Estes sistemas vieram em contrapartida aos sistemas de engenharia, cujo alto rigor matemático se mostrava ineficiente para tratar problemas de manipulação simbólica. Destes sistemas menos rígidos do ponto de vista matemático, resultaram áreas diversas como redes neurais e lógica nebulosa, cuja pesquisa envolvida vem crescendo a cada dia.

Atualmente, os sistemas conexionistas, em especial redes neurais artificiais, podem ser vistos como um elo de ligação entre Sistemas de Inteligência Artificial Simbólicos como, por exemplo, os sistemas baseados em regras de inferência nebulosa, e os Métodos da Engenharia [Bar89].

O principal objetivo deste capítulo é apresentar conceitos básicos das áreas de lógica nebulosa e redes neurais, descritos nas seções 2.2 e 2.3, respectivamente, e definir os conceitos elementares da teoria que inter-relaciona estas duas áreas, como será visto na seção 2.4.

2.2 Lógica Nebulosa

A teoria de lógica nebulosa é bastante extensa e hoje envolve áreas distintas como controle e manipulação simbólica. Entretanto, nesta seção serão apresentados, de forma

sucinta, apenas os conceitos básicos, necessários para o entendimento dos métodos de interpolação nebulosa e extração de pontos significativos, como será visto posteriormente.

2.2.1 Definições e Terminologia de Conjuntos Nebulosos

Universo de Discurso

Seja U um conjunto de objetos denominados genericamente por $\{u\}$, o qual pode ser contínuo ou discreto. U é denominado universo de discurso, onde u representa um elemento genérico de U.

Conjunto Nebuloso

Um conjunto nebuloso F, em um universo de discurso U, é caracterizado pela função de pertinência μ_F que toma valores no intervalo $[0,1]$, definida como:

$$\mu_F : U \rightarrow [0,1].$$

Um conjunto nebuloso pode ser visto como uma generalização do conceito de conjuntos ordinários, uma vez que as funções de pertinência destes últimos tomam somente os valores $\{0,1\}$. Portanto, para conjuntos ordinários existe uma borda definida entre membros e não membros da classe representada pelo conjunto. Já para os conjuntos nebulosos, esta borda não está bem definida e a transição de membro para não membro se apresenta de forma gradual [Kau75]. Um elemento u pode pertencer a uma determinada classe, representada pelo conjunto nebuloso F, com um grau $\mu_F(u)$.

O conjunto nebuloso F em U pode ser representado como um conjunto de pares ordenados de um elemento genérico u e o seu grau de pertinência. Portanto,

$$F = \{(u, \mu_F(u))/u \in U\}.$$

para o caso de U finito e discreto, tem-se

$$F = \sum_{i=1}^{n} \mu_F(u_i)/u_i.$$

Quando U é contínuo, tem-se F representado por:

$$F = \int_U \mu_F(u)/u.$$

Suporte e Ponto Limitante

O suporte de um conjunto nebuloso F é o conjunto ordinário de todos os pontos u em U tal que $\mu_F(u) > 0$.

Em particular, o elemento u em U para o qual $\mu_F = 0.5$ é denominado ponto limitante (*crossover point*).

Conjunto Nebuloso Unitário (*Singleton*)

Conjunto nebuloso cujo suporte consiste de um único ponto em U. Se A é um conjunto nebuloso unitário, então

$$A = \mu_A(u)/u.$$

Produto Cartesiano

Sejam $A_1, A_2, \cdots, A_n$, conjuntos nebulosos em $U_1, U_2, \cdots, U_n$, respectivamente. O produto cartesiano de $A_1, \cdots, A_n$ é um conjunto nebuloso no espaço do produto $U_1 \times U_2 \times \cdots \times U_n$, com função de pertinência:

$$\mu_{A_1 \times \cdots \times A_n}(u_1, u_2, \cdots, u_n) = \min\{\mu_{A_1}(u_1), \cdots, \mu_{A_n}(u_n)\} \text{ ou}$$

$$\mu_{A_1 \times \cdots \times A_n}(u_1, u_2, \cdots, u_n) = \mu_{A_1}(u_1).\mu_{A_2}(u_2) \cdots \mu_{A_n}(u_n)$$

Relação Nebulosa

Uma relação nebulosa é definida como um conjunto nebuloso no espaço do produto cartesiano $U \times V$ [Kau75]. Por exemplo, a relação $x \gg y$, $x, y \in \Re$, pode ser vista como um conjunto nebuloso $R \subset \Re^2$. A função de pertinência $\mu_R(x, y)$ poderia possuir os seguintes valores representativos:

$$\mu_R(10, 5) = 0 \qquad \mu_R(100, 10) = 0.7 \qquad \mu_R(1000, 1) = 1$$

Uma relação nebulosa n-ária é um conjunto nebuloso em $U_1 \times U_2 \times \cdots \times U_n$, podendo ser descrita como:

$$R_{U_1 \times \cdots \times U_n} = \{[(u_1, \cdots, u_n), \mu_R(u_1, \cdots, u_n)]/(u_1, \cdots, u_n) \in U_1 \times \cdots \times U_n\}$$

Composição de Relações

Se R e S são relações nebulosas em $U \times V$ e $V \times W$, respectivamente, a composição de R e S é uma relação nebulosa denotada por $R \circ S$ e definida por:

$$R \circ S = \{[(u, w), \sup_v(\mu_R(u, v) * \mu_S(v, w))], u \in U, v \in V, w \in W\}$$

onde $*$ pode ser qualquer operador na classe de normas triangulares [Ped89] (normas-t), como por exemplo, mínimo, produto algébrico, produto limitado, etc ...

2.2.2 Operações com Conjuntos Nebulosos

Sejam A e B dois conjuntos nebulosos em U, com funções de pertinência μ_A e μ_B, respectivamente. As operações de união, interseção e complemento, para conjuntos nebulosos, são definidas através de suas funções de pertinência.

União

A função de pertinência $\mu_{A\cup B}$ da operação união $A \cup B$ é definida, ponto-a-ponto, por:

$$\mu_{A\cup B}(u) = \max\{\mu_A(u), \mu_B(u)\},$$

para todo $u \in U$.

Interseção

A função de pertinência $\mu_{A\cap B}$ da interseção $A \cap B$ é definida, ponto-a-ponto, como:

$$\mu_{A\cap B}(u) = \min\{\mu_A(u), \mu_B(u)\},$$

para todo $u \in U$.

Complemento

Seja $\bar{A}$ o complemento de um conjunto nebuloso A. Então, a função de pertinência $\mu_{\bar{A}}$ é definida, ponto-a-ponto, por:

$$\mu_{\bar{A}}(u) = 1 - \mu_A(u),$$

Para todo $u \in U$.

2.2.3 Variáveis Lingüísticas

O uso de conjuntos nebulosos determina uma base para manipulação de conceitos vagos e imprecisos. Em particular, as variáveis lingüísticas podem ser representadas através dos conjuntos nebulosos. A seguir, serão definidos os conceitos de conjuntos nebulosos normais e convexos e também número nebuloso, necessários para a caracterização de uma variável lingüística.

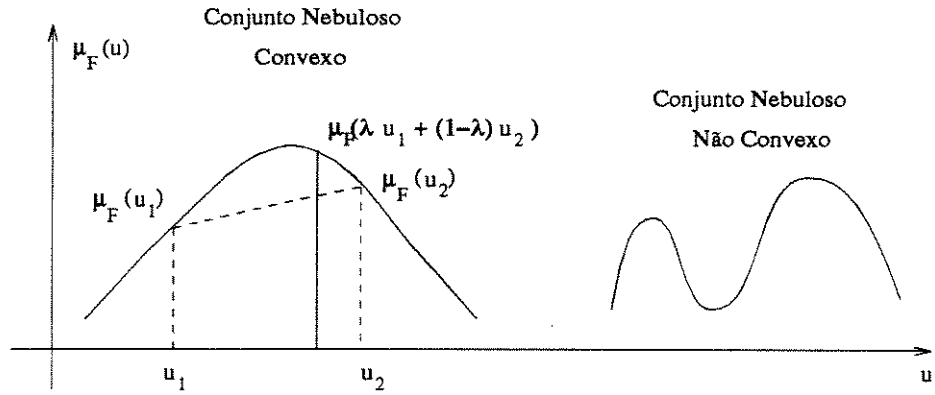


Figura 2.1: Conjuntos nebulosos convexos e não convexos.

Conjunto Nebuloso Normal

Conjunto com grau máximo de pertinência igual a *um*, ou seja,

$$\max_{u \in U} \mu_F(u) = 1.$$

Conjunto Nebuloso Convexo

Conjunto cuja função de pertinência deve obedecer à restrição

$$\mu_F(\lambda u_1 + (1-\lambda)u_2) \geq \min\{\mu_F(u_1), \mu_F(u_2)\},$$

para $u_1, u_2 \in U$ e $\lambda \in [0,1]$. A figura 2.1 ilustra os exemplos de um conjunto convexo e não convexo.

Número Nebuloso

Um número nebuloso F em um universo de discurso U contínuo (eixo real) é um conjunto nebuloso F, em U, normal e convexo.

Uma variável lingüística, como o próprio nome sugere, é uma variável cujos valores são palavras ou sentenças em linguagem natural [Zad75]. Por exemplo, *idade* pode ser vista como uma variável lingüística, cujos valores podem ser *jovem, muito jovem, não jovem, velho, mais ou menos velho* e assim por diante. A uma variável lingüística associa-se uma distribuição de possibilidades e também números nebulosos.

Caracterização de uma Variável Lingüística

Uma variável lingüística é caracterizada por uma quíntupla $(x, T(x), U, G, M)$, onde x representa o nome da variável; $T(x)$ é o conjunto de termos de x, isto é, o conjunto de

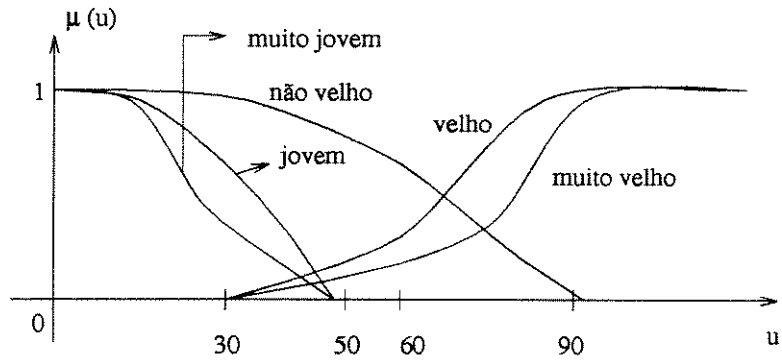


Figura 2.2: Funções de pertinência dos valores lingüísticos jovem, muito jovem, velho, não velho, muito velho.

nomes dos valores lingüísticos de x, onde cada valor é um número nebuloso definido em U; G é a regra sintática para geração dos nomes dos valores de x; e M é a regra semântica para associar cada valor ao seu significado. Como exemplo, seja *idade* uma variável lingüística, então, o conjunto de termos T(idade), poderia ser,

$$T(\text{idade}) = \{\text{jovem, muito jovem, velho, não velho, muito velho}\}$$

onde cada termo em T(idade) é caracterizado por um número nebuloso no universo de discurso $U = [0, 100]$. As possíveis funções de pertinência desses números nebulosos estão ilustradas na figura 2.2.

Deve-se salientar que a determinação da função de pertinência está diretamente ligada ao contexto. Por exemplo, seja a variável lingüística *velocidade*. Um valor de 150 Km/hora pode ser considerado *alto* caso se trate de velocidade de automóvel. Entretanto, se o contexto for alterado para velocidade de avião, este valor passa a ser visto como um valor *muito baixo*.

Uma questão importante diz respeito às partições nebulosas dos espaços de entrada e saída. Em geral, uma variável lingüística é associada a um conjunto de termos, onde cada elemento é definido no mesmo universo de discurso. Então, uma partição nebulosa determina quantos termos lingüísticos devem existir. A figura 2.3 traz um exemplo típico, onde são mostradas duas partições nebulosas para um mesmo universo normalizado $[-1, 1]$. A primeira partição consiste de três termos: N (negativo), ZE (zero), e P (positivo). A segunda partição apresenta um refinamento maior e determina sete termos: GN(grande negativo), MN (médio negativo), PN (pequeno negativo), ZE (zero), PP (pequeno positivo), MP (médio positivo) e GP (grande positivo). A escolha do grau de refinamento (quantidade de termos obtidos na partição) vai depender da precisão necessária para a aplicação em questão. O

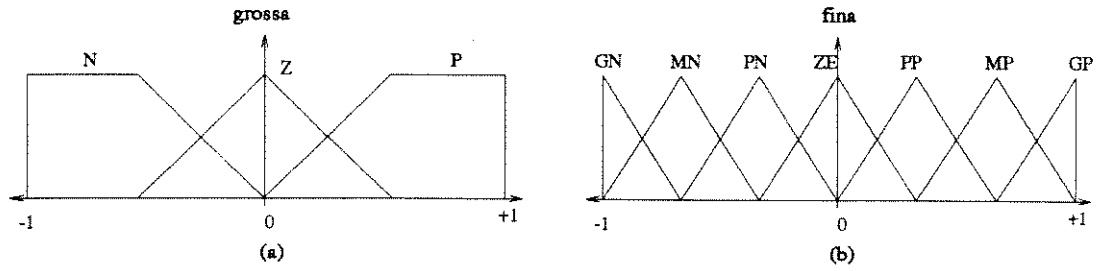


Figura 2.3: Exemplo de partições diferentes para o mesmo universo de discurso. (a) partição grossa. (b) partição fina.

grau de refinamento da partição define a cardinalidade do conjunto de termos T, em um espaço de entradas nebulosas, que por sua vez determina o número máximo de regras que serão construídas.

2.2.4 Mecanismos de Inferência em Raciocínio Aproximado

Normalmente, contrói-se um sistema inteligente baseado em um conjunto de regras extraídas do conhecimento de um especialista. Em geral, o conjunto de regras

$$R = \{R_1, R_2 \cdots, R_n\}$$

tem a forma de múltiplas entradas e múltiplas saídas, onde R_i pode ser representada pela declaração condicional,

$$\text{Se} \quad (x \text{ é } A_i \ \underline{\text{e}} \ y \text{ é } B_i) \ \underline{\text{então}} \ (z_1 \text{ é } C_i; \cdots; z_p \text{ é } Q_i).$$

Para maior simplicidade, será considerado um sistema de duas entradas x e y com saída única z, na forma,

$$\begin{array}{lll} \underline{\text{entrada}} & : & x \text{ é } A' \ \underline{\text{e}} \ y \text{ é } B' \\ R_1 & : & \text{Se} \ \ x \text{ é } A_1 \ \underline{\text{e}} \ y \text{ é } B_1 \ \ \underline{\text{então}} \ \ z \text{ é } C_1 \\ R_2 & : & \text{Se} \ \ x \text{ é } A_2 \ \underline{\text{e}} \ y \text{ é } B_2 \ \ \underline{\text{então}} \ \ z \text{ é } C_2 \\ & & \vdots \\ R_n & : & \text{Se} \ \ x \text{ é } A_n \ \underline{\text{e}} \ y \text{ é } B_n \ \ \underline{\text{então}} \ \ z \text{ é } C_n \\ \hline \underline{\text{saída}} & & z \text{ é } C' \end{array} \tag{2.1}$$

onde, x, y, z são variáveis lingüísticas; A_i, B_i e C_i são os valores lingüísticos das variáveis x, y, z nos universos de discurso U, V, W, respectivamente, com $i = 1, 2, \cdots, n$.

A regra nebulosa Se $(x$ é A_i e y é $B_i)$ então $(z$ é $C_i)$ é implementada como uma implicação (relação) nebulosa R_i, cuja função de pertinência μ_{R_i} pode ser definida por:

$$\begin{aligned} \mu_{R_i} &\triangleq \mu_{(A_i \text{ e } B_i \rightarrow C_i)}(u,v,w) \\ &= [\mu_{A_i}(u) \text{ e } \mu_{B_i}(v)] \rightarrow \mu_{C_i}(w) \end{aligned},$$

onde A_i e B_i é um conjunto nebuloso $A_i \times B_i$ em $U \times V$; $R_i \triangleq (A_i \text{ e } B_i) \rightarrow C_i$ pode ser vista como uma relação nebulosa em $U \times V \times W$; e o operador $\rightarrow$ denota a função de implicação nebulosa.

O conseqüente C' é deduzido da regra de inferência composicional sup-$*$, através da determinação da função de implicação nebulosa, do conectivo e e da interação entre as regras R_i. A seguir serão ilustrados estes três conceitos.

1. **Função de Implicação Nebulosa**

   Existem duas regras importantes de inferência para implicações nebulosas, utilizadas em raciocínio aproximado, que são - *modus ponens* generalizado e *modus tollens* generalizado [Lee90]. A partir da regra de inferência composicional introduzida por Zadeh [Zad73], vários autores têm proposto novas funções de implicação nebulosa. Em geral, as famílias de funções de implicação nebulosa podem ser classificadas em três categorias principais - conjunção nebulosa, disjunção nebulosa e implicação nebulosa. A seguir será definida a conjunção nebulosa que serve de base para a função de implicação nebulosa Operação de Mínimo definida por Mamdani.

   Conjunção Nebulosa

   $$A \rightarrow B = A \times B = \int_{U \times V} \mu_A(u) * \mu_B(v)/(u,v)$$

   para todo $u \in U$ e $v \in V$. Onde o operador $*$ representa uma *norma-t* [Ped89].

   Operação de Mínimo de Mamdani

   $$\begin{aligned} R_c &= A \rightarrow B = A \times B \\ &= \textstyle\int_{U \times V} \mu_A(u) \wedge \mu_B(v)/(u,v) \end{aligned}$$

   onde $\wedge$ determina a operação de mínimo (interseção).

2. **Conectivo e**

   Geralmente, o conectivo e é implementado como uma conjunção nebulosa num produto de espaços, onde as variáveis envolvidas estão em universos de discurso diferentes.

Seja, por exemplo, uma relação R

$$\text{Se } (A \text{ e } B) \text{ então } C.$$

O antecedente $(A \text{ e } B)$ é interpretado como um conjunto nebuloso no espaço do produto $U \times V$, com função de pertinência dada por:

$$\mu_{A\times B}(u,v) = \min\{\mu_A(u), \mu_B(v)\}$$

ou então,

$$\mu_{A\times B}(u,v) = \mu_A(u).\mu_B(v).$$

3. **Interação entre as Regras R_i**

Quando um sistema nebuloso é caracterizado por um conjunto de regras, a ordem destas regras não importa para a definição da saída inferida. Portanto, a função que inter-relaciona as regras R_i deve possuir propriedades de comutatividade e associatividade. Os operadores em normas triangulares (norma-t) e co-normas (norma-s) possuem estas propriedades. Conseqüentemente, estes operadores podem ser usados com o objetivo de se obter uma interação entre as regras R_i, $i = 1, \cdots, n$.

2.2.5 Obtenção do Conseqüente C'

Lema 2.2.1 *Se as entradas do sistema dado em 2.1 são conjuntos nebulosos unitários (singletons), com $A' = u_0$ e $B' = v_0$, então, o resultado derivado da aplicação do operador mínimo de Mamdani R_c pode ser expresso como:*

$$R_c : \alpha_i^{\wedge} \wedge \mu_{C_i}(w)$$

ou

$$R_c : \alpha_i^{\circ} \wedge \mu_{C_i}(w)$$

onde

$$\alpha_i^{\wedge} = \mu_{A_i}(u_0) \wedge \mu_{B_i}(v_0), \qquad \alpha_i^{\circ} = \mu_{A_i}(u_0).\mu_{B_i}(v_0).$$

Portanto,

$$R_c : \mu_{C'} = \bigcup_{i=1}^{n} \alpha_i \wedge \mu_{C_i}$$

e o peso α_i pode ser definido como contribuição da i$^{\text{ésima}}$ regra para a saída inferida. O fator α_i pode ser calculado dos dois modos anteriormente definidos, $\alpha_i^{\wedge}$ e α_i°. A prova deste

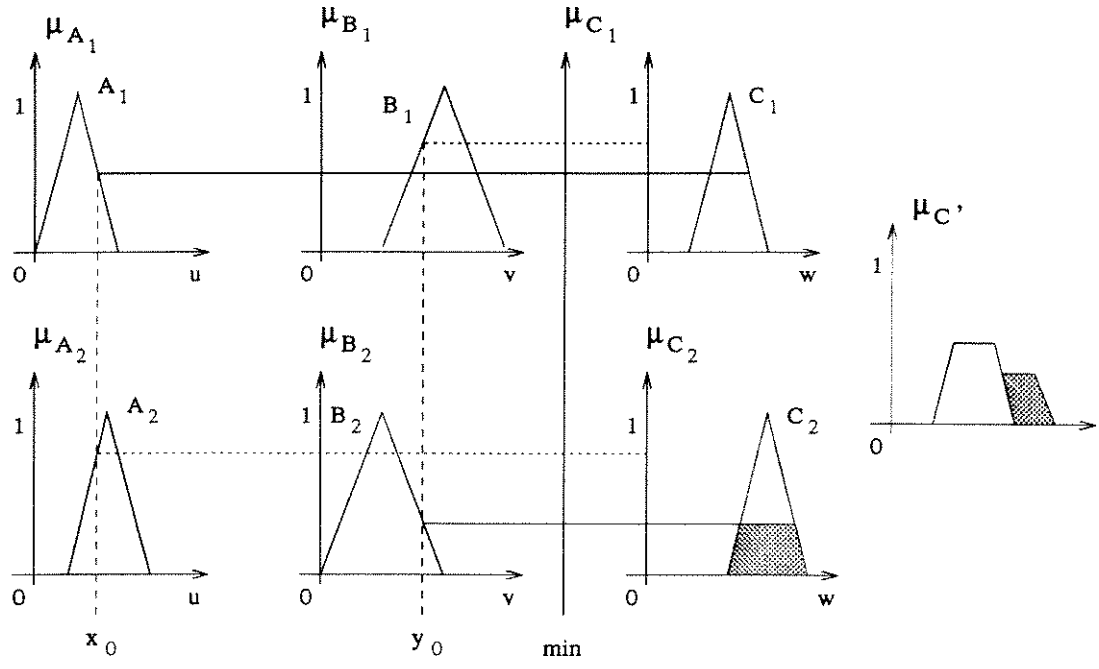


Figura 2.4: Diagrama de representação do raciocínio nebuloso do tipo 1.

lema, assim como outras propriedades dos mecanismos de inferência, podem ser encontradas em [Lee90].

Seja R o conjunto de regras dado em 2.1, para $\alpha_i = \mu_{A_i}(x_0) \wedge \mu_{B_i}(y_0)$. Então, para a determinação do conseqüente C' é necessária a definição do tipo de raciocínio nebuloso. Existem quatro tipos atualmente empregados na área de controle [Lee90], entre os quais se destacam:

- Tipo 1 - **Operador de Mínimo de Mamdani como Função de Implicação Nebulosa**

  A i$^{\text{ésima}}$ regra determina uma saída

  $$\mu_{C_i'}(w) = \alpha_i \wedge \mu_{C_i}(w).$$

  e a função de pertinência da saída C' pode ser definida como:

  $$\mu_{C'}(w) = \bigcup_{i=1}^{n} \mu_{C_i'}(w) = \bigcup_{i=1}^{n} \alpha_i \wedge \mu_{C_i}(w).$$

  A figura 2.4 ilustra o processo de raciocínio nebuloso para as condições dadas no lema 2.2.1, com função de implicação nebulosa definida pelo operador mínimo de

Mamdani. Este método, como será visto na seção 4.3.1, é utilizado na determinação de um dos parâmetros da interpolação nebulosa.

- Tipo 2 - **O Conseqüente da Regra é uma Função das Variáveis Lingüísticas de Entrada**

  Neste método de raciocínio, a i$^{\text{ésima}}$ regra tem a forma:

$$R_i : \text{Se} \quad (x \text{ é } A_i \ \underline{\text{e}} \ y \text{ é } B_i) \quad \underline{\text{então}} \quad z = f_i(x, y)$$

  onde x, y, z são variáveis lingüísticas; A_i e B_i são valores lingüísticos das variáveis x e y nos universos de discurso U e V, respectivamente, com $i = 1, \cdots, n$; e f_i é uma função de x, y definida no espaço de entrada.

  Por simplicidade, e a título de ilustração, será adotado um sistema de 3 regras

$$\begin{aligned} R_1 &: \text{Se} \quad x \text{ é } A_1 \ \underline{\text{e}} \ y \text{ é } B_1 \quad \underline{\text{então}} \quad z = f_1(x, y) \\ R_2 &: \text{Se} \quad x \text{ é } A_2 \ \underline{\text{e}} \ y \text{ é } B_2 \quad \underline{\text{então}} \quad z = f_2(x, y) \\ R_3 &: \text{Se} \quad x \text{ é } A_3 \ \underline{\text{e}} \ y \text{ é } B_3 \quad \underline{\text{então}} \quad z = f_3(x, y) \end{aligned}$$

  O método proposto por Takagi e Sugeno [TS83] define o valor da saída inferida, para a primeira regra, como sendo $\alpha_1 f_1(x_0, y_0)$; para a segunda, $\alpha_2 f_2(x_0, y_0)$; e $\alpha_3 f_3(x_0, y_0)$. A saída não nebulosa é definida por:

$$z_0 = \frac{\alpha_1 f_1(x_0, y_0) + \alpha_2 f_2(x_0, y_0) + \alpha_3 f_3(x_0, y_0)}{\alpha_1 + \alpha_2 + \alpha_3}.$$

  Os métodos de interpolação nebulosa, como será visto nas seções 4.2 e 4.3, utilizam este tipo de raciocínio para a definição da saída interpolada.

Existem aplicações onde é necessário que a saída inferida seja não nebulosa, como a saída obtida pelo método descrito anteriormente (tipo 2). Para estas aplicações e nos casos onde o método de raciocínio nebuloso determina saídas nebulosas, deve existir um processamento extra que transforme a saída nebulosa em não nebulosa (*crisp*). A seção a seguir ilustra alguns métodos para se realizar este processamento.

2.2.6 Cálculo de Saídas não Nebulosas

O processo para o cálculo das saídas não nebulosas consiste, basicamente, do mapeamento do espaço de saídas nebulosas, definido sobre o universo de discurso da saída, em um espaço não nebuloso.

Os critérios mais conhecidos para o cálculo deste mapeamento são:

- Critério do Máximo

  Este método produz como saída não nebulosa o ponto onde a distribuição de possibilidades (função de pertinência) da saída alcança o valor máximo.

- Método da Média dos Máximos

  Esta estratégia gera uma saída não nebulosa que representa a média de todos os valores que alcançam o valor máximo na função de pertinência. Mais especificamente, para o caso de um universo de discurso discreto, tem-se:

$$z_0 = \sum_{j=1}^{m} \frac{w_j}{m},$$

  onde w_j representa os valores do suporte para os quais a função de pertinência atinge o valor máximo $\mu_z(w_j)$, e m é a quantidade destes valores.

- Método do Centro de Área

  Este método define como saída não nebulosa o centro de gravidade da distribuição de possibilidades. Para o caso discreto, tem-se:

$$z_0 = \sum_{j=1}^{n} \frac{\mu_z(w_j).w_j}{\sum_{j=1}^{n} \mu_z(w_j)},$$

  onde n é o número de níveis de quantização da saída, ou seja, o número de segmentos gerados pela discretização do universo de discurso.

2.3 Redes Neurais Artificiais

Existem várias razões para se solucionarem problemas complexos através de modelos de máquinas com processamento paralelo. A inspiração no cérebro humano vem da capacidade deste em resolver, de forma rápida e satisfatória, problemas mal definidos e que exijam enorme esforço computacional como, por exemplo, reconhecimento de imagens visuais, reconhecimento de voz, etc...

A principal semelhança entre o sistema nervoso biológico e redes neurais artificiais é que ambos consistem, basicamente, de um grande número de elementos com processamento simples, alta conectividade e que, juntos, são capazes de resolver problemas complexos e ambíguos.

Atualmente, o estudo de redes neurais reúne assuntos amplos e complexos. Todavia, nesta seção serão tratados somente alguns aspectos básicos voltados para a classificação de padrões. A seção a seguir define alguns conceitos necessários para o entendimento do modelo e algoritmo de treinamento da rede neural, utilizada no processo de extração de pontos significativos, como será visto na seção 5.3.

2.3.1 Definições e Conceitos Iniciais

- Padrões de Entrada-Saída da Rede Neural

  Os padrões de entrada são, normalmente, conjuntos de vetores coluna representando entradas discretas ou contínuas. Estes vetores se apresentam à camada de entrada da rede, um de cada vez a cada passo de processamento, produzindo na saída um vetor de elementos discretos ou contínuos. A característica de continuidade da saída vai depender da função de ativação.

- Conjunto de Conexões

  Em uma rede neural, os elementos processadores se conectam, ou seja, trocam informações, através das sinapses. A intensidade com a qual esta conexão se realiza é denominada de peso da sinapse. Desta forma, a sinapse determina o efeito que a saída de uma unidade processadora[1] exerce sobre a outra, ou sobre ela mesma, dependendo da topologia adotada.

[1] Usualmente, em redes neurais, estes elementos são denominadas de neurônios. Entretanto, neste trabalho o termo neurônio será usado para denominar as unidades processadoras de um outro modelo de rede, como será visto na seção 2.4

- Função de Ativação

  Os sinais de entrada, ponderados pelos pesos das respectivas conexões, são agrupados e submetidos à função de ativação. Esta função define estados de ativação que podem ser discretos ou contínuos[2]. A função sinal é um exemplo típico do caso discreto. As funções sigmóide e tangente hiperbólica são as funções não lineares mais usuais para o caso contínuo. A escolha de uma das duas depende do intervalo de variação desejado para a saída da rede, uma vez que, a função sigmóide estabelece um intervalo de variação para a saída de $[0, 1]$ e a função tangente hiperbólica gera saídas que variam de -1 a 1.

- Topologia de Redes Neurais

  Existem três tipos básicos de topologias:

  1. Redes em Camadas
  2. Redes Recorrentes em Camadas
  3. Redes Totalmente Conectadas.

  O primeiro tipo, onde os elementos estão distribuídos em camadas, é o mais restritivo e determina que só há conexões entre os elementos de camadas adjacentes e estas conexões têm sentido de propagação da entrada para a saída. As redes recorrentes também apresentam as unidades em camadas, com a diferença que realimentações dos níveis posteriores para os anteriores são permitidas. Para o terceiro tipo, redes totalmente conectadas, são permitidas conexões entre todas as unidades processadoras.

- Algoritmos de Treinamento

  Os algoritmos de aprendizado se dividem em duas classes principais:

  1. Algoritmos de aprendizado supervisionado.
  2. Algoritmos de aprendizado auto-organizado.

  A diferença entre os dois está na utilização ou não dos padrões de saídas desejadas. O treinamento supervisionado é feito em função do erro (saída desejada - saída da rede). Já os treinamentos auto-organizados utilizam apenas os vetores de padrões de entrada. Nesta seção serão abordados apenas os aspectos relativos ao aprendizado supervisionado.

  Um outro tipo de separação é feito com base nas regras utilizadas:

[2]Existe ainda a função de saída que é aplicada sobre o estado de ativação. Neste caso, a função de saída adotada será a função identidade. Com isso, o tipo de saída (contínua ou discreta) é definido apenas pela função de ativação

1. Regras de Correção de Erro
2. Regras de Gradiente

As regras de correção alteram os pesos da rede de modo a corrigir o erro na resposta, para o padrão de entrada apresentado. As regras de gradiente, por sua vez, mudam os valores dos pesos, durante a apresentação de cada padrão, pela descida do gradiente do erro. O erro mínimo quadrático é calculado pela média de todos os padrões de treinamento. O exemplo mais conhecido de algoritmo de treinamento supervisionado com regras de gradiente é o *backpropagation*, cujo princípio de funcionamento será mostrado na seção 2.3.3.

- Combinador Linear Adaptativo

  Um combinador linear adaptativo representa o elemento básico na composição da maioria das redes neurais, e outros sistemas adaptativos [WL90]. Em implementações digitais, este elemento recebe no tempo k um vetor padrão de entrada $X_k = [x_{0k}, x_{1k}, \cdots, x_{nk}]^T$ e uma resposta desejada d_k. Os componentes do vetor de entrada são ponderados pelo vetor de pesos $W_k = [w_{0k}, w_{1k}, \cdots, w_{nk}]^T$. A soma das entradas ponderadas é calculada, produzindo um saída linear $S_k = X_k^T W_k$.

2.3.1.1 Classificadores Lineares

Muitas redes neurais utilizam, como processador básico, o elemento linear adaptativo *Adaline* mostrado na figura 2.5. Este elemento consiste de um combinador linear adaptativo em cascata com um bloco processador da função de ativação $Y_k = \mathrm{sgn}(S_k)$, onde sgn é a função sinal. O peso w_{0k}, que é conectado a uma entrada de valor constante +1, funciona como limiar da função de ativação. Um único elemento processador é capaz de realizar apenas as funções lógicas linearmente separáveis. Esta limitação foi responsável por um longo perídodo de desinteresse na área de redes neurais, denominado de "período de inverno das redes neurais" [MP69].

A figura 2.6(a) mostra um elemento do tipo *Adaline* com duas entradas binárias e 2.6(b) representa todas as possíveis combinações destas entradas. O *Adaline* é capaz de separar os padrões de entrada em duas categorias, dependendo dos valores dos pesos. A condição de limiar crítica acontece quando a saída linear s se iguala a zero:

$$s = x_1 w_1 + x_2 w_2 + w_0.$$

Esta relação linear está bem ilustrada na figura 2.6(b). O esquema desta figura determina saída *um* para o padrão de entrada (1,1) e *zero* para os outros casos, ou seja, a implementação

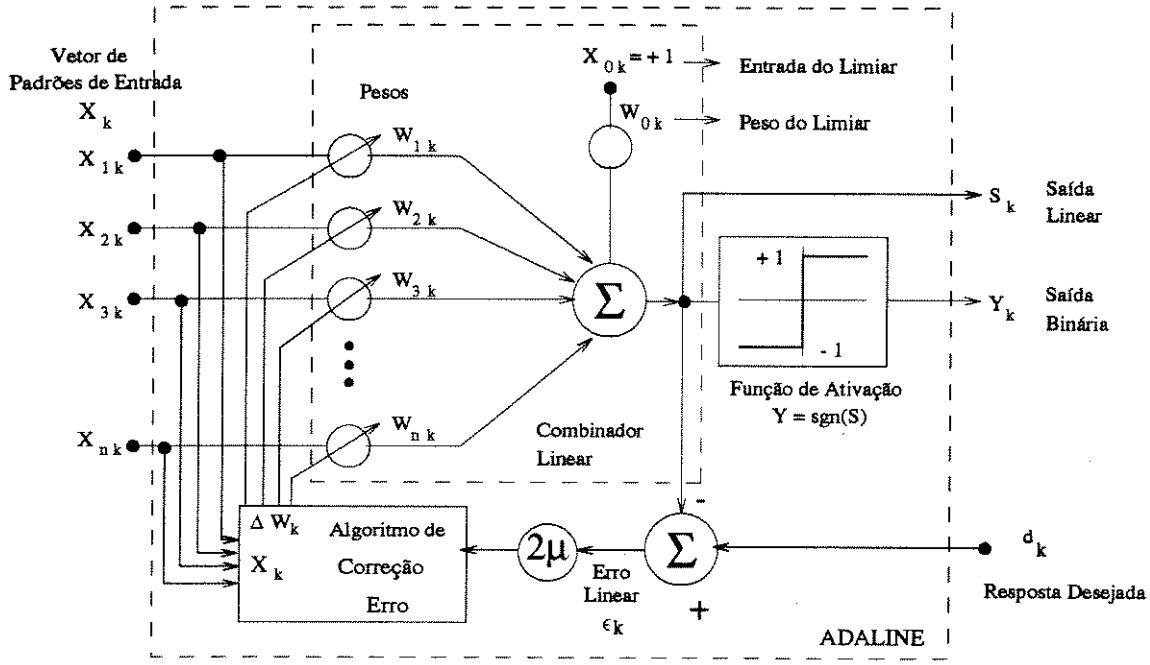


Figura 2.5: Elemento Linear Adaptativo (*Adaline*).

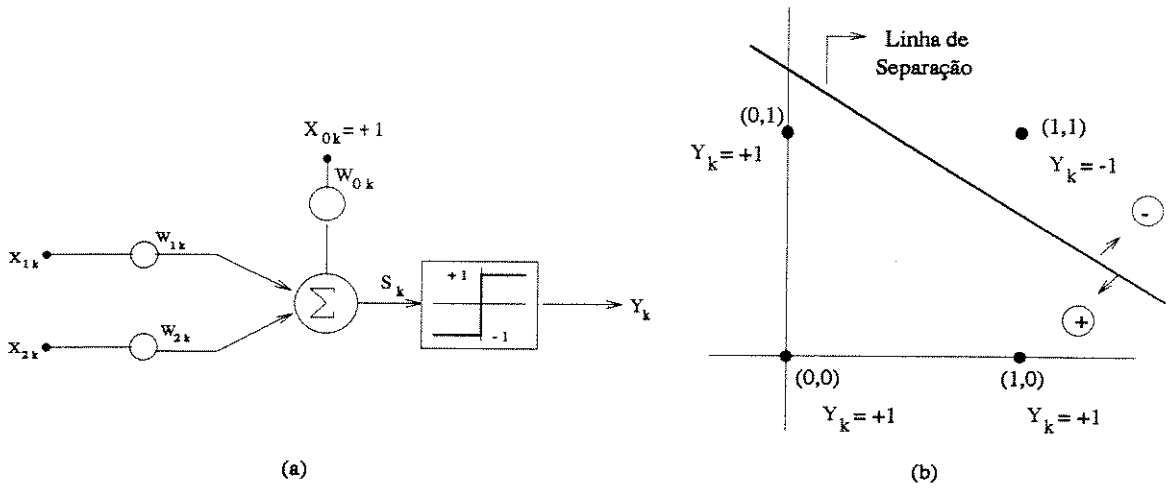


Figura 2.6: (a) Adaline de duas entradas. (b) Separação no espaço de padrões.

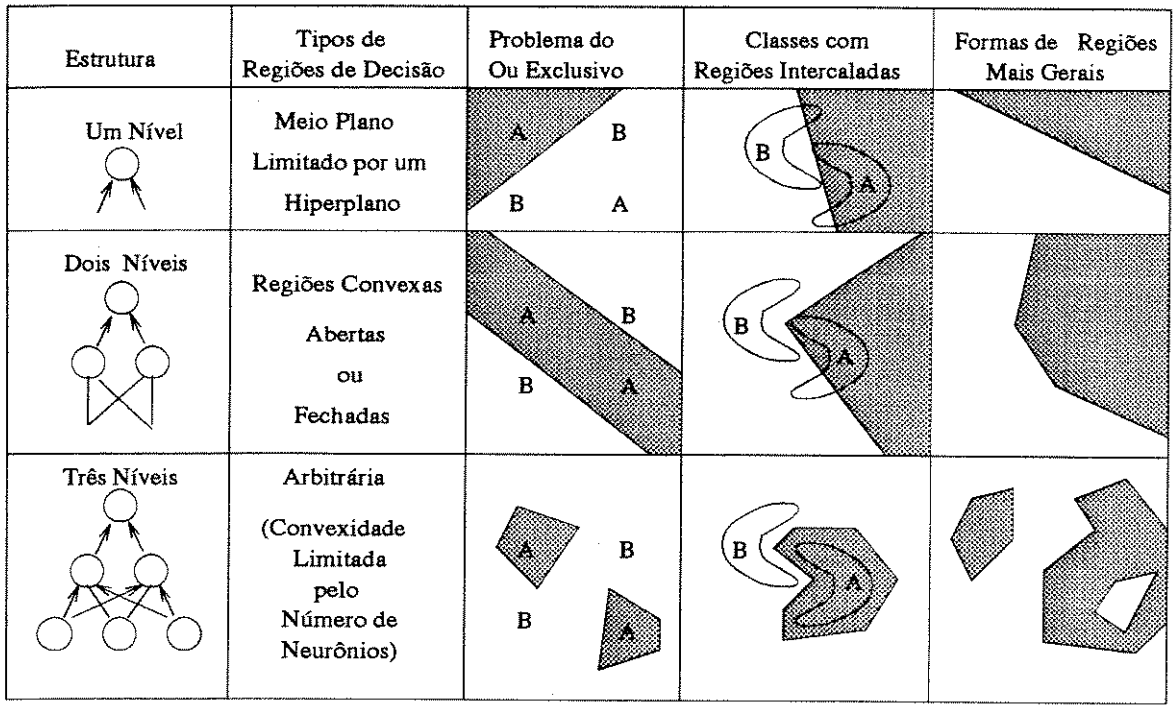


Figura 2.7: Regiões de decisão formadas por diferentes estruturas de redes.

da função lógica AND. Este é um exemplo simples de uma função linearmente separável. Entretanto, existem funções lógicas igualmente simples mas que não podem ser separadas de forma linear. Um exemplo bastante típico é a função OU-exclusivo.

2.3.1.2 Classificadores Não Lineares

Os classificadores lineares são restritos em sua capacidade, principalmente por se limitarem à discriminação de padrões com formas linearmente separáveis. A introdução da propriedade de separação não linear pode ser feita de duas maneiras:

1. Fixação de um pré-processador com funções não lineares aplicadas ao vetor de entrada.
2. Implementação de uma rede neural com mais de uma camada.

Verifica-se que a primeira solução, obtida através da introdução de funções polinomiais nos elementos pré-processadores, possibilita a formação de regiões de decisão não linares. Entretanto, experiências demostram que melhores generalizações, isto é, a capacidade da rede em responder corretamente aos padrões não treinados, são obtidas utilizando-se redes com mais de um nível de elementos processadores. A figura 2.7 [Lip87] traz uma comparação das regiões de decisão criadas por estruturas com um, dois e três níveis de elementos processadores; onde a função de ativação utilizada é a função sinal. As colunas 2 e 3 trazem

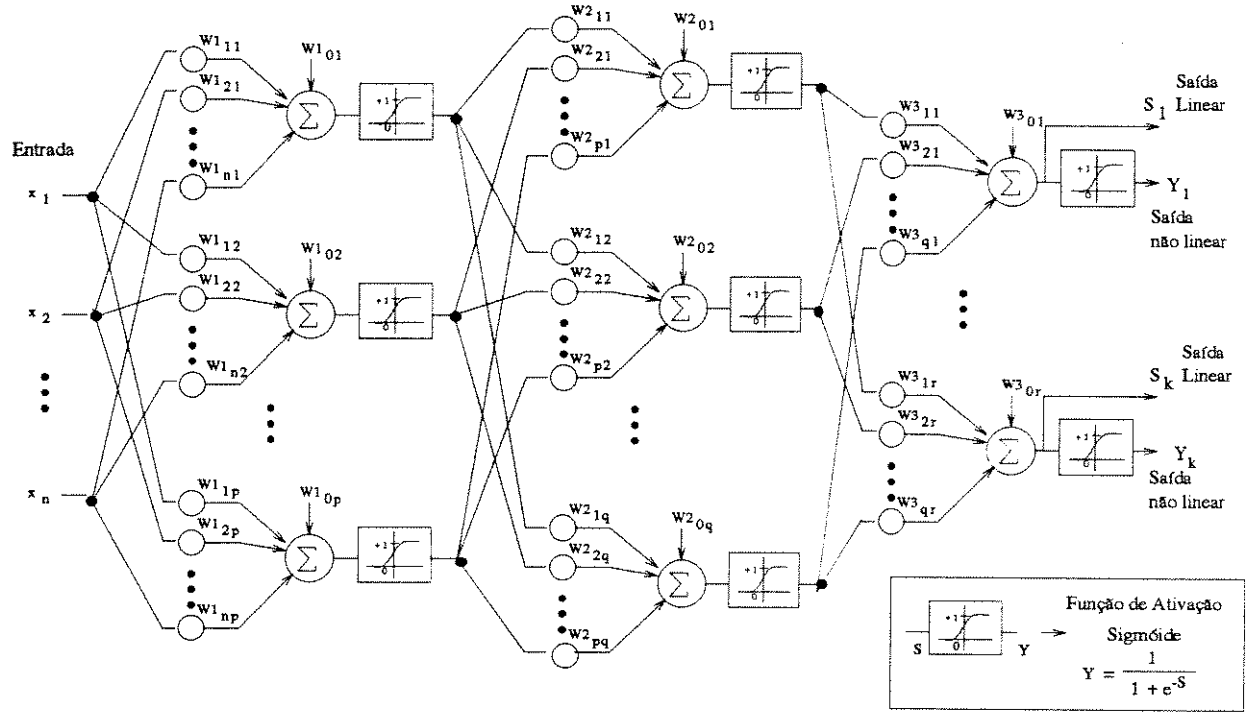


Figura 2.8: Modelo do "perceptron" de três camadas.

exemplos de problemas não linearmente separáveis. A primeira situação mostrada é a do OU-EXCLUSIVO, onde se observa uma distribuição disjunta que pode ser resolvida por uma rede com dois níveis, como mostra a região sombreada. Já a segunda situação apresenta classes com regiões intercaladas ("meshed regions") e, neste caso, somente uma rede com no mínimo três níveis de elementos processadores é capaz de realizar a separação. Isto se deve ao fato de que três níveis possiblitam a formação de regiões mais complexas.

Conforme será visto na seção 5.3, o problema de classificação de padrões abordado neste trabalho se encaixa na segunda situação. O modelo adotado será o "perceptron" treinado pelo algoritmo "backpropagation" descritos nas seções 2.3.2 e 2.3.3, respectivamente.

2.3.2 Modelo "Perceptron" de Múltiplas Camadas

O modelo de rede denominado *perceptron* com múltiplas camadas é uma estrutura de rede neural com propagação direta de informação, que apresenta um ou mais níveis de elementos processadores entre os níveis de entrada e saída. A figura 2.8 mostra o modelo com três níveis de elementos processadores, onde a função de ativação é dada pela função sigmóide.

Conforme visto na seção anterior, para a função de ativação sinal são necessários no mínimo três níveis de elementos, de modo a permitir a formação de regiões mais complexas. Todavia, para o caso de função de ativação sigmóide, pouco se sabe sobre a capacidade de classificação da rede. Deve-se ressaltar no entanto, que a característica mais interessante em processamento via redes neurais é a capacidade de generalização da rede. Neste aspecto, o estudo da relação capacidade de classificação/ capacidade de generalização abre um caminho que, embora apresente ainda poucos resultados expressivos, aparece como uma área bastante atraente.

2.3.3 Algoritmo de Treinamento "Backpropagation"

A publicação do algoritmo *backpropagation* por Rumelhart *et al.* [RM86] representou, indiscutivelmente, o desenvolvimento de maior influência no campo de redes neurais durante a última década.

O processo de funcionamento da técnica de propagação retroativa do erro é mostrado na figura 2.9 [WL90], para uma rede de dois níveis. A idéia básica do algoritmo é ajustar, a cada instante k, os pesos da rede na direção oposta à do gradiente do erro instantâneo, como definido a seguir:

$$W_{k+1} = W_k + \mu(-\nabla_k), \tag{2.2}$$

com W_k representando o conjunto de todos os pesos da rede; μ indicando a taxa de aprendizagem (velocidade com que os pesos são alterados); e o gradiente do erro ∇_k dado por:

$$\nabla_k = \frac{\partial \varepsilon_k^2}{\partial W_k} = \begin{bmatrix} \frac{\partial \varepsilon_k^2}{\partial W_{1k}} \\ \frac{\partial \varepsilon_k^2}{\partial W_{2k}} \\ \vdots \\ \frac{\partial \varepsilon_k^2}{\partial W_{mk}} \end{bmatrix} \tag{2.3}$$

onde m determina o total de conexões.

A soma instantânea do erro quadrático ε_k^2 é definida como a soma dos quadrados do erro em cada saída N_y da rede. Então,

$$\varepsilon_k^2 = \sum_{i=1}^{N_y} \epsilon_{i_k}^2 = \sum_{i=1}^{N_y} (d_{i_k} - y_{i_k})^2.$$

A equação 2.3 pode ser reescrita como:

$$\nabla_k = \frac{\partial \varepsilon_k^2}{\partial W_k} = \frac{\partial \varepsilon_k^2}{\partial s_k} \frac{\partial s_k}{\partial W_k} = \frac{\partial \varepsilon_k^2}{\partial s_k} \frac{\partial W_k^T X_k}{\partial W_k}.$$

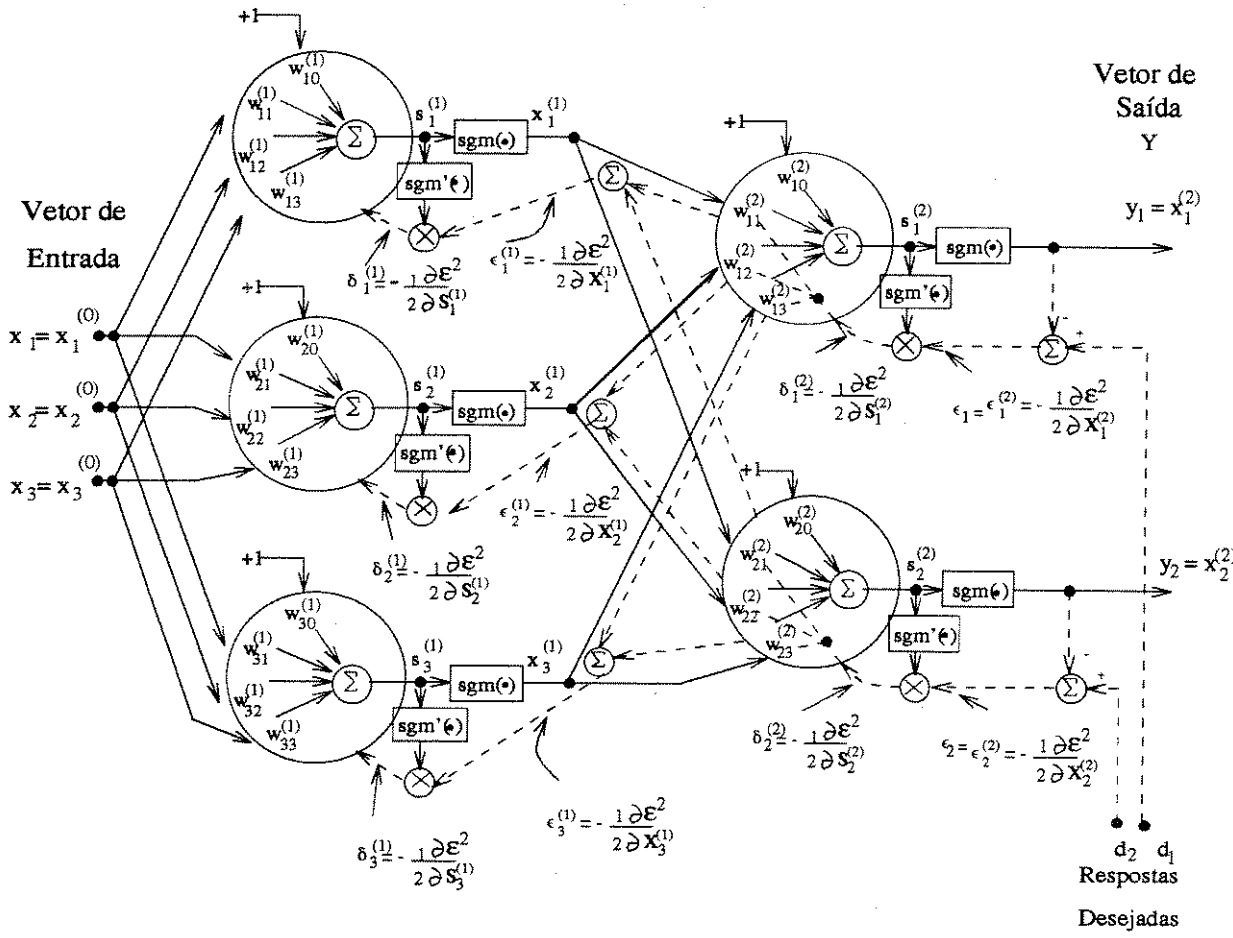


Figura 2.9: Exemplo do "backpropagation" aplicado à uma rede de dois níveis de elementos processadores.

Mas W_k e X_k são independentes. Portanto, o gradiente do erro passa a ser dado por:

$$\nabla_k = \frac{\partial \varepsilon_k^2}{\partial s_k} X_k.$$

Definindo-se $\delta_k = \frac{-1}{2} \frac{\partial \varepsilon_k^2}{\partial S_k}$, tem-se

$$\nabla_k = -2\delta_k X_k.$$

Então, equação 2.2 passa a ser dada por:

$$W_{k+1} = W_k + 2\mu\delta_k X_k \tag{2.4}$$

A partir deste ponto, o índice k indicando o instante atual será suprimido. Todo o processamento descrito a seguir será entendido como a apresentação de uma entrada e sua respectiva saída desejada (para um conjunto de padrões de treinamento), em um instante qualquer.

Para o exemplo da figura 2.9, a soma dos erros quadráticos é dada por:

$$\varepsilon_i^2 = (d_1 - y_1)^2 + (d_2 - y_2)^2 \tag{2.5}$$

Na sua forma mais simples, o treinamento pelo algoritmo "backpropagation" começa pela apresentação do vetor de padrões de entrada X à rede, propagando a informação para a frente de forma a gerar um vetor resposta Y e computar o erro na saída. O próximo passo envolve propagar o efeito do erro para trás de forma a associar a "derivada do erro quadrático" δ a cada unidade processadora, computando o gradiente de cada δ. Finalmente, o último passo consiste da atualização dos pesos para cada elemento, baseado no correspondente gradiente. Um novo padrão é apresentado e o processo se repete. O valores dos pesos iniciais são gerados randomicamente[3].

Para se ter uma idéia melhor sobre os cálculos associados ao algoritmo *back-propagation*, a figura 2.9 será analisada em detalhes. Cada um dos cinco círculos grandes representa um combinador linear. As linhas sólidas indicam o sentido direto de propagação da informação de entrada e as linhas pontilhadas, os caminhos reversos que são usados de forma associada aos cálculos das derivadas δ do erro quadrático.

Após propagar a informação de entrada pelos níveis da rede, até gerar a resposta

[3]Os valores devem ter pequenas variações em torno do zero. Entretanto, o algoritmo nem sempre funciona para todos os pesos inicializados em zero ou escolhidos de forma inadequada.

na saída, calcula-se a derivada do erro (δ) associado à $j^{\text{ésima}}$ unidade do nível n[4]. Então,

$$\delta_j^{(n)} = -\frac{1}{2}\frac{\partial \varepsilon^2}{\partial s_j^{(n)}}.$$

Essencialmente, estas derivadas definem a sensibilidade da variação da soma dos erros quadráticos de saída em relação à saída linear da unidade processadora associada.

2.3.3.1 Cálculo das Variações δ na Camada de Saída

Para a primeira unidade na camada de saída (fig. 2.9),

$$\delta_1^{(2)} = -\frac{1}{2}\frac{\partial \varepsilon^2}{\partial s_1^{(2)}}. \tag{2.6}$$

Aplicando-se 2.5 à equação 2.6, tem-se:

$$\delta_1^{(2)} = -\frac{1}{2}\frac{\partial((d_1 - y_1)^2 + (d_2 - y_2)^2)}{\partial s_1^{(2)}} = -\frac{1}{2}\frac{\partial(d_1 - \text{sgm}(s_1^{(2)}))^2}{\partial s_1^{(2)}} - \frac{1}{2}\frac{\partial(d_2 - \text{sgm}(s_2^{(2)}))^2}{\partial s_1^{(2)}}. \tag{2.7}$$

Observando-se que d_1 e $s_1^{(2)}$ são independentes e que o segundo termo da equação anterior é nulo, obtém-se,

$$\delta_1^{(2)} = -(d_1 - \text{sgm}(s_1^{(2)}))\frac{\partial(-\text{sgm}(s_1^{(2)}))}{\partial s_1^{(2)}} = (d_1 - \text{sgm}(s_1^{(2)}))\ \text{sgm}'(s_1^{(2)}). \tag{2.8}$$

Definindo-se $\epsilon_1^{(2)} = d_1 - \text{sgm}(s_1^{(2)})$, a equação 2.8 passa a ser dada por:

$$\delta_1^{(2)} = \epsilon_1^{(2)}\ \text{sgm}'(s_1^{(2)}) \tag{2.9}$$

O cálculo de $\delta_2^{(2)}$, para a segunda unidade da camada de saída é feito de forma análoga. Então, o valor de δ na camada de saída é calculado multiplicando-se o erro de saída do elemento j pela derivada da não linearidade de função sigmóide associada.

2.3.3.2 Cálculo da Variação δ para a Camada 1

Para o primeiro elemento da camada 1, tem-se

$$\delta_1^{(1)} = -\frac{1}{2}\frac{\partial \varepsilon^2}{\partial s_1^{(1)}}. \tag{2.10}$$

[4] A nomenclatura utilizada estabelece os índices superior e inferior como camada e posição da unidade processadora, respectivamente.

Aplicando-se a regra da cadeia e considerando-se que ε^2 é completamente determinado por $s_1^{(2)}$ e $s_2^{(2)}$, obtém-se:

$$\delta_1^{(1)} = -\frac{1}{2}\left(\frac{\partial \varepsilon^2}{\partial s_1^{(2)}}\frac{\partial s_1^{(2)}}{\partial s_1^{(1)}} + \frac{\partial \varepsilon^2}{\partial s_2^{(2)}}\frac{\partial s_2^{(2)}}{\partial s_1^{(1)}}\right).$$

Utilizando-se as definições de $\delta_1^{(2)}$ e $\delta_2^{(2)}$, e substituindo-se a versões expandidas das saídas do combinador linear, tem-se $\delta_1^{(1)}$ dado por:

$$\begin{aligned}\delta_1^{(1)} = \delta_1^{(2)}\frac{\partial s_1^{(2)}}{\partial s_1^{(1)}} + \delta_2^{(2)}\frac{\partial s_2^{(2)}}{\partial s_1^{(1)}} &= \delta_1^{(2)}\frac{\partial}{\partial s_1^{(1)}}\left(w_{10}^{(2)} + \textstyle\sum_{i=1}^{3} w_{1i}^{(2)}\ \mathrm{sgm}(s_i^{(1)})\right) \\ &+ \delta_2^{(2)}\frac{\partial}{\partial s_1^{(1)}}\left(w_{20}^{(2)} + \textstyle\sum_{i=1}^{3} w_{2i}^{(2)}\ \mathrm{sgm}(s_i^{(1)})\right).\end{aligned}$$

Nota-se que $\partial[\mathrm{sgm}(s_i^{(n)})]/\partial s_j^{(n)} = 0$ se $i \neq j$. Portanto,

$$\delta_1^{(1)} = \delta_1^{(2)} w_{11}^{(2)}\ \mathrm{sgm}'(s_1^{(1)}) + \delta_2^{(2)} w_{21}^{(2)}\ \mathrm{sgm}'(s_1^{(1)}) = [\delta_1^{(2)} w_{11}^{(2)} + \delta_2^{(2)} w_{21}^{(2)}]\ \mathrm{sgm}'(s_1^{(1)})$$

Assumindo-se

$$\epsilon_1^{(1)} = \delta_1^{(2)} w_{11}^{(2)} + \delta_2^{(2)} w_{21}^{(2)}, \tag{2.11}$$

obtém-se,

$$\delta_1^{(1)} = \epsilon_1^{(1)}\ \mathrm{sgm}'(s_1^{(1)}) \tag{2.12}$$

Observando-se as linhas pontilhadas da figura 2.9, nota-se que $\delta_1^{(1)}$ é calculado de acordo com as equações 2.11 e 2.12.

O procedimento para o cálculo dos valores $\delta^{(n)}$, nas camadas intermediárias, envolve a multiplicação de cada derivada $\delta^{(n+1)}$ (associada a cada elemento do nível posterior) pelo peso correspondente. Estas derivadas, ponderadas pelos pesos, são então somadas produzindo-se o termo de erro $\epsilon^{(n)}$, o qual é multiplicado por $\mathrm{sgm}'(s^{(n)})$.

Esta ilustração foi feita para uma rede de dois níveis de unidades processadores. No entanto, para o caso de três níveis o processo de retropropagar as derivadas instantâneas do erro quadrático, de um nível para o anterior, é análoga. O processo é realizado até que se calculem as variações δ para todos os elementos processadores.

2.4 Rede Neural para Processamento Simbólico

O modelo de neurônio descrito por *Rocha* [Roc92], é definido como um elemento processador complexo que combina processamentos elétricos e químicos. Este dois tipos de processamentos são definidos pela dinâmica do acoplamento de transmissores e receptores

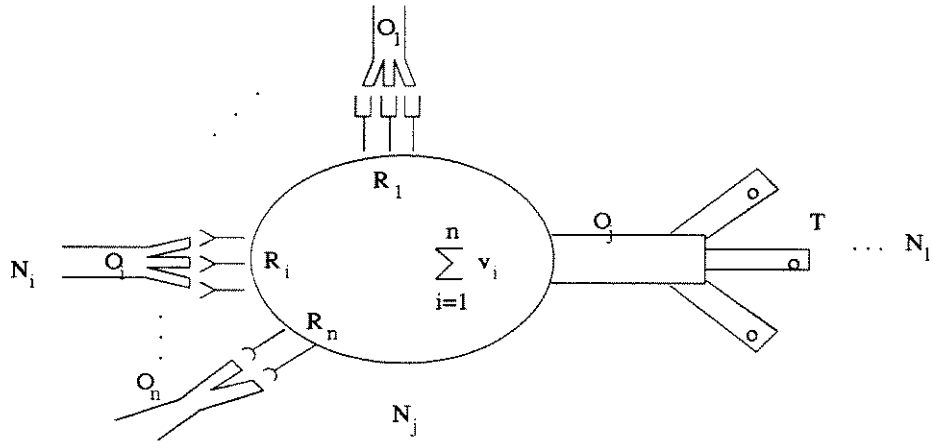


Figura 2.10: Modelo do neurônio.

realizado nas sinapses. Deste modo, a utilização de um conjunto destes elementos processadores permite a obtenção de uma rede neural capaz de realizar tanto processos numéricos quanto simbólicos.

2.4.1 Modelo do Neurônio Formal

O neurônio N_j, ilustrado na figura 2.10, pode ser definido como a estrutura a seguir:

$$N_j = \{O_p, O_j, T, R, C, \Theta, \alpha, g\}$$

onde,

- $O_p (p = 1, \cdots, n)$ é o conjunto de entradas pré-sinápticas atuando em N_j por todos os seus n axônios pré-sinápticos.
- O_j é o código da saída de N_j.
- T é o conjunto de transmissores usados por N_j para trocar mensagens com outros neurônios.
- R é o conjunto de receptores que se acoplam aos transmissores $t_i \in T_p$, com T_p definindo o conjunto de transmissores realizados pelos neurônios pré-sinápticos. A força da conexão w_i com o i$^{\text{ésimo}}$ neurônio pré-sináptico é dada por:

$$w_i = (M(t) \wedge M(r)) * \mu(t, r) \odot v_0 \tag{2.13}$$

onde $M(t)$ é a quantidade de transmissores na célula pré-sináptica N_i; $M(r)$ é a quantidade de receptores r disponíveis para se acoplarem com t; $\mu(t,r)$ é a afinidade do acoplamento $t \wedge r$; e v_0 representa a distribuição espacial das sinapses. Os operadores $\circ, \wedge, *$ e $\odot$ podem ser definidos como normas triangulares ou co-mormas.

A atividade do neurônio pós-sináptico N_j, realizada pela ação do neurônio pré-sináptico N_i, pode ser considerada

$$v_i = O_i \circ w_i \tag{2.14}$$

para w_i dado pela eq. 2.13; e O_i definido como a saída codificada pelo neurônio n_i.

- Θ é a função usada (geralmente um somatório) para agregar a atividade pós-sináptica atual

$$a_j = \Theta(v_i) = \sum_{i=1}^{n} O_i \circ w_i \tag{2.15}$$

- α representa o conjunto de limiares e g a função de codificação, para uma saída codificada O_j dada por:

$$O_j = \begin{cases} O^i & \text{se } a_j < \alpha_1 \\ O^u & \text{se } a_j \geq \alpha_2 \\ g(a_j) & \text{de outro modo} \end{cases} \tag{2.16}$$

- C é o conjunto de controladores que podem ser ativados pelo acoplamento (t/r), onde o elemento c_j é dado pela relação

$$t_i \wedge r_j \mapsto c_j$$

para $r_j \in R$; $t_i \in T_p$; e $c_j \in C$.

Como será visto na seção 2.4.3 cada controlador c_j exerce diferentes ações, seja sobre o próprio neurônio N ou sobre os seus vizinhos.

2.4.2 Atividades Realizadas pelo Neurônio

A estrutura do neurônio, mostrada na seção anterior, permite a definição deste como um processador de propósitos gerais, cuja programação depende da especificação da função g e dos limiares α, da quantidade de transmissores realizados, e do nível de ativação v. Como mostra a figura 2.11, as principais ações realizadas pelo neurônio são:

1. **Associação**

   Seja R a população total de receptores pós-sinápticos do neurônio N_j. R é a família

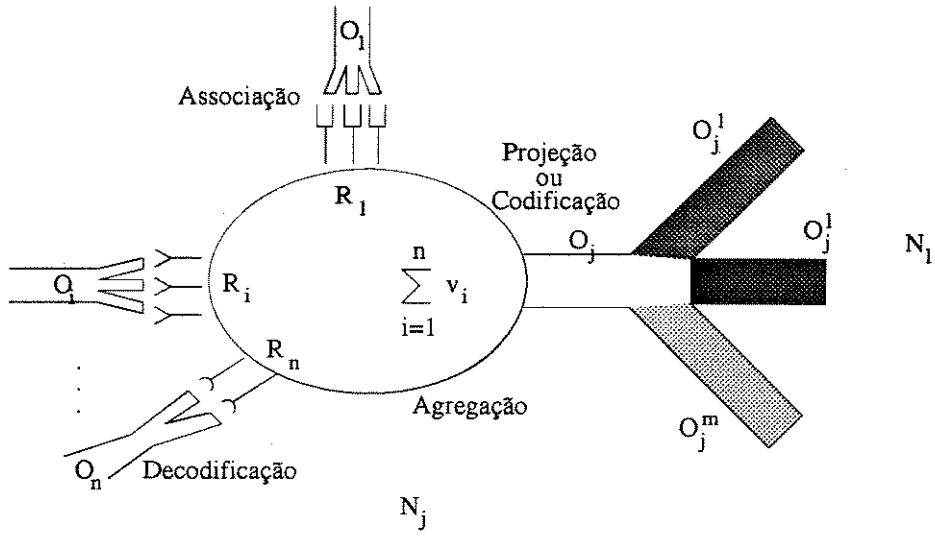


Figura 2.11: Neurônio N_j como um processador.

dos subconjuntos R_i de receptores especializados em se acoplarem a diferentes transmissores t_i. O termo v_0 na eq. 2.13, produzida pela ativação do receptor $r_i \in R_i$, depende não só da distribuição espacial destes receptores, como também do tipo de receptores e da dinâmica do acoplamento (t/r). Então, cada subconjunto R_i representa um tipo possível de ativação em N_j (ver fig. 2.11), e v_i pode ser visto como a medida de compatibilidade entre a saída codificada pelo neurônio pré-sináptico e a máxima ativação possível $M(t) \wedge M(r)$. Portanto, aplicando-se 2.13 a 2.14, tem-se

$$v_i = O_i \circ [(M(t) \wedge M(R)) * \mu(t,r) \odot v_0]. \tag{2.17}$$

A equação 2.17 representa uma generalização da ativação do modelo tradicional onde tem-se v_i dado por:

$$v_i = O_i.w_i$$

onde o operador (.) é dado pelo produto algébrico.

2. **Agregação**

   As diferentes atividades nos distintos terminais pré-sinápticos são agregadas no axônio como uma conseqüência das propriedades elétricas do neurônio [Roc92]. Geralmente, o resultado a_j desta agregação é obtido pela somatória dos diferentes níveis de ativação v_i determinados pelos terminais pré-sinápticos, como mostra a equação 2.15.

3. **Projeção ou Codificação**

   A atividade a_j agregada no corpo do axônio é codificada em O_j de acordo com a equação 2.16.

O modelo do neurônio formal pode exibir propriedades de filtragem diferentes para os vários ramos do axônio. Isto significa que a saída O_j pode ser particionada em subconjuntos O_{jr} $(r = 1, \cdots, m)$, dependendo das propriedades de filtragem dos ramos do axônio, como ilustra a figura 2.11.

Este tipo de projeção difere da proposição de redes neurais artificiais onde a ativação no axônio se difunde, igualmente, sobre todos os terminais de cada neurônio na rede [Roc92].

4. **Decodificação**

Cada terminal pré-sináptico faz diferentes contatos com a célula pós-sináptica. Este padrão do ramo do terminal é um dos principais fatores na determinação da distribuição total de transmissores $M(t)$ dentro do neurônio pré-sináptico. A saída O_i codificada pelo neurônio pré-sináptico é transformada, em cada terminal p_i, em uma quantidade m_i de transmissores realizados na sinapse, conforme a equação a seguir:

$$m_i = O_i \circ M(t). \tag{2.18}$$

Então, considerando-se os operadores como o um tipo único de norma, por exemplo a norma-t *produto algébrico*, tem-se a equação 2.17 redefinida como:

$$v_i = m_i \wedge M(r) * \mu(t, r) \odot v_0. \tag{2.19}$$

onde $\wedge, *$ e $\odot$ indicam um tipo único de operação.

2.4.3 Ação dos Controladores

O transmissor t_i liberado pela célula pré-sináptica N_i, ao se acoplar ao receptor r_j da célula pós-sináptica N_j, pode ativar uma ou mais moléculas controladoras c_j de acordo com a equação a seguir:

$$t_i \wedge r_j \mapsto c_j : \text{ ação } \in A. \tag{2.20}$$

Estes controladores podem exercer distintas ações, tanto no próprio neurônio N_j quanto nos seus vizinhos. As ações exercidas pelo controlador c_j podem ser de controle qualitativo ou quantitativo. Assim por exemplo:

- A saída pré sináptica O_i é recodificada em pulsos m_i de transmissores $t_i \in T_p$ para agirem nos receptores $r_j \in R$ do neurônio pós-sináptico N_j. A quantidade m_i de transmissores t_i, realizados pelo neurônio pré-sináptico N_i, pode ser definida pelas

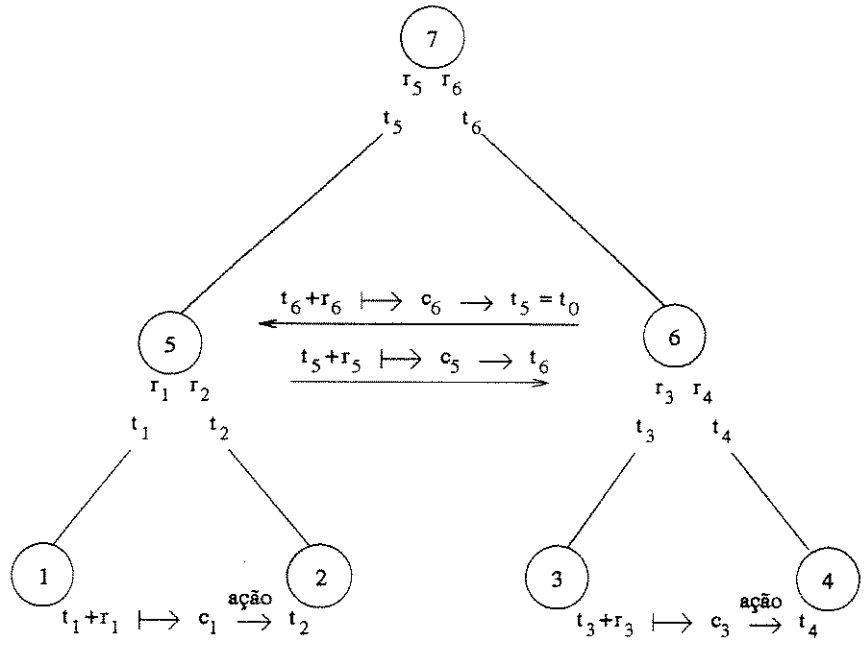


Figura 2.12: Processamento simbólico.

moléculas de controladores c_k liberadas pela célula pós-sináptica (efeito retroativo) ou por outros neurônios vizinhos:

$$m_i = a(c_k) \circ (O_i \circ M(t_i)), \qquad k \in 1, 2, \cdots, j, \cdots, k$$

onde $a(c_k)$ representa a ação do controlador c_k sobre a quantidade de transmissores realizados m_i.

- O controlador c_j pode atuar, ainda de forma quantitativa, na função de codificação do próprio neurônio N_j. Neste caso, a saída codificada passa a ser definida por:

$$O_j = a(c_j) \circ \Theta_{i=1}^{n}(O_i \circ w_i)$$

onde O_i é a saída codificada no neurônio N_i e w_i é a força da conexão com N_i.

- O controle qualitativo do controlador c_j pode alterar o tipo de molécula controladora.

$$t_j + r_j \rightarrow c_k = r_k.$$

2.4.4 Processamento Simbólico

A rede neural, cujo elemento processador é dado pelo modelo de neurônio definido na seção 2.4.1, permite tanto o processamento numérico quanto o processamento simbólico. A figura 2.12 ilustra o modelo da rede que processa o algoritmo decrito a seguir:

ALGORITMO

Se N_1 é ativado $(t_1 \wedge r_1 \mapsto c_1)$ então
.
.
definir o transmissor de N_2 como t_2
N_2 é habilitado a interagir com N_5
.
Se N_5 é ativado $(t_5 \wedge r_5 \mapsto c_5)$ então
.
.
definir o transmissor de N_6 como t_6
N_6 é habilitado a interagir com N_7 pois N_7 tem r_6
.
Se N_3 é ativado $(t_3 \wedge r_3 \mapsto c_3)$ então
.
.
definir o transmissor de N_4 como t_4
N_4 é habilitado a interagir com N_6 pois N_6 tem r_4
.
Se N_6 é ativado $(t_6 \wedge r_6 \mapsto c_6)$ então
.
.
definir o transmissor de N_5 como t_0
N_5 **não** está habilitado a interagir com N_7
pois N_7 não tem r_0
.
fim se
fim se
fim se
fim se

Capítulo 3

Métodos de Interpolação

3.1 Introdução

Uma função $g(x)$ interpola um dado conjunto de $n+1$ pontos, se o gráfico de $g(x)$ passa através de cada um desses pontos. Isto significa que a classe de funções $g^*(x)$ deve satisfazer a cada uma das seguintes condições de interpolação:

$$g^*(x_i) = f(x_i), \qquad 0 \leq i \leq n. \tag{3.1}$$

onde $f(x)$ é a função que se deseja interpolar e x_i são pontos dados. Serão consideradas duas classes $g^*(x)$ - funções polinomiais e funções polinomiais por partes. Em ambas as análises será imposta a restrição de que os pontos devem ser distintos:

$$x_0 < x_1 < \ldots < x_n. \tag{3.2}$$

A seção 3.2 descreve a interpolação polinomial onde o polinômio p_n pode ser calculado de duas formas diferentes - representação de Lagrange (seção 3.2.1) e representação de Newton (3.2.2). A seção (3.2.3) analisa o erro cometido ao se usar p como uma aproximação de f. Para a classe de funções polinomiais por partes serão analisados os casos locais (seções 3.3.1 e 3.3.2), onde o polinômio interpolador é obtido somente com base na influência dos pontos, dentro do subintervalo de interpolação $[x_{i-1}, x_i]$; e também os casos globais (seção 3.3.3) onde a influência se dá por todos os pontos do intervalo de interpolação.

3.2 Interpolação Polinomial

Polinômios algébricos são a classe mais importante de funções de interpolação encontradas na literatura matemática, pois são facilmente somados, multiplicados, integrados ou diferenciados. Entretanto, como será visto posteriormente, apresentam sérios problemas do ponto de vista de precisão e eficiência.

Seja a classe de funções g^* formada pelo conjunto de todos os polinômios p_n de grau $\leq n$, ou seja, $g(x) \in g^*, g(x) = p_n$ e tem a forma:

$$g(x) = p_n(x) = a_0 + a_1 x + \ldots + a_n x^n. \tag{3.3}$$

Teorema 3.2.1 *Dado um conjunto de $n+1$ pontos $(x_i, f(x_i))$, $0 \leq i \leq n$, existe um polinômio $p(x)$ de grau $\leq n$, que interpola $f(x_i)$ em x_i. Este polinômio é único entre o conjunto de todos os polinômios de grau máximo n* [Atk89].

Prova: Supõe-se $f(x)$ uma função real definida em cada um dos $n+1$ valores reais distintos $x_0 < x_1 < \ldots < x_n$. Seja $p_m(x)$ o polinômio de grau m, que interpola $f(x)$ em $[x_0, x_n]$, dado por:

$$p_m(x) = a_0 + a_1 x + \ldots + a_m x^m$$

Impondo-se as $n+1$ condições de interpolação (eq. 3.1) a $p_m(x)$, para $m = n$ tem-se:

$$\begin{aligned} a_0 + a_1 x_0 + \ldots + a_n x_0^n &= f(x_0) \\ a_0 + a_1 x_1 + \ldots + a_n x_1^n &= f(x_1) \\ &\vdots \\ a_0 + a_1 x_n + \ldots + a_n x_n^n &= f(x_n) \end{aligned} \tag{3.4}$$

Portanto, o sistema de $n+1$ equações lineares fica, na forma de matriz, reduzido a:

$$Xa = y,$$

onde:

$$\begin{aligned} X &= [x_i^j] \qquad i, j = 0, 1, \ldots, n \\ a &= [a_0, a_1, \ldots, a_n]^T \\ y &= [f(x_0), \ldots, f(x_n)] \end{aligned}$$

A matriz X é chamada de **Matriz de Vandermonde**. Sabe-se que o seu determinante é diferente de zero (solução única) se os pontos forem distintos, isto é, $x_i \neq x_j$ para $i \neq j$. Esta

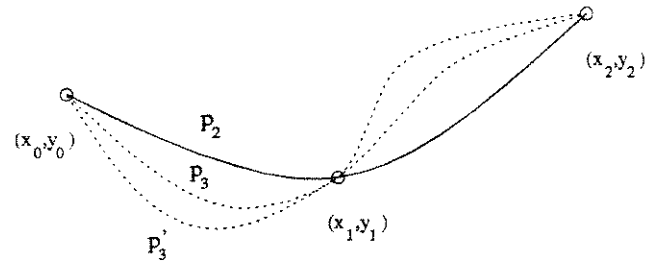


Figura 3.1: Exemplos de interpolação para diferentes graus do polinômio $p_m(x)$. Linha cheia: $m = 2$; linha pontilhada: $m = 3$.

condição está garantida pela restrição inicial (eq. 3.2), portanto, existe um único polinômio $p_n(x)$ que passa pelos pontos $(x_i, f(x_i))$, $0 \leq i \leq n$.

Nesta análise fica claro por que se deve escolher $m = n$. Se $m > n$, o sistema linear correspondente é indeterminado, ou seja, p_n não é único. Por outro lado, se $m < n$, o sistema não apresenta solução para o problema de interpolação[1] a menos que os pontos estejam distribuídos de forma a atender a algumas restrições. A figura 3.1 ilustra o caso de três pontos $x_0 < x_1 < x_2$ e três polinômios de interpolação diferentes, p_2, p_3 e p_3'. Neste caso, escolhe-se $m = 2$. Se, ao contrário, fosse escolhido um grau $m > n$ $(m = 3)$, existiria mais de uma solução. Para o caso de $m = 1$, não haveria solução, a menos que os pontos estivessem alinhados.

O polinômio p_n pode ser obtido pela solução do sistema linear $Xa = y$. Entretanto, existem várias razões para se evitar este método, entre as quais destaca-se o fato de serem estes sistemas extremamente mal-condicionados. Há várias formas de representação do polinômio p_n. Neste trabalho serão descritas as duas formas mais usuais de representação - Fórmula de Lagrange e Fórmula de Newton.

A seção de erro, traz uma análise do erro cometido ao se interpolar $f(x)$ por um polinômio $p_n(x)$, nos $n + 1$ pontos dados $(x_0 < x_1 < \ldots < x_n)$. Uma outra forma de interpolação polinomial é dada pelos polinômios de Hermite, e será descrita na seção 3.2.4.

[1] o caso de $m < n$ pode ser resolvido através de quadrados mínimos, o que no entanto, não atende às restrições dadas na equação 3.1

3.2.1 Fórmula de Lagrange

Uma maneira de se representar $p_n(x)$ é através da fórmula de Lagrange. A solução do problema de interpolação é formada pela combinação linear de $n+1$ polinômios l_i :

$$p_n(x) = y_0 l_0(x) + y_1 l_1(x) + \ldots + y_n l_n(x) = \sum_{i=0}^{n} y_i l_i(x). \tag{3.5}$$

onde $l_i(x)$ deve atender às seguintes condições:

1. l_i deve possuir grau $\leq n$
2. $l_i(x_i) = 1$
3. l_i deve ter n zeros x_j, $j \neq i$

Da condição 3 tem-se:

$$l_i(x) = c(x - x_0)\ldots(x - x_{i-1})(x - x_{i+1})\ldots(x - x_n). \tag{3.6}$$

Aplicando-se a condição 2 à equação 3.6,

$$c = \frac{1}{\prod_{j\neq i=0}^{n}(x_i - x_j)},$$

portanto,

$$l_i = \prod_{j\neq i=0}^{n} \frac{(x - x_j)}{(x_i - x_j)}, \qquad i = 0, \ldots, n.$$

A equação 3.5 define a fórmula de Lagrange para o polinômio de interpolação $p_n(x)$. A principal desvantagem desta fórmula aparece quando se insere um ponto extra da curva original, à base de dados. O novo polinômio $p_{n+1}(x)$ não pode ser obtido facilmente de p_n, pois além do cálculo de $l_{n+1}(x)$, as fórmulas das bases canônicas l_i, $i = 0, \ldots, n$, também devem ser alteradas.

3.2.2 Fórmula de Newton

A fórmula de Newton faz uso das diferenças divididas para obter, recursivamente, polinômios de ordem mais alta. Esta forma de interpolação é bastante conveniente para o caso onde os pontos estão igualmente espaçados.

O polinômio que interpola os $n+1$ pontos distintos $x_0, x_1, \ldots, x_n$, pode ser obtido do polinômio anterior, acrescido de um termo de correção $c_n(x)$:

$$p_n(x) = p_{n-1}(x) + c_n(x). \tag{3.7}$$

Cálculo do Termo de Correção $c_n(x)$

$$c_n(x) = p_n(x) - p_{n-1}(x),$$

$$c_n(x_i) = p_n(x_i) - p_{n-1}(x_i) = f(x_i) - f(x_i) = 0 \qquad i = 0, \ldots, n-1.$$

Conclui-se, portanto, que $c_n(x)$ possui no mínimo n zeros:

$$c_n(x) = a_n(x)(x - x_0)\ldots(x - x_{n-1}). \tag{3.8}$$

Aplicando-se a equação 3.8 à equação 3.7 obtém-se:

$$p_n(x) = p_{n-1}(x) + a_n(x)(x - x_0)\ldots(x - x_{n-1})$$

Definindo $p_0 = a_0 = f(x_0)$,

$$\begin{aligned} p_1(x) &= a_0 + a_1(x - x_0) \\ p_2(x) &= a_0 + a_1(x - x_0) + a_2(x - x_0)(x - x_1) \\ &\vdots \\ p_n(x) &= a_0 + a_1(x - x_0) + \ldots + a_n(x - x_0)\ldots(x - x_{n-1}) \end{aligned} \tag{3.9}$$

e substituindo-se, sucessivamente, os valores dos nós x_i, $i = 0, \ldots, n$, na equação de p_n (eq. 3.9), verifica-se que o cálculo de a_0 envolve somente $f(x_0)$, o cálculo de a_1 envolve $f(x_0)$ e $f(x_1)$, e assim por diante. Portanto, a_n pode ser definido por:

$$a_n = f[x_0, x_1, \ldots, x_n],$$

e a equação 3.9 pode ser reescrita como:

$$p_n(x) = a_n x^n + a'_{n-1} x^{n-1} + \ldots + a'_0. \tag{3.10}$$

Da fórmula de Lagrange obtém-se:

$$p_n(x) = \sum_{i=0}^{n} f(x_i) \prod_{\substack{j=0 \\ j \neq i}}^{n} \frac{(x - x_j)}{(x_i - x_j)} = \sum_{i=0}^{n} M_i \prod_{\substack{j=0 \\ j \neq i}}^{n} (x - x_j), \tag{3.11}$$

onde

$$M_i = \frac{f(x_i)}{\prod_{\substack{j=0 \\ j \neq i}}^{n} (x_i - x_j)}.$$

Expandindo o produtório da equação 3.11 e rearranjando, tem-se:

$$p_n(x) = \sum_{i=0}^{n} M_i b^n x^n + \sum_{i=0}^{n} M_i b^{n-1} x^{n-1} + \ldots + \sum_{i=0}^{n} M_i b^0. \tag{3.12}$$

onde, $\sum_{i=0}^{n} M_i b^k$ é o coeficiente de x^k e $b^n = 1$. Comparando-se 3.10 e 3.12 obtém-se:

$$a_n = \sum_{i=0}^{n} M_i = \sum_{i=0}^{n} \frac{f(x_i)}{\prod_{\substack{j=0 \\ j \neq i}}^{n} (x_i - x_j)} = f[x_0, \ldots, x_n] \tag{3.13}$$

Portanto, o polinômio $p_n(x)$ pode ser definido, em termos das diferenças divididas, como:

$$p_n(x) = f[x_0] + (x - x_0) f[x_0, x_1] + \ldots + (x - x_0) \ldots (x - x_{n-1}) f[x_0, \ldots, x_n] \tag{3.14}$$

A literatura [Atk89] e [IK66] abrange largamente o estudo de formas computacionais mais eficientes destes coeficientes.

A principal vantagem da fórmula de Newton em relação à representação de Lagrange é a facilidade de se inserirem novos pontos, uma vez que o novo polinômio interpolador pode ser obtido do anterior, acrescentando-se o termo de correção.

3.2.3 Análise de Erro

A análise do erro de interpolação ($\| f - p_n \|$) com o qual o polinômio interpolante aproxima a função $f(x)$, não depende somente do conhecimento dos valores de f nos nós[2]. Por exemplo, considerando os $n+1$ pontos $(x_0, f(x_0)), \ldots, (x_n, f(x_n))$, onde $x_0 < x_1 < \ldots < x_n$, as condições de interpolação definem o comportamento do polinômio somente nos pontos x_i $i = 0, \ldots, n$. Nenhuma restrição é imposta ao longo do intervalo $[x_{i-1}, x_i]$. Deste modo, a norma do erro ($\| f(x) - p_n(x) \|$) pode ser arbitrariamente grande fora dos nós de interpolação. Visto isso, a análise de erro torna necessário um conhecimento acerca do comportamento da função $f(x)$, no intervalo considerado. A norma do erro pode ser determinada, por exemplo, em termos da $(n+1)^{\text{ésima}}$ derivada de f, se esta existir [Atk89].

Teorema 3.2.2 *Seja $[a, b]$ um intervalo qualquer que contenha todos os $n+1$ pontos $x_0, \ldots, x_n$. Seja $f \in C^n[a, b]$, onde C^n é o espaço das funções que possuem n derivadas contínuas, e f possua derivada de ordem $n+1$ para $a < x < b$. Então, dado um $x \in [a, b]$, existe um número ξ_x em (a, b), tal que:*

$$f(x) - p_n(x) = (x - x_0) \ldots (x - x_n) \frac{f^{(n+1)}(\xi_x)}{(n+1)!} \tag{3.15}$$

Prova: Da fórmula de Newton (eq. 3.14) tem-se:

$$p_n(x) = p_{n-1}(x) + a_n \Psi_{n-1}(x)$$

[2] Para $h(t)$ contínua em $[a, b]$, o valor da função real $\| \|$, definido por $\| h \| = \max_{a \leq t \leq b} | h(t) |$, é denominado **norma de Tchebycheff de h**

onde,

$$a_n = f[x_0, x_1, \ldots, x_n] \quad \text{e} \quad \Psi_{n-1} = (x - x_0)\ldots(x - x_{n-1}).$$

Seja $t \in [a, b]$ um ponto qualquer diferente de $x_0, \ldots, x_n$. Então, o polinômio $p_{n+1}(x)$ que interpola $f(x)$ nos $n + 2$ pontos $x_0, \ldots, x_n$,t é dado por:

$$p_{n+1}(x) = p_n(x) + \lambda\Psi_n(x).$$

Calculando $p_{n+1}(x)$ em $x = t$, tem-se:

$$f(t) = p_{n+1}(t) = p_n(t) + \lambda\Psi_n(t)$$

e para t arbitrário,

$$E(p_n(x)) = f(x) - p_n(x) = \lambda(x - x_0)\ldots(x - x_n).$$

Desde que o erro $E(p_n(x))$ e $\Psi_n(x)$ têm zeros nos $n + 1$ pontos $x_0, \ldots, x_n$, a função

$$g(x) = f(x) - p_n(x) - \lambda\Psi_n(x) \tag{3.16}$$

possui, no mínimo $n + 1$ zeros, para λ dado por uma constante qualquer. Para se estimar o erro no ponto $x = \alpha \neq x_0, \ldots, x_n$ em $[a, b]$, escolhe-se λ de modo que $g(\alpha) = 0$, ou seja, g tenha no mínimo $n + 2$ zeros. Conseqüentemente,

$$\lambda = \frac{f(\alpha) - p_n(\alpha)}{(\alpha - x_0)\ldots(\alpha - x_n)}$$

e a função g passa a ser definida por:

$$g(x) = f(x) - p_n(x) - \frac{(x - x_0)\ldots(x - x_n)}{(\alpha - x_0)\ldots(\alpha - x_n)}(f(\alpha) - p_n(\alpha)) \tag{3.17}$$

Teorema 3.2.3 (Teorema de Rolle) *Se $f \in C[a, b]$ e é diferenciável em (a, b), então existe, no mínimo, um ponto $\xi \in (a, b)$, tal que, $f'(\xi) = 0$.*

Considerando-se os $n + 2$ pontos $x_0, \ldots x_n, \alpha$, onde g se anula, arranjados de forma ordenada e aplicando-se o teorema 3.2.3(citado em [Atk89]) em cada um dos $n + 1$ subintervalos, conclui-se que g' possui no mínimo $n + 1$ zeros. Aplicando-se, novamente, o teorema anterior, observa-se que g'' tem no mínimo n zeros. Continuando a análise, conclui-se que $g^{(n+1)}$ possui no mínimo um zero em (a, b) que pode ser definido por ξ_x. Diferenciando-se a equação 3.17, $n + 1$ vezes, obtém-se:

$$g^{(n+1)}(x) = f^{(n+1)}(x) - \frac{(n+1)!}{(\alpha - x_0)\ldots(\alpha - x_n)}(f(\alpha) - p_n(\alpha)), \tag{3.18}$$

dado que $p^{(n+1)} = 0$ e o termo x^{n+1} é reduzido a $(n+1)!$ no produto $(x-x_0)\ldots(x-x_n)$. Substituindo-se ξ_x em 3.18 o valor da derivada de ordem $n+1$ é nulo ($g^{(n+1)}(\xi_x) = 0$) e, para α arbitrário,

$$f(x) - p_n(x) = (x-x_0)\ldots(x-x_n)\frac{f^{(n+1)}(\xi_x)}{(n+1)!}.$$

Portanto, para se estimar o erro é necessário o cálculo de $f^{(n+1)}$. Normalmente não se conhece o valor de ξ_x e trabalha-se com o limitante:

$$E(p_n(x)) \leq \max\left(\frac{f^{(n+1)}(\xi_x)}{(n+1)!}\right) \max \prod_{i=0}^{n}(x-x_i). \tag{3.19}$$

O problema da Aproximação

Considera-se a aproximação de uma dada função $f(x)$, em um determinado intervalo $[a,b]$, por uma interpolação polinomial. Em particular, considera-se o intervalo de interpolação igualmente espaçado.

Para cada $n \geq 1$, define-se

$$h = (b-a)/n, \qquad x_j = a + jh, \qquad j = 0, 1, \cdots, n.$$

Seja $p_n(x)$ o polinômio que interpola $f(x)$ em $x_0, \ldots, x_n$ e o erro de interpolação dado por:

$$\max_{a \leq x \leq b} \mid f(x) - p_n(x) \mid . \tag{3.20}$$

Existe uma série de funções, inclusive bem comportadas, para as quais o erro não converge para zero quando $n \to \infty$. O exemplo de não convergência mais famoso foi estudado, primeiramente, por Runge (citado por Prenter [Pre75]). Para a função

$$f(x) = \frac{1}{1+x^2} \qquad -5 \leq x \leq 5,$$

mostra-se, em Isaacson e Keller [IK66], que para $3.64 <\mid x \mid < 5$,

$$\sup_{n \geq k} \mid f(x) - p_n(x) \mid = \infty \qquad \text{e} \qquad k \geq 0.$$

Então, o polinômio $p_n(x)$ não converge para $f(x)$, quando $n \to \infty$, para nenhum desses valores de x. À primeira vista pode parecer não intuitivo, mas a não convergência está baseada no comportamento dos polinômios $y = (x-x_0)(x-x_1)\cdots(x-x_n)$ perto dos extremos de $[a,b] = [x_0, x_n]$. A figura 3.2 ilustra o gráfico do polinômio interpolador de grau 10 da função $f(x)$. Existem situações onde a escolha de uma partição adequada (pontos não igualmente espaçados), evita o efeito oscilatório. Uma distribuição bastante conhecida é dada pela interpolação de Chebyshev [Boo78].

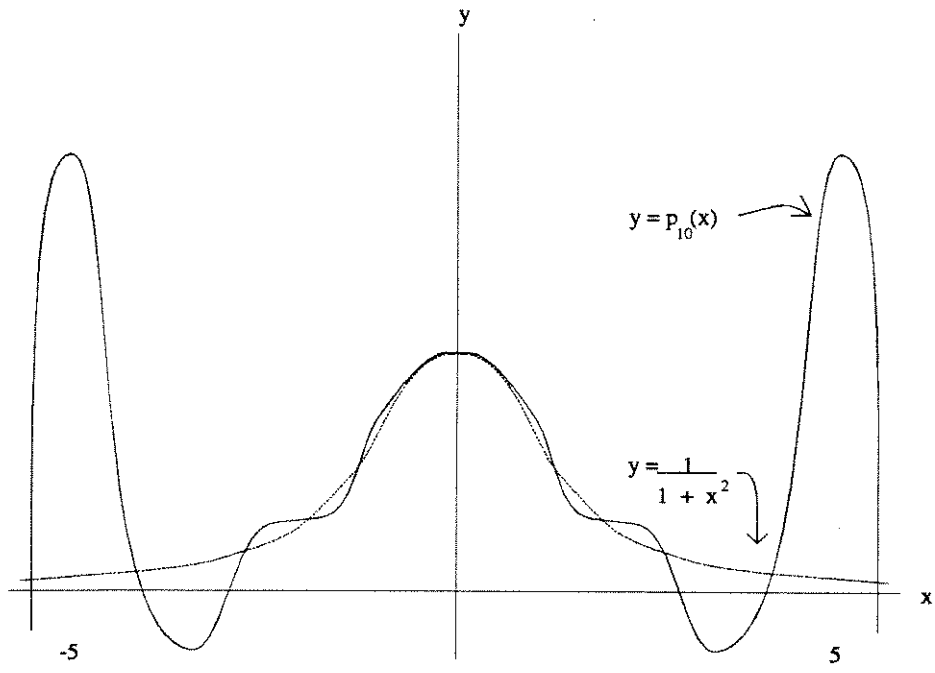


Figura 3.2: Interpolação de $f(x) = 1/(1+x^2)$ pelo polinômio $p_{10}(x)$.

3.2.4 Interpolação de Hermite

A partir deste ponto, a forma de interpolação polinomial dada pelos polinômios de Newton ou Lagrange será chamada, genericamente, de Interpolação de Lagrange. A interpolação de Hermite generaliza a interpolação de Lagrange, uma vez que utiliza um polinômio que não só interpola f em cada nó x_i, mas também interpola um certo número de derivadas consecutivas de f em cada x_i, $i = 1, \ldots, k$. Em particular, dado um conjunto de números reais positivos $x_1 < x_2 < \ldots < x_k$ e $m_1, m_2, \ldots, m_k$, é possível se encontrar um único polinômio $p(x)$, de grau $m_1 + m_2 + \ldots + m_{k-1}$, que soluciona o problema de interpolação:

Encontrar $p(x)$ que satisfaça

$$\begin{aligned} p^{(j)}(x_1) &= f^{(j)}(x_1) \qquad j = 0, 1, \ldots, m_1 - 1. \\ p^{(j)}(x_2) &= f^{(j)}(x_2) \qquad j = 0, 1, \ldots, m_2 - 1. \\ &\vdots \\ p^{(j)}(x_k) &= f^{(j)}(x_k) \qquad j = 0, 1, \ldots, m_k - 1. \end{aligned} \tag{3.21}$$

Definindo-se $N = m_1 + m_2 + \ldots + m_k$, tem-se um único polinômio, entre todos os polinômios de grau $\leq N - 1$, que satisfaz 3.21. Diz-se que $p(x) - f(x)$ tem um zero de ordem m_i em x_i, $i = 1, \ldots, k$, e $p(x)$ é chamado de polinômio de Hermite que interpola f nos

pontos x_i's.

Seja p dado por:

$$p(x) = a_N x^N + a_{N-1} x^{N-1} + \ldots + a_0. \tag{3.22}$$

A prova de existência pode ser obtida aplicando-se as novas condições de interpolação eq. 3.21 à equação 3.22 citeprenter-75. Obtêm-se, desta forma, m_i equações para cada nó x_i, totalizando $N+1$ equações para $N+1$ incógnitas. Para se provar a unicidade, basta mostrar a não singularidade da matriz dos coeficientes do sistema linear obtido.

É interessante observar que uma fórmula específica pode ser obtida quando $m_1 = m_2 = \ldots = m_k = 2$. Em particular seja $l_i(x)$ uma base canônica de ordem $(k-1)$ solucionando o problema $l_i(x_j) = \delta_{ij}$ para $i, j = 1, 2, \ldots, k$, onde:

$$\delta_{ij} = \begin{cases} 0 & \text{se } i \neq j \\ 1 & \text{se } i = j \end{cases}$$

Sejam

$$\phi_i(x) = [1 - 2l_i'(x_i)(x - x_i)]l_i^2(x)$$
$$\psi_i(x) = (x - x_i)l_i^2(x).$$

O polinômio de Hermite p(x) que satisfaz todas as condições de interpolação (3.21) é dado por:

$$p(x) = \sum_{i=1}^{k} f(x_i)\phi_i(x) + \sum_{i=1}^{k} f'(x_i)\psi_i(x).$$

O conjunto de polinômios linearmente independentes $\{\phi_i(x)\}$ e $\{\psi_i(x)\}$ formam uma base para o conjunto de todos os polinômios de Hermite definidos pela partição $x_1 < \ldots < x_k$.

3.2.4.1 Polinômios de Hermite Cúbicos

O polinômio de Hermite de grau 3, dado por $p(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3$, e que soluciona o problema de interpolação

$$p(a) = f(a) \quad p'(a) = f'(a), \qquad p(b) = f(b) \quad p'(b) = f'(b), \tag{3.23}$$

onde a e b sao dois números reais e distintos, pode ser obtido pela união de trechos de polinômios cúbicos de Hermite, como ilustra a figura 3.3. Nota-se que não só os valores da

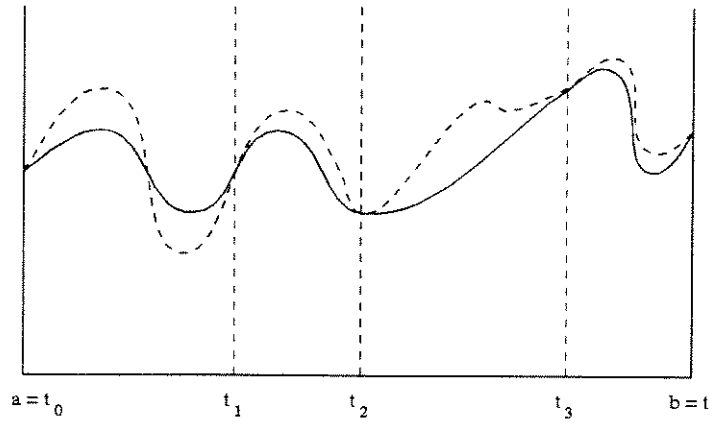


Figura 3.3: Polinômio de Hermite cúbico por partes, interpolando a função f. Linha cheia: s; linha pontilhada: f.

função f coincidem com os valores de p nos nós t_i, mas também os valores de f' e p' são os mesmos nos pontos dados.

Uma maneira de se provar a existência de p é colocá-lo na forma:

$$p(t) = p(a)\phi_1(t) + p(b)\phi_2(t) + p'(a)\psi_1(t) + p'(b)\psi_2(t) \tag{3.24}$$

onde

$$\phi_1(t) = \frac{(t-b)^2[(a-b)+2(a-t)]}{(a-b)^3}$$

$$\phi_2(t) = \frac{(t-a)^2[(b-a)+2(b-t)]}{(a-b)^3}$$

$$\psi_1(t) = \frac{(t-a)(t-b)^2}{(a-b)^2}$$

$$\psi_2(t) = \frac{(t-a)^2(t-b)}{(a-b)^2}$$

e $t_1 = a$, $t_2 = b$. Nota-se que:

$$\begin{cases} \phi_i(t_j) = \delta_{ij} \\ \phi_i'(t_j) = 0 \qquad 1 \leq i,j \leq 2 \end{cases}$$

e

$$\begin{cases} \psi_i(t_j) = 0 \\ \psi_i'(t_j) = \delta_{ij} \qquad 1 \leq i,j \leq 2 \end{cases}$$

A prova da unicidade pode ser encontrada em [Pre75].

3.2.4.2 Análise de Erro

O erro $\| f - p \|$ é estimado de forma similar ao cálculo para a interpolação de Lagrange. Por exemplo, se $f \in C^{N+1}[a,b]$, prova-se que para cada $x \in [a,b]$, existe um ξ em (a,b) tal que:

$$f(x) - p(x) = \frac{f^{N+1}(\xi)}{(N+1)!}(x - x_1)^{m_1}(x - x_2)^{m_2} \ldots (x - x_k)^{m_k} \tag{3.25}$$

onde, $N = (\sum_{i=1}^{k} m_i) - 1$.

Prova: Da equação 3.16, pode-se obter:

$$g(x) = f(x) - p_n(x) - \lambda \Psi_n'(x),$$

onde

$$\lambda = \frac{f(\alpha) - p_n(\alpha)}{(\alpha - x_1)^{m_1} \ldots (\alpha - x_k)^{m_k}} \quad \text{e} \quad \Psi_n'(x) = (x - x_1)^{m_1} \ldots (x - x_k)^{m_k}.$$

Aplicando-se $N+1$ vezes o Teorema de Rolle a $g(x)$, e para $x = \xi_x$, obtém-se:

$$g^{N+1}(\xi_x) = f^{N+1}(\xi_x) - \frac{f(\alpha) - p_n(\alpha)}{(\alpha - x_1)^{m_1} \ldots (\alpha - x_k)^{m_k}}(N+1)!.$$

Para α arbitrário,

$$f(x) - p(x) = \frac{f^{N+1}(\xi)}{(N+1)!}(x - x_1)^{m_1} \ldots (x - x_k)^{m_k}$$

Para se estimar $f^{(j)}(x) - p^{(j)}(x)$ em dois pontos, procede-se de maneira análoga [Pre75] e obtém-se um limitante para o erro dado por:

$$\| f^{(j)}(x) - p^{(j)} \| \leq \frac{\| f^{(2m)} \|}{(2m-j)!\ 2^{2m-[j]}} h^{2m-j} \qquad 0 \leq j \leq 2m \tag{3.26}$$

onde p_n é o polinômio de grau $\leq 2m-1$, que interpola $f \in C^{2m}[a,b]$ e suas derivadas em a e b; $[j] = j$ se j é par; $[j] = j+1$ se j é ímpar e $h = b - a$.

3.3 Interpolação Polinomial por Partes

A interpolação polinomial por partes, ao contrário da interpolação polinomial simples, consiste de segmentos de diferentes polinômios unidos de forma a constituir uma curva contínua. Apesar deste tipo de interpolação requerer uma computação mais complexa, apresenta boas características de precisão e eficiência.

Existe uma grande variedade de tipos de polinômios por partes que se pode obter da classe g^* (eq. 3.1). Estes tipos se difereciam, basicamente, pelo grau de suavidade (diferenciabilidade) da curva interpoladada obtida e pela influência global ou local dos pontos dados. Para os problemas locais, o polinômio $p(x)$ em cada subintervalo $[x_{i-1}, x_i]$ é completamente determinado pelos dados de interpolação (nós), dentro e na vizinhança de $[x_{i-1}, x_i]$. As interpolações de Lagrange e Hermite por partes (seções 3.3.1 e 3.3.2, respectivamente) são exemplos de problemas locais. Os problemas globais, no entanto, determinam a escolha de $p(x)$ em cada subintervalo $[x_{i-1}, x_i]$, com base nos pontos dados ao longo de todo intervalo de interpolação $[a, b]$. Estes problemas, embora mais complicados do ponto de vista computacional, são largamente utilizados, onde o exemplo mais comum é a função spline (seção 3.3.3).

3.3.1 Interpolação de Lagrange por Partes

Nos casos onde a função que se quer interpolar possui comportamento suave, o fenômeno de Runge pode ser evitado. Entretanto, quando isto não acontece, uma solução alternativa pode ser obtida pela união de trechos de polinômios de Lagrange de forma que estes interpolem os pontos dados. A função resultante $s(x)$ é conhecida como polinômio de Lagrange por partes de grau m, e pode ser definida por:

$$s(x) = \begin{cases} a_0 + a_1 x + \ldots + a_m x^m, & x_0 \leq x \leq x_m \\ a_{m+1} + a_{m+2} x + \ldots + a_{2m+1} x^m, & x_m \leq x \leq x_{2m} \\ a_{2m+2} + a_{2m+3} x + \ldots + a_{3m+2} x^m, & x_{2m} \leq x \leq x_{3m} \\ \vdots & \end{cases}$$

onde os termos a_i são constantes a serem determinadas por $s(x_i) = f(x_i)$, $i = 0, 1, \ldots, n$ e o número de pontos (n) deve ser múltiplo de m. Se $m = 1$, $s_1(x) = s(x)$, obtém-se a interpolação de Lagrange linear por partes. Para o caso de $m = 2$, $s_2(x) = s(x)$ é quadrática por partes (figura 3.4). Polinômios de ordem mais alta podem ser obtidos, o que no entanto não garante a diferenciabilidade nos pontos x_i. Entretanto, para problemas que aproximam funções não diferenciáveis, o erro estimado $\| s - f \|$ pode ser relativamente baixo mesmo quando $m = 1$. Uma aplicação bastante conhecida de interpolação de Lagrange linear por partes é a regra trapezoidal de integração numérica.

Para a análise de erro, supondo-se f como sendo duas vezes continuamente diferenciável, e aplicando-se a fórmula do erro obtida na seção 3.2.3,

$$\| f - s_1 \| \leq \frac{\| f'' \|}{4} h^2; \tag{3.27}$$

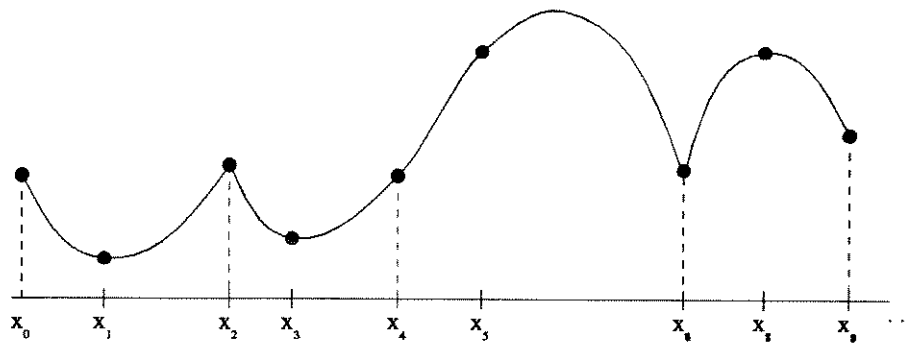


Figura 3.4: Polinômio de Lagrange quadrático por partes.

e para f três vezes continuamente derivável,

$$\| f - s_2 \| \leq \frac{\| f''' \|}{6} h^3;$$

Em ambos os casos, garante-se a convergência da aproximação de f por polinômios (grau 1 ou 2), quando h tende a zero. A equação 3.27, demonstra que mesmo para polinômios de grau 1, obtém-se uma convergência satisfatória (h^2) para o caso de f possuir segunda derivada contínua.

3.3.2 Interpolação de Hermite por Partes

A seção anterior mostra que, embora graus mais elevados de polinômios por partes possam ser obtidos sem contudo incorrer em problemas de oscilação, a curva obtida não é suave na maioria dos casos. Uma solução possível é acrescentar condições de continuidade nas derivadas da função de interpolação 3.21, utilizando-se polinômios de Hermite por partes.

A construção dos polinômios de Hermite por partes $s(x)$, para uma dada função f, com nós π: a $= x_0 < x_1 < \ldots < x_n = b$, é simples quando as bases ϕ_{ij}, $j = 0, 1$ são escolhidas corretamente:

$$\phi_{i0}(x) = \begin{cases} \frac{(x - x_{i-1})^2}{(x_i - x_{i-1})^3}[2(x_i - x) + (x_i - x_{i-1})], & x_{i-1} \leq x \leq x_i \\ \frac{(x_{i+1} - x)^2}{(x_{i+1} - x_i)^3}[2(x_{i+1} - x) + (x_{i+1} - x_i)], & x_i \leq x \leq x_{i+1} \\ 0 & x \ni [x_{i-1}, x_{i+1}] \end{cases}$$

e

$$\phi_{i1}(x) = \begin{cases} \dfrac{(x - x_{i-1})^2(x - x_i)}{(x_i - x_{i-1})^2}, & x_{i-1} \leq x \leq x_i \\ \dfrac{(x - x_{i+1})^2(x - x_i)}{(x_{i+1} - x_i)^2}, & x_i \leq x \leq x_{i+1} \\ 0 & x \ni [x_{i-1}, x_{i+1}] \end{cases}$$

Todos os polinômios $\phi_{ij}(x)$ são unicamente determinados pelas restrições:

$$\begin{cases} \phi_{i0}(x) = \delta_{ij} \\ \phi'_{i0}(x) = 0 \qquad 1 \leq i, j \leq 2 \end{cases}$$

e

$$\begin{cases} \phi_{i1}(x) = 0 \\ \phi'_{i1}(x) = \delta_{ij} \qquad 1 \leq i, j \leq 2 \end{cases}$$

Então, o polinômio de Hermite cúbico por partes é dado por:

$$s(x) = \sum_{i=0}^{n} f(x_i)\phi_{i0}(x) + \sum_{i=0}^{n} f'(x_i)\phi_{i1}(x).$$

3.3.2.1 Análise de Erro

Os polinômios de Hermite (seção 3.2.4) resultam numa interpolação mais suave, mas sofrem da mesma instabilidade (fenômeno de Runge), que os polinômios de Lagrange, nos casos em que as derivadas sucessivas de f se tornam muito grandes e os pontos da partição não estão distribuídos adequadamente. A escolha de polinômios de Hermite por partes elimina essa desvantagem. Esta análise pode ser feita levando-se em conta que a estimativa do erro de Hermite (seção 3.2.4.2) se mantém. Então, para $f \in C^{2m}[a, b]$

$$\| f^{(j)} - s^{(j)} \| \leq \frac{\| f^{(2m)} \|}{(2m - j)!\, 2^{2m-[j]}} h^{2m-j}, \qquad 0 \leq j \leq 2m - 1,$$

com a diferença de que para f fixo, $\| f^{(2m)} \|$ é uma constante independente da partição π, ou seja, f independe de h. Segue que

$$\| f^{(j)} - s^{(j)} \| \to 0 \qquad (0 \leq j \leq 2m - 2),$$

para $h \to 0$.

Verifica-se que para $j = 0$ e $m = 2$ (polinômios cúbicos),

$$\| f - s \| = \frac{h^4}{384} \cdot \max_{x_{i-1} \leq x \leq x_i} | f^{(4)}(t) | \qquad x_{i-1} \leq x \leq x_i.$$

Portanto, a taxa de convergência é h^4 se f possui até quarta derivada contínua.

3.3.3 Splines

O termo funções splines define um desenvolvimento matemático recente para modelar um velho dispositivo mecânico utilizado por desenhistas. As splines mecânicas consistem de tiras flexíveis de material elástico, fixadas por pinos nos pontos de interpolação (nós). A curva spline se estabiliza numa forma final que minimiza a sua energia de potencial, ou seja, o trabalho realizado para produzir a inflexão [ANW67]. A teoria da torção [Sok56] assume que esta energia é proporcional à integral com respeito ao comprimento do arco do quadrado da curvatura da spline. Para o caso de splines de ordem par, com interpolação nos pontos ou nós, prova-se a exitência destes polinômios através da relação básica de integral, obtida de splines cúbicas [Hol57]

$$\int_a^b | f''(x) |^2 \, dx = \int_a^b | s''(x) |^2 \, dx + \int_a^b | f''(x) - s''(x) |^2 \, dx. \tag{3.28}$$

Teorema 3.3.1 (Teorema de Holladay.) *Seja a partição π: $a = x_0 < x_1 < \cdots < x_N = b$ e um conjunto de números reais $\{y_i\}$ $(i = 0, 1, \ldots, N)$ dados. Então, para todas as funções $f(x)$ que possuem primeira e segunda derivadas contínuas em $[a, b]$ e $f(x_i) = y_i$ $(i = 0, \ldots, N)$, a função $s(x)$ com pontos de junção em x_i e com $s''(a) = s''(b) = 0$, minimiza a integral*

$$\int (s''(x))^2 dx. \tag{3.29}$$

Muito da teoria atual de spline teve início com este teorema e com a sua prova. A integral 3.29 é uma boa aproximação da integral do quadrado da curvatura da curva $y = f(x)$, conseqüentemente, o conteúdo do teorema de Holliday é muitas vezes denominado de propriedade de curvatura mínima.

Quando os pontos de junção[3] $(x_1, y_1), \ldots, (x_n, y_n)$ são dados, a curva spline dada pela eq. 3.29 é minimizada e satisfaz às seguintes restrições:

$$s(x_i) = y_i \qquad (i = 1, 2, \ldots, n), \tag{3.30}$$

[3]Define-se como pontos de junção, os pontos limitantes dos trechos (sub-intervalos) de interpolação, e que em alguns casos, não vão coincidir com os pontos de interpolação.

onde s e s' são contínuas em $[x_1, x_n]$. Diz-se portanto, que $s(x)$ é uma spline de ordem $m \geq 1$ se satisfaz às seguintes propriedades:

$$\textbf{(P1)} \quad s(x) \text{ é um polinômio de grau } < m \text{ em cada subintervalo } [x_{i-1}, x_i].$$

$$\textbf{(P2)} \quad s^{(r)}(x) \text{ é contínua em } [a, b], \text{ para } 0 \leq r \leq m-2. \tag{3.31}$$

A teoria da flexão elementar sugere que $s(x)$ seja um polinômio cúbico entre cada par de nós consecutivos, e que os polinômios adjacentes unam-se continuamente com primeira e segunda derivadas contínuas.

3.3.3.1 Splines Cúbicas

Splines cúbicas são as funções splines mais populares por uma série de razões. São funções suaves e, quando usadas para interpolar dados, não apresentam o comportamento oscilatório característico dos polinômios de interpolação de graus elevados. Uma outra razão aparece quando se analisa o limitante do erro cometido ao se interpolar um curva suave por uma spline cúbica (seção 3.3.3.2).

Para o problema de interpolação, deseja-se encontrar uma spline cúbica $s(x)$ que atenda às restrições da equação 3.30. Primeiramente observa-se o número de graus de liberdade existentes na escolha de $s(x)$ de modo a satisfazer 3.30. Escrevendo-se

$$s(x) = a_i + b_i x + c_i x^2 + d_i x^3 \qquad x_{i-1} \leq x \leq x_i \qquad i = 1, \ldots, n, \tag{3.32}$$

obtêm-se $4n$ coeficientes $\{a_i, b_i, c_i, d_i\}$ desconhecidos. As restrições 3.30 em $s(x)$ e as restrições de continuidade de P_2,

$$s^{(j)}(x_i + 0) = s^{(j)}(x_i - 0) \qquad i = 1, \ldots, n-1, \qquad j = 0, 1, 2$$

definem juntas $n + 1 + 3(n-1) = 4n - 2$ restrições, para $4n$ incógnitas. Portanto, existem pelo menos dois graus de liberdade na escolha dos coeficientes de 3.32. Então, são impostas condições extras para se obter uma única spline $s(x)$ que interpole os dados.

Definindo-se

$$M_i = s''(x_i) \qquad e \qquad M_{i+1} = s''(x_{i+1}) \qquad i = 0, \ldots, n$$

e dado que $s(x)$ é cúbica em $[x_{i+1}, x_i]$, tem-se $s''(x)$ linear e dada por:

$$s''(x) = \frac{(x_{i+1} - x)M_i + (x - x_i)M_{i+1}}{h_i} \qquad i = 0, 1, \ldots, n-1, \tag{3.33}$$

onde $h_i = x_{i+1} - x_i$. Conclui-se, portanto, que $s''(x)$ é contínua em $[x_0, x_n]$. Integrando-se 3.33 duas vezes, obtém-se:

$$s(x) = \frac{(x_{i+1} - x)^3 M_i + (x - x_i)^3 M_{i+1}}{6h_i} + C(x_{i+1} - x) + D(x - x_i)$$

com C e D arbitrários. As condições de interpolação 3.30 implicam em:

$$C = \frac{y_i}{h_i} - \frac{h_i M_i}{6} \qquad D = \frac{y_{i+1}}{h_i} - \frac{h_i M_{i+1}}{6}$$

Portanto,

$$s(x) = \frac{(x_{i+1} - x)^3 M_i + (x - x_i)^3 M_{i+1}}{6h_i} + \frac{(x_{i+1} - x) y_i + (x - x_i) y_{i+1}}{h_i}$$

$$- \frac{h_i}{6} [(x_{i+1} - x) M_i + (x - x_i) M_{i+1}] \qquad x_i \leq x \leq x_{i+1} \qquad 0 \leq i \leq n-1$$

Esta fórmula implica na continuidade de $s(x)$ em $[a, b]$, assim como as condições 3.30. Para se determinarem as constantes $M_0, \ldots, M_n$, é necessário garantir a continuidade de $s'(x)$ em $x_1, \ldots, x_{n-1}$:

$$\lim_{x \to x_1^+} s'(x) = \lim_{x \to x_1^-} s'(x) \qquad i = 1, \ldots, n-1 \tag{3.34}$$

Em $[x_i, x_{i+1}]$,

$$s'(x) = \frac{-(x_{i+1} - x)^2 M_i + (x - x_i)^2 M_{i+1}}{2h_i} + \frac{y_{i+1} - y_i}{h_i} - \frac{(M_{i+1} - M_i) h_i}{6} \tag{3.35}$$

e em $[x_{i-1}, x_i]$

$$s'(x) = \frac{-(x_i - x)^2 M_{i-1} + (x - x_{i-1})^2 M_i}{2h_{i-1}} + \frac{y_i - y_{i-1}}{h_{i-1}} - \frac{(M_i - M_{i-1}) h_{i-1}}{6}$$

Utilizando-se 3.34 e algumas manipulações,

$$\frac{h_{i-1}}{6} M_{i-1} + \frac{h_i + h_{i-1}}{3} M_i + \frac{h_i}{6} M_{i+1} = \frac{y_{i+1} - yi}{h_i} - \frac{(y_i - y_{i-1}) h_{i-1}}{6} \tag{3.36}$$

para $i = 1, \ldots, n-1$.

Deste modo, obtêm-se $n-1$ equações para $n+1$ incógnitas $M_0, \ldots, M_n$. Existem várias formas de se eliminarem os dois graus de liberdade (ver [Boo78]), entretanto, serão citados os três casos mais comuns. Geralmente são especificadas condições de fim de intervalo (x_0 e x_n):

Caso 1 Condições de derivada para os extremos. A função $s(x)$ deve satisfazer

$$s'(x_0) = y'_0 \qquad s'(x_n) = y'_n \tag{3.37}$$

para y'_0 e y'_n dados. Utilizando-se estas condições e 3.35, para $i = 0$ e $i = n-1$, obtêm-se as equações adicionais:

$$\frac{h_0}{3}M_0 + \frac{h_0}{6}M_1 = \frac{y_1 - y_0}{h_0} - y'_0$$

$$\frac{h_{n-1}}{6}M_{n-1} + \frac{h_{n-1}}{3}M_n = y'_n - \frac{y_n - y_{n-1}}{h_{n-1}}$$

que combinadas com 3.36, resultam no sistema linear:

$$AM = D$$

com

$$D^T = \left[\frac{y_1 - y_0}{h_0} - y'_0, \frac{y_2 - y_1}{h_1} - \frac{y_1 - y_0}{h_0}, \ldots, \frac{y_n - y_{n-1}}{h_{n-1}} - \frac{y_{n-1} - y_{n-2}}{h_{n-2}}, y'_n - \frac{y_n - y_{n-1}}{h_{n-1}}\right]$$

$$M^T = [M_0, M_1, \ldots, M_n]$$

$$A = \begin{bmatrix} \frac{h_0}{3} & \frac{h_0}{6} & 0 & 0 & & \ldots & 0 \\ \frac{h_0}{6} & \frac{h_0+h_1}{3} & \frac{h_1}{6} & & & & \\ 0 & \frac{h_1}{6} & \frac{h_1+h_2}{3} & \frac{h_2}{6} & & & \vdots \\ 0 & & & \ddots & & & \\ \vdots & & & & \frac{h_{n-2}}{6} & \frac{h_{n-2}+h_{n-1}}{3} & \frac{h_{n-1}}{6} \\ 0 & & \ldots & & & \frac{h_{n-1}}{6} & \frac{h_{n-1}}{3} \end{bmatrix} \tag{3.38}$$

Esta matriz é simétrica, definida positiva, e diagonalmente dominante, e o sistema linear $AM + D$ possui solução única. A spline cúbica resultante é denominada de **spline cúbica completa.**

Caso 2 Condições de extremos livres:

$$s''(x_0) = s''(x_n) = 0. \tag{3.39}$$

A spline resultante é denominada **spline natural**. A condição 3.39 produz taxa de erro da ordem de (h^2) nas proximidades dos extremos - a menos que $y''(x_0) = y''(x_n) = 0$. Deste modo a taxa de convergência (ver seção 3.3.3.2 de erro em splines) fica reduzida, diminuindo a eficiência do método. Entretanto, do ponto de vista matemático [ANW67], mostra-se que a

função spline cúbica natural é a única função que possui as propriedades de curvatura mínima entre todas as funções de interpolação e que possuem segunda derivada integrável.

Caso 3 Nos casos onde os valores das derivadas nos extremos não estão disponíveis, são necessárias novas condições, de modo a completar o sistema de equações 3.36. Isto pode ser conseguido impondo-se a continuidade de $s^{(3)}(x)$ em x_1 e x_{n-1}; o que equivale a definir $s(x)$ como sendo uma função spline cúbica com pontos de junção $\{x_0, x_2, x_3, \ldots, x_{n-2}, x_n\}$, e exigir que s interpole todos os pontos $\{x_0, x_1, x_2, \ldots, x_{n-1}, x_n\}$.

Uma outra maneira de se tratar a mesma condição é impor a restrição de que o primeiro trecho de polinômio (p_1) [4] e o último trecho (p_n) interpolem f em pontos adicionais (não junções) x_1 e x_n. Deste modo, obtêm-se $n-3$ trechos de polinômios ao invés de $n-1$. O primeiro trecho p_1 obedeçe às seguintes condições:

$$\begin{aligned} p_1(x_j) &= s(x_j) \qquad j = 0, 2 \\ p_1(x_i) &= f(x_i) \qquad i = 0, 1, 2 \\ p_1'(x_2) &= s'(x_2) \end{aligned}$$

e analogamente para o último trecho,

$$\begin{aligned} p_{n-3}(x_j) &= s(x_j) \qquad j = 0, 2 \\ p_{n-3}(x_i) &= f(x_i) \qquad i = 0, 1, 2 \\ p_{n-3}'(x_{n-2}) &= s'(x_{n-2}) \end{aligned}$$

Isto altera o sistema linear e a notação, mas a função s reultante é idêntica ao cálculo anterior. Analisando-se desta forma, fica bastante claro que os pontos de interpolação e os pontos de junção não devem, necessariamente, coincidir.

3.3.3.2 Análise de Erro

As análises mais recentes de erro de aproximação de uma dada função f por uma spline s, definida num intervalo $[a, b]$, seguem da relação básica de integral 3.28 e segunda relação de integral:

$$\int_a^b (f^{(n)}(x) - s^{(n)}(x))^2 dx = \int_a^b (f(x) - s(x)) f^{(2n)} dx. \tag{3.40}$$

Uma análise detalhada de erro em spline requer um desenvolvimento extremamente extenso [Boo78], [ANW67]. Nesta seção, serão mostrados apenas os resultados do limitante do erro de interpolação de uma função por uma spline cúbica:

[4] O sub-índice 1, diferente da notação de interpolação polinomial onde este significava o grau do polinômio, neste caso, significa o sub-intervalo de interpolação ao qual o polinômio pertence

Teorema 3.3.2 *Seja* $f(x) \in C^4[a,b]$, *uma partição dada por:*

$$\pi\colon\ a = x_0 < x_1 < \cdots < x_n = b$$

e

$$h = \max_{1 \le i \le n} (x_i - x_{i-1})$$

Seja $s_c(x)$ *a spline cúbica completa que interpola* f *na partição* π. *Para o caso 1 tem-se:*

$$s_c(x_i) = f(x_i) \qquad s_c'(a) = f'(a) \qquad s_c'(b) = f'(b)$$

Então para $j = 0, 1, 2$, *obtém-se o limitante*

$$\| f^{(j)}(x) - s_c^{(j)}(x) \| \le c_j \, h^{(4-j)} \, \| f^{(4)}(x) \| \tag{3.41}$$

Valores aceitáveis para as constantes c_j são:

$$c_0 = \frac{5}{384} \qquad c_1 = \frac{1}{24} \qquad c_2 = \frac{3}{8}$$

A prova deste teorema assim como uma análise completa de erro, podem ser encontradas em [Boo78].

Fazendo-se j=0 em 3.41, nota-se que, para uma partição uniforme π, a taxa de convergência é proporcional à taxa para polinômio de Hermite por partes. Entretanto, a maior motivação para o uso das splines completas é a característica de pouca oscilação que estas apresentam quando comparadas com todas as funções suaves que satisfazem as condições de interpolação 3.30.

3.3.3.3 B-splines

As splines cúbicas, descritas na seção 3.3.3.1, apresentam um efeito global, isto é, mudando-se apenas um ponto, todos os pontos da curva serão afetados. Uma maneira de se eliminar este efeito global é expressar a spline através de B-splines. Neste caso, apenas uma parte da curva é afetada pela mudança de um ponto. Como anteriormente, consideram-se os pontos $\{x_0, x_1, \ldots, x_n\}$. Define-se

$$x_+^r = \begin{cases} 0 & x < 0 \\ x^r & x \ge 0 \end{cases}$$

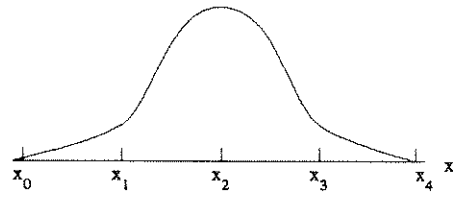


Figura 3.5: B-spline $B_0(x)$.

como uma spline de ordem $r+1$ e com somente um nó $x=0$. Isto pode ser usado como uma maneira alternativa de se representar uma função spline. Seja $s(x)$ uma função spline de ordem m com nós $\{x_0, x_1, \ldots, x_n\}$. Então para $x_0 \leq x \leq x_n$,

$$s(x) = p_{m-1}(x) + \sum_{j=1}^{n-1} \beta_j (x - x_j)_+^{m-1}$$

com p_{m-1} sendo um polinômio de grau $\leq m-1$ unicamente definido e $\beta_1, \beta_2, \ldots, \beta_{n-1}$ coeficientes únicos. O maior problema desta representação é que, geralmente, são gerados sistemas mal-condicionados. Por esta razão, introduz-se uma nova representação de $s(x)$ com propriedades numéricas melhores.

Inicialmente, o número de nós é alterado com a inclusão arbitrária de nós adicionais:

$$x_{-3} < x_{-2} < x_{-1} < x_0 \qquad x_n < x_{n+1} < x_{n+2} < x_{n+3}$$

Para $i = -3, -2, \ldots, n-1$, define-se

$$B_i(x) = (x_{i+4} - x_i) f_x[x_i, x_{i+1}, x_{i+2}, x_{i+3}, x_{i+4}] \tag{3.42}$$

como sendo a diferença dividida de quarta ordem da função

$$f_x(t) = (t-x)_+^3$$

Portanto,

$$f_x(t) = \begin{cases} 0 & x > t \\ (t-x)^3 & x \leq t \end{cases} \tag{3.43}$$

A função $B_i(x)$ é denominada *B-spline*. A equação 3.13, aplicada à fórmula de $B_i(x)$ resulta,

$$B_i(x) = (x_{i+4} - x_i) \sum_{j=1}^{i+4} \frac{(x_j - x)^3}{\Psi_i'(x_j)}$$

onde $\Psi i(x) = (x - x_i)(x - x_{i+1})(x - x_{i+2})(x - x_{i+3})(x - x_{i+4})$. Conclui-se que $B_i(x)$ é

uma spline cúbica com nós $x_i, \ldots, x_{i+4}$. Um gráfico de uma spline típica está ilustrada na figura 3.5. A seguir serão mostradas algumas propriedades de B-splines.

(a) $B_i(x) = 0 \qquad x \ni [x_i, x_{i+4}]$

Da propriedade de diferenças divididas,

$$f[x_0, x_1, \ldots, x_n, x] = \begin{cases} \text{polinômio de grau } m-n-1 & \text{se} \quad n < m-1 \\ a_m & \text{se} \quad n = m-1 \\ 0 & \text{se} \quad n > m-1 \end{cases} \tag{3.44}$$

onde, $f(x) = a_m x^m + a_{m-1} x^{m-1} + \cdots + a_0$.

- Para $t \in [x_i, x_{i+4}]$,

  1. se $x \leq x_i$ então $x \leq t$ e da equação 3.43 conclui-se que f_x possui grau 3. De 3.44 tem-se, para $n = 3 > m - 1$,

     $$f_x[x_i, x_{i+1}, x_{i+2}, x_{i+3}, x_{i+4}] = 0$$

     portanto, $B_i(x) = 0$.

  2. se, por outro lado, $x \geq x_{i+4} \geq t$, então $f_x \equiv 0$ e $B_i(x) = 0$. Portanto, para pontos fora do intervalo $[x_i, x_{i+4}]$, a função $B_i(x)$ é nula.

(b) $\sum_{i=-3}^{n-1} B_i(x) = 1 \qquad x_0 < x < x_n$

Da relação de diferenças divididas,

$$B_i(x) = f_x[x_{i+1}, x_{i+2}, x_{i+3}, x_{i+4}] - f_x[x_i, x_{i+1}, x_{i+2}, x_{i+3}]. \tag{3.45}$$

Assumindo-se x entre dois nós consecutivos ($x_k \leq x \leq x_{k+1}$), tem-se de **(a)** que as únicas B-splines que são diferentes de zero são $B_{k-3}(x), B_{k-2}(x), \cdots, B_k(x)$. Portanto,

$$\sum_{i=-3}^{n-1} B_i(x) = \sum_{i=k-3}^{k} B_i(x)$$

Utilizando-se 3.45,

$$\begin{aligned} \textstyle\sum_{i=-3}^{n-1} B_i(x) &= \textstyle\sum_{i=k-3}^{k} (f_x[x_{i+1}, x_{i+2}, x_{i+3}, x_{i+4}] - f_x[x_i, x_{i+1}, x_{i+2}, x_{i+3}]) \\ &= \textstyle\sum_{i=k-3}^{k} (f_x[x_{k+1}, x_{k+2}, x_{k+3}, x_{k+4}] - f_x[x_{k-3}, x_{k-2}, x_{k-1}, x_k]) \end{aligned}$$

Para $t \in [x_{k+1}, x_{k+4}]$, implica em $x < t$ e $f_x(t)$ é cubica. Novamente, da equação 3.44, para $n = 2 = m - 1$,

$$f_x[x_i, x_{i+1}, x_{i+2}, x_{i+3}, x_{i+4}] = a_m = 1$$

e para $t \in [x_{k-3}, x_k]$, $x > t$ e portanto $f_x(t) \equiv 0$. Entao,

$$\sum_{i=-3}^{n-1} B_i(x) = 1 - 0 = 1.$$

(c) $0 \leq B_i(x) \leq 1$ para todo x;

(d) $\int_{x_i}^{x_i+4} B_i(x)dx = \frac{(x_{i+4} - x_i)}{4}$;

(e) $s(x) = \sum_{i=-3}^{n-1} \alpha_i B_i(x)$;

onde $s(x)$ é uma spline cúbica com nós $\{x_0, \ldots, x_n\}$, $x_0 \leq x \leq x_n$ e com escolha única de $\alpha_{-3}, \ldots, \alpha_{n-1}$.

A prova das propriedades (c), (d) e (e) pode ser encontrada em [Boo78].

Capítulo 4

Interpolação Nebulosa

4.1 Introdução

A descrição de processos com representação analítica desconhecida requer uma base de dados cuja dimensão seja suficientemente grande a fim de determinar, de forma precisa, relações de entrada-saída. Todavia, na maioria dos casos, esta condição pode levar a um esforço computacional tal, que o problema se torne intratável. Nestes casos, a compressão da base de dados através de regras nebulosas se torna uma saída viável para a solução do problema.

O objetivo desta seção é descrever um método de interpolação baseado em lógica nebulosa, para representar um sistema incerto de entrada-saída. Uma curva, por exemplo, cuja representação analítica não se conhece, pode ser representada por um conjunto mínimo de pontos. Os pontos que não foram considerados podem ser obtidos, tomando-se como base o conjunto de pontos dados e aplicando-se um algoritmo de interpolação estruturado por meio de regras nebulosas.

O método de interpolação nebulosa, desenvolvido por Uchino *et al.* [UYY90], utiliza regras nebulosas lineares para interpolar funções, passando por determinados pares de entrada-saída. O algoritmo original (INL)- Interpolação Nebulosa Linear - descrito na seção 4.2, funciona somente quando os pontos obedecem a uma condição de seqüencialidade, ou seja, as curvas são funções.

Este capítulo apresenta ainda algumas modificações no algoritmo original, de modo a torná-lo mais flexível e com larga aplicação em compressão de dados. Desta forma, o novo método (INNL) - Interpolação Nebulosa Não Linear, por nós desenvolvido [ZRR92] e

descrito na seção 4.3, determina curvas interpoladas mais suaves com a introdução de regras não lineares. Uma outra modificação, aqui proposta, é a possibilidade de se interpolarem curvas quaisquer (funções ou não-funções).

A seção 4.3.1 apresenta a introdução da não linearidade nas funções de pertinência dos conjuntos nebulosos que são a base para a interpolação INNL. A justificativa teórica que garante uma maior suavidade na curva interpolada é mostrada na seção 4.3.2. Na seção 4.4 uma análise é feita, no sentido de se provar a convergência do método INNL. Finalmente, a seção 4.5 traz a proposta que permite a interpolação de curvas fechadas.

4.2 Princípios da Interpolação Nebulosa - Regras Lineares

O princípio básico da interpolação nebulosa deriva do fato de que, entre dois pontos existe uma reta nebulosa (curva), a qual pode ser interpolada com base na influência destes dois pontos e mais dois (anterior e posterior), através das retas que os unem. Portanto, para realizar a interpolação em cada trecho, o algoritmo necessita de uma sequência de quatro pontos que serão denominados pontos de suporte. A participação de cada reta (reta suporte) é ponderada por funções de pertinência de conjuntos nebulosos, definidos em cada trecho de interpolação.

Sejam os pares de entrada-saída, ou pontos originais da curva, dados por:

$$(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n).$$

Portanto, para um intervalo $[x_{i+1}, x_{i+2}]$, os pontos de suporte são,

$$(x_i, y_i), (x_{i+1}, y_{i+1}), (x_{i+2}, y_{i+2}), (x_{i+3}, y_{i+3})$$

onde, $x_i < x_{i+1} < x_{i+2} < x_{i+3}$.

Sejam $Y_{i+1}^{(j)}(x)$, $(j = 1, 2, 3)$, as retas que unem os pontos sucessivos

$$(x_{i+j-1}, y_{i+j-1}) \text{ e } (x_{i+j}, y_{i+j}).$$

As regras para a interpolação são dadas por:

$$\begin{array}{lllllll} R_{i+1}^{(1)}: & \text{Se } x & \text{é } A_{i+1}^{(1)} & \text{então} & y_{i+1}(x) & = & Y_{i+1}^{(1)}(x) \\ R_{i+1}^{(2)}: & \text{Se } x & \text{é } A_{i+1}^{(2)} & \text{então} & y_{i+1}(x) & = & Y_{i+1}^{(2)}(x) \\ R_{i+1}^{(3)}: & \text{Se } x & \text{é } A_{i+1}^{(3)} & \text{então} & y_{i+1}(x) & = & Y_{i+1}^{(3)}(x) \end{array}$$

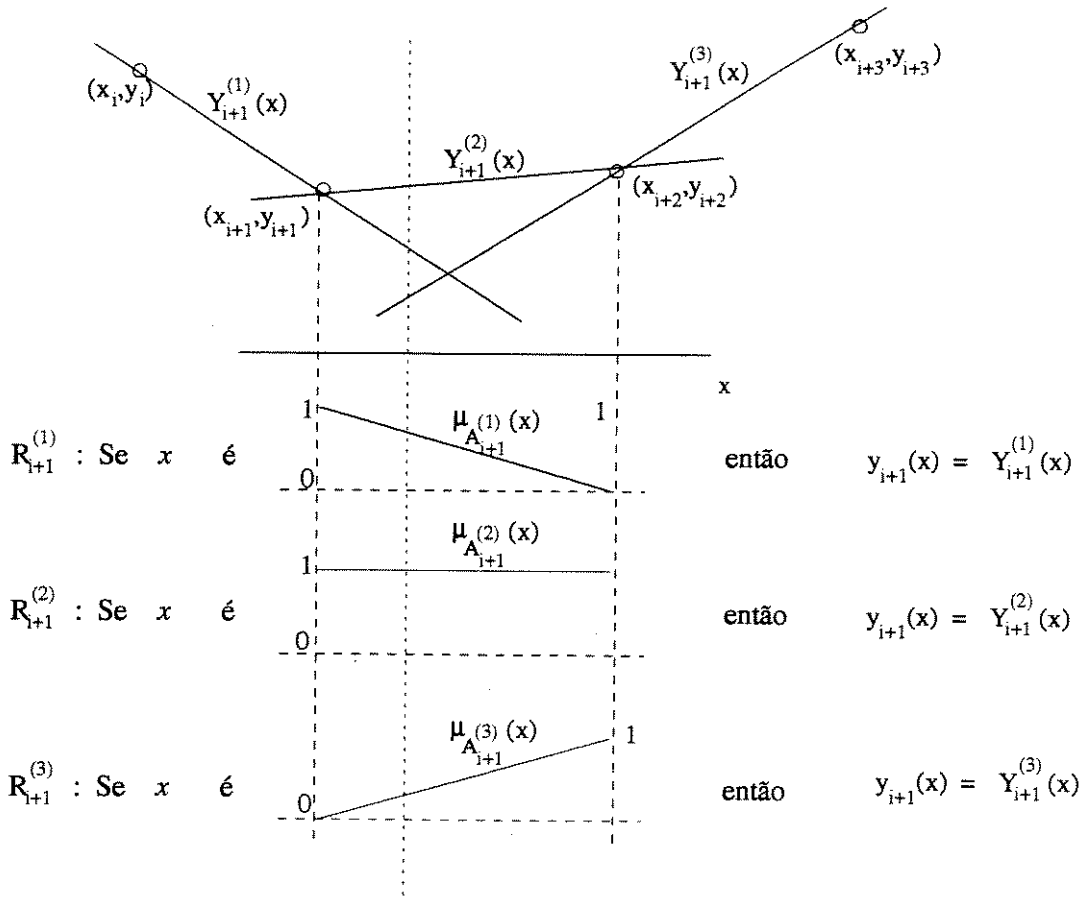


Figura 4.1: Princípios básicos da interpolação nebulosa

onde, $A_{i+1}^{(j)}$ é um conjunto nebuloso dado, com função de pertinência $\mu_{A_{i+1}^{(j)}}(x)$. A saída não nebulosa obtida pela aplicação do raciocínio aproximado do tipo 2 (seção 2.2.4), para uma entrada $x \in [x_{i+1}, x_{i+2}]$ é dada por:

$$y_{i+1}(x) = \frac{\sum_{j=1}^{3} \mu_{A_{i+1}^{(j)}}(x)\ Y_{i+1}^{(j)}(x)}{\sum_{j=1}^{3} \mu_{A_{i+1}^{(j)}}(x)}$$

A figura 4.1 ilustra o princípio básico da interpolação. A forma das funções de pertinência dos conjuntos $A_{i+1}^{(j)}$ casa com a idéia intuitiva de que, no início do intervalo de interpolação, a influência da reta $Y_{i+1}^{(1)}(x)$ é maior e decresce à medida que se caminha para o fim do intervalo; a reta $Y_{i+1}^{(2)}(x)$ influencia sempre (função constante = 1); e no final do intervalo, a reta $Y_{i+1}^{(3)}(x)$ passa a ter maior contribuição. No caso onde os pontos (x_i, y_i) e

(x_{i+3}, y_{i+3}) não são dados, as alturas [Kau75] dos conjuntos $A_{i+1}^{(1)}$ e $A_{i+1}^{(2)}$ são iguais a zero. Isto determina que, para somente dois pontos dados (x_{i+1}, y_{i+1}), e (x_{i+2}, y_{i+2}), a melhor interpolação entre eles é uma reta; o que também é intuitivamente correto.

Este método, embora prime pela simplicidade, não garante a suavidade da curva interpolada. A figura 4.2 ilustra a curva original $f(x) = x^2$ e a curva interpolada $g(x)$ obtida pelo método INL, descrito anteriormente. A não suavidade nos pontos de junção fica bastante evidente, pela descontinuidade da derivada da curva interpolada (fig. 4.3). Uchino *et al.* [UYY90], propõe nestes casos, a criação de regras suplementares de modo a reduzir o salto da derivada, tornando a curva interpolada mais suave.

As regras suplementares são bastante simples e estabelecem um novo conjunto de pontos de suporte,

$$p_1 : ((x_{i-1} + x_i)/2, y_i')$$
$$p_2 : (x_i, y_i)$$
$$p_3 : ((x_i + x_{i+1})/2, y_{i+1}')$$
$$p_4 : (x_{i+1}, y_{i+1})$$

onde, $p_2 : (x_i, y_i)$ é o ponto de descontinuidade. Um outro conjunto de pares entrada-saída é gerado, onde p_1 e p_3 são pontos suplementares extraídos da base de dados original. Aplica-se outra vez o algoritmo para o novo conjunto de dados, onde se observam os novos pontos de descontinuidade. O processo é repetidamente empregado até que o salto da derivada não ultrapasse um valor ξ, considerado aceitável para a aplicação em questão.

Um fato a ser salientado, é a não praticidade do método para os casos onde o valor de ξ é muito baixo. A recorrência sucessiva à base de dados original para a determinação dos pontos suplementares, pode gerar um conjunto de pares entrada-saída de mesma ordem de grandeza da base original. Deste modo, inviabiliza-se a aplicação do método de interpolação como forma de compressão de dados.

4.3 Interpolação Nebulosa Utilizando Regras Não Lineares

O método de interpolação nebulosa com regras não lineares (INNL), desenvolvido neste trabalho, propõe uma solução para suavização da curva interpolada, sem a necessidade de informações adicionais da base de dados original. As informações necessárias são extraídas do próprio conjunto de pontos entrada-saída. Desta maneira, dependendo do grau de sua-

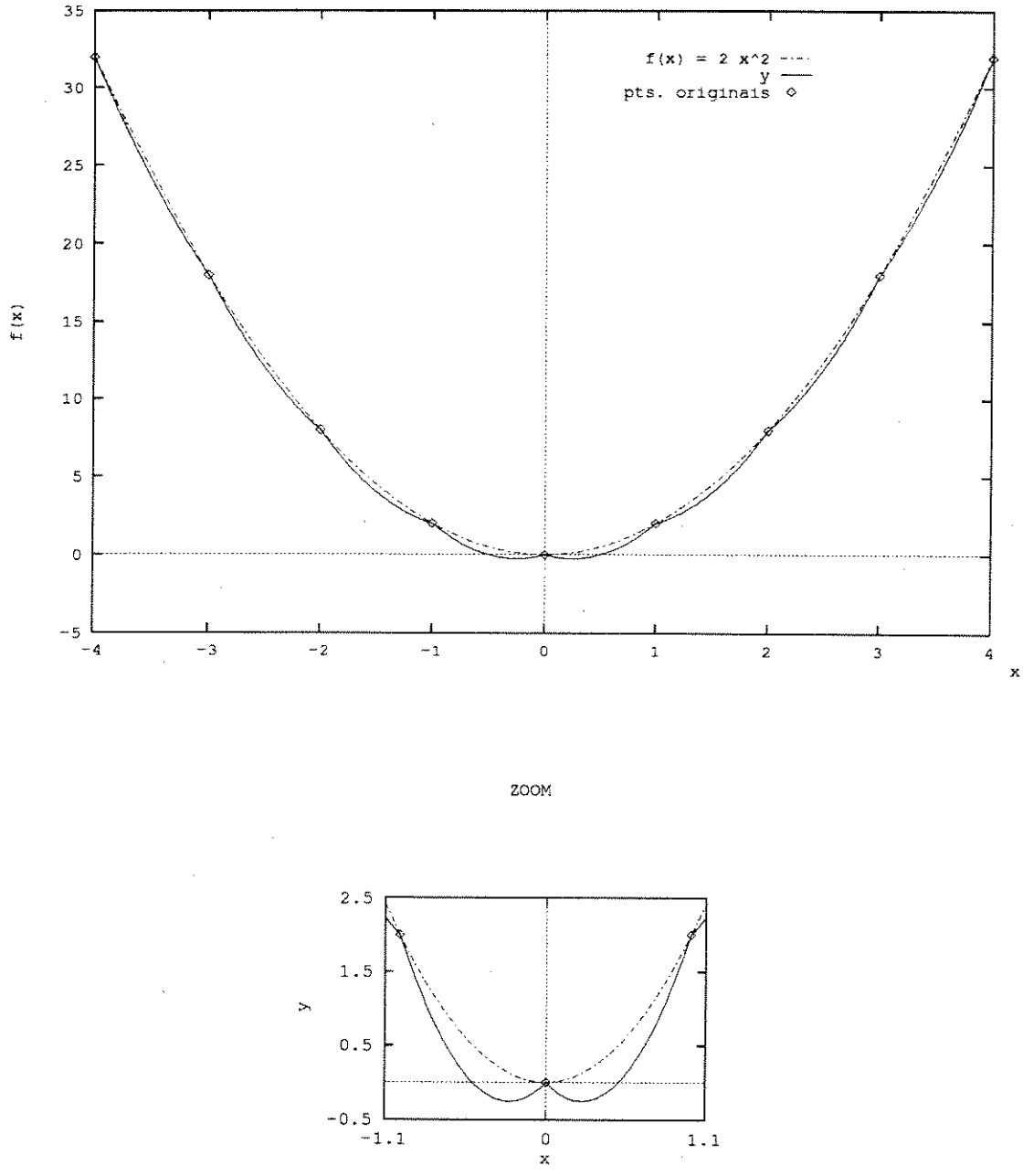


Figura 4.2: Curva original $f(x) = 2x^2$ X curva interpolada y

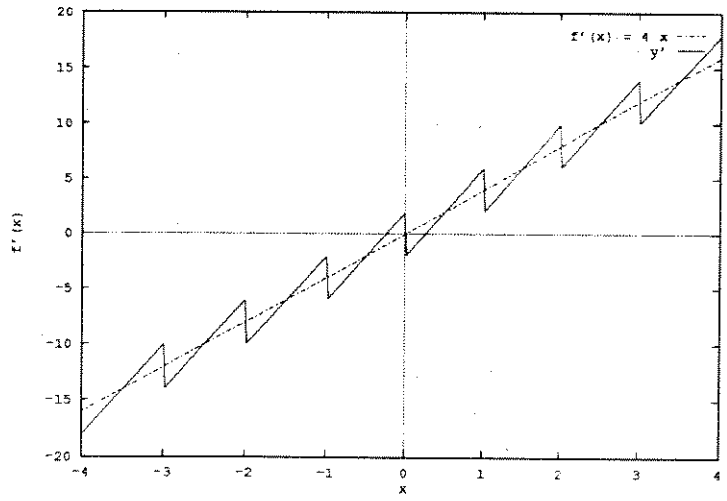


Figura 4.3: Gráfico da derivada da curva original $f'(x) = 4x$ e derivada da curva interpolada $y'(x)$

vidade e precisão da curva interpolada desejada, torna-se desnecessário o cálculo de pontos suplementares.

A principal modificação em relação ao método original, consiste da introdução da não linearidade nas funções de pertinência $\mu_{A_{i+1}^{(j)}}(x)$, $(j = 1, 2, 3)$. Esta não linearidade é obtida elevando-se a função $\mu_{A_{i+1}^{(j)}}$ a uma potência $k^{(j)}$, (j=1,2,3), que pode ser constante > 1 ou inferida de regras nebulosas (seção 4.3.1). Então, a função $g(x)$ que define uma saída não nebulosa $y_{i+1}(x)$ passa a ser determinada por:

$$g(x) = \frac{\sum_{j=1}^{3} [\mu_{A_{i+1}^{(j)}}(x)]^{k^{(j)}} \; Y_{i+1}^{(j)}(x)}{\sum_{j=1}^{3} [\mu_{A_{i+1}^{(j)}}(x)]^{k^{(j)}}}. \tag{4.1}$$

Verifica-se que a simples introdução de uma potência > 1 suaviza a curva interpolada resultante, como mostra a figura 4.4 e garante a continuidade derivada $g'(x)$ (figura 4.5). A justificativa teórica para a suavização está demonstrada na seção 4.3.2.

4.3.1 Cálculo da Potência $k^{(j)}$

Esta seção ilustra a nova base de regras para se calcular a potência $k^{(j)}, (j = 1, 2, 3)$ com base nos ângulos entre as retas $Y_{i+1}^{(1)}(x)/Y_{i+1}^{(2)}(x)$ e $Y_{i+1}^{(2)}(x)/Y_{i+1}^{(3)}(x)$.

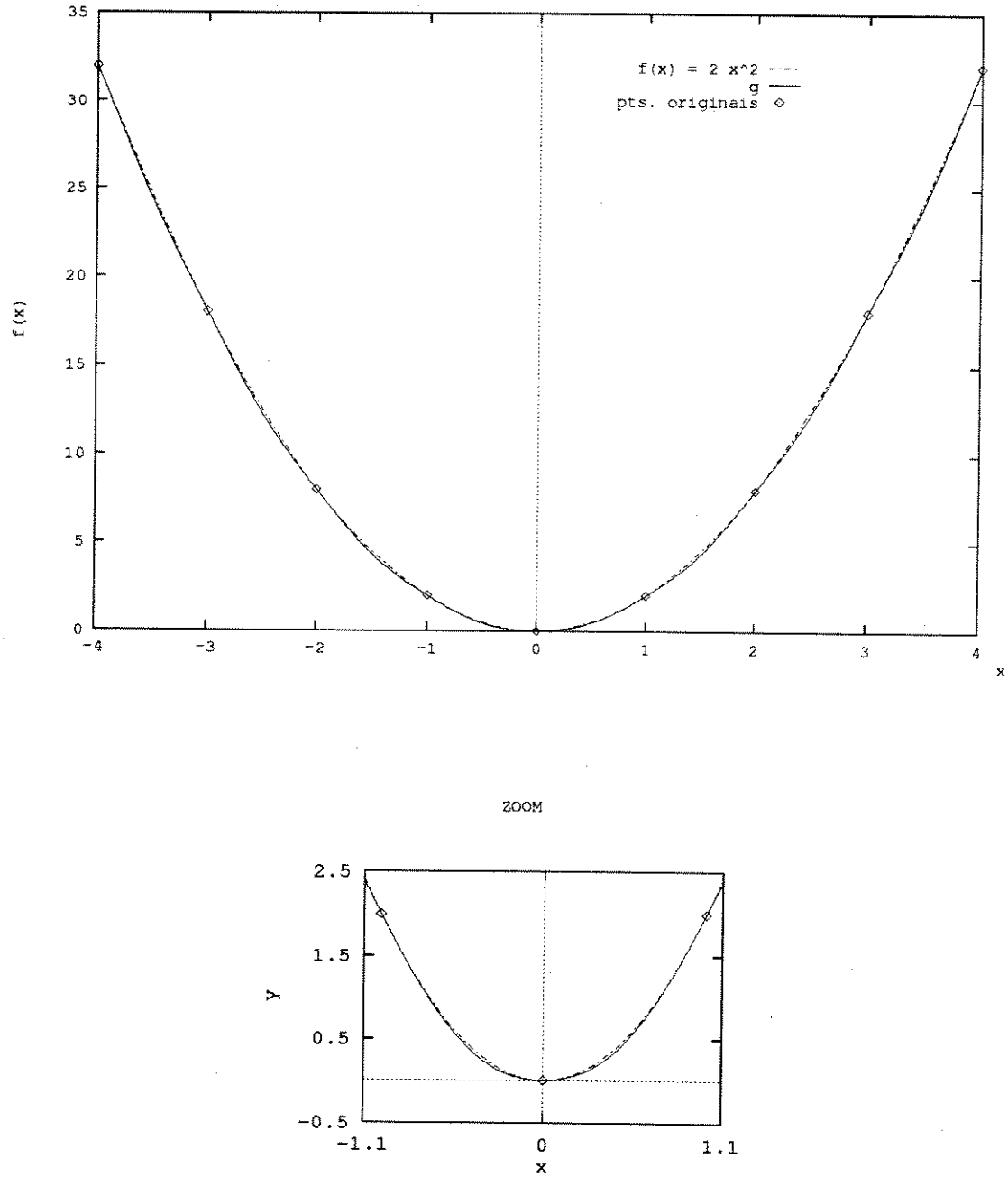


Figura 4.4: Curva original $f(x) = 2x^2$ X curva interpolada g

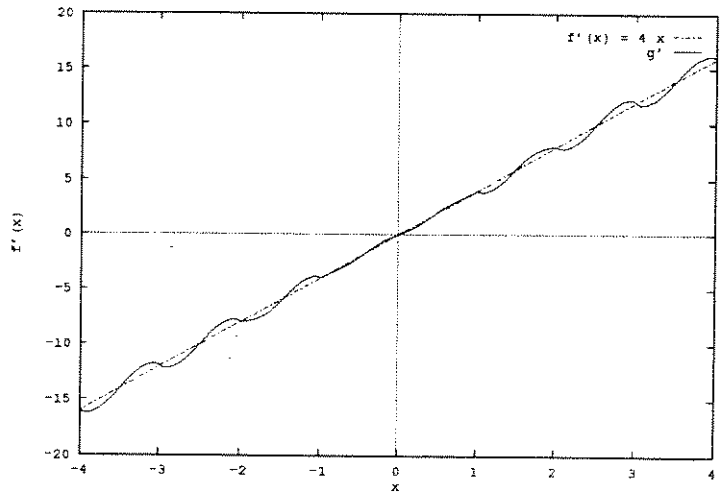


Figura 4.5: Gráfico da derivada da curva original $f^{'}(x) = 4x$ e derivada da curva interpolada $g^{'}(x)$

Seja $\alpha^{(m)}, m \in \{1,3\}$, os ângulos entre as retas $Y_{i+1}^{(j)}(x)$ $(j = 1,2,3)$, como ilustra a figura 4.6. Seja $k^{(m)}, (m = 1,3)$ a potência a ser determinada. Então, o conjunto de regras suplementares com antecedente $\alpha^{(m)}$ e conseqüente $k^{(m)}$ é dado por:

$R_{i+1}^{(1)}$:	Se	$\alpha^{(m)}$	é	pequeno	então	$k^{(m)}$	é	baixo
$R_{i+1}^{(2)}$:	Se	$\alpha^{(m)}$	é	médio	então	$k^{(m)}$	é	médio
$R_{i+1}^{(3)}$:	Se	$\alpha^{(m)}$	é	grande	então	$k^{(m)}$	é	alto

onde, $m \in 1,3$; o valor de $k^{(2)}$ é dado por uma constante qualquer, pois a função de pertinência $\mu_{A_{i+1}^{(2)}}(x) =$ constante $= 1$ não é alterada pela potência $k^{(j)}$.

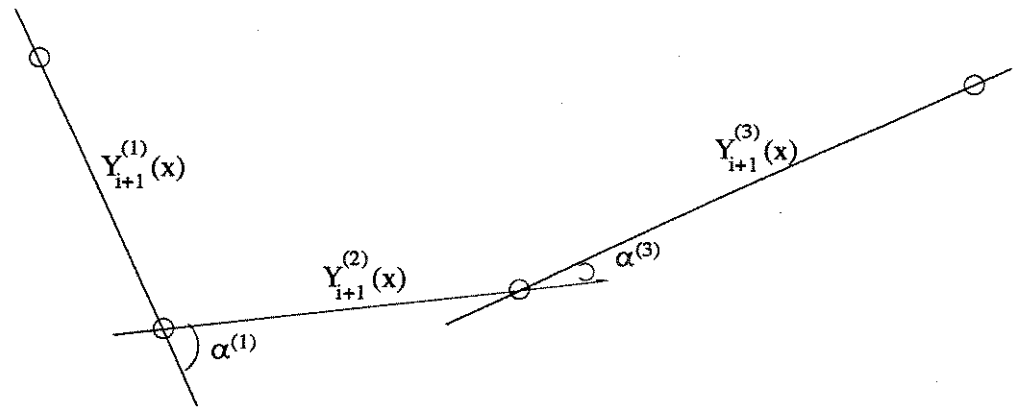


Figura 4.6: Ângulo entre as retas suporte

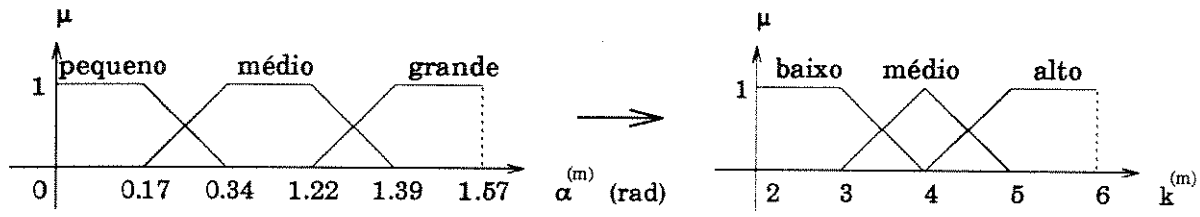


Figura 4.7: Regras de inferência nebulosa para o cálculo da potência $k^{(j)}$.

Estas regras estão mostradas na figura 4.7. Verifica-se que o universo de discurso da saída inferida $k^{(m)}$ se limita ao intervalo $[2,6]$. Estes valores forma escolhidos de modo que o valor mínimo garanta a continuidade da derivada g' da função de interpolação, como será visto na seção a seguir e o limite superior tenta evitar os efeitos oscilatórios.

A concepção das regras se baseia na idéia de que se os ângulos são menores (retas quase alinhadas), polinômios de graus menores são mais adequados, ao passo que ângulos maiores (mudança brusca na direção das retas) exigem polinômios com graus mais elevados. Esta idéia pode ser melhor ilustrada pelo seguinte raciocínio:

- se as 3 retas de suporte $Y_{i+1}^{(j)}(x)$, $(j = 1,2,3)$ estão quase alinhadas ($\alpha^{(m)}$ é pequeno), então a influência destas retas deve decrescer lentamente. Isto pode ser obtido com uma potência $k^{(m)}$ baixa, ou seja, o valor da função de pertinência

$$[\mu_{A_{i+1}^{(m)}}(x)]^{k^{(m)}} \quad (m \in \{1,3\})$$

  está mais próximo do valor linear $\mu_{A_{i+1}^{(m)}}(x)$.

- se existe uma mudança brusca de direção de uma reta para a outra, por exemplo, da primeira para a segunda reta ($\alpha^{(1)}$ grande), deve-se ter a influência da primeira reta $\mu_{A_{i+1}^{(1)}}$ caindo rapidamente para dar lugar à influência da terceira reta $\mu_{A_{i+1}^{(3)}}$. Logo, o valor de $k^{(1)}$ deve ser alto o suficiente para atender à condição anterior e não provocar efeitos oscilatórios. A figura 4.8 traz os valores das funções lineares e não lineares para $k^{(1)} = 5$ e $k^{(3)} = 2$. O cálculo das saídas não nebulosas é feito empregando-se o método do centro de área descrito na seção 2.2.6.

4.3.2 Justificativa Teórica

O objetivo desta seção é apresentar uma justificativa matemática para a introdução da potência $k^{(j)}$, de forma a se eliminar a descontinuidade da derivada da função interpolada (fig. 4.3), garantindo-se assim, uma curva resultante mais suave.

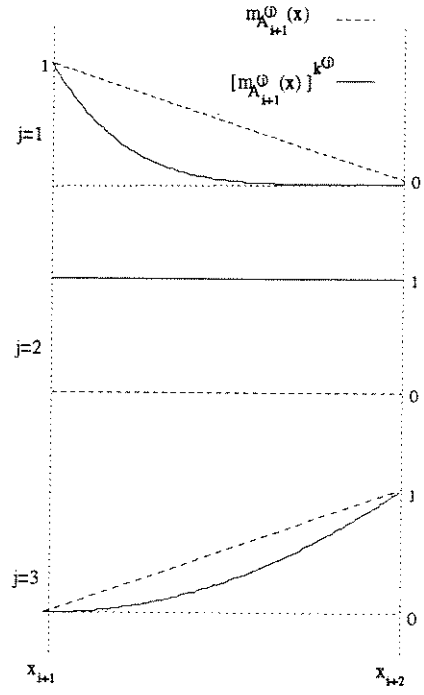


Figura 4.8: Comparação entre as funções de pertinência para diferentes potências $k^{(j)}$, onde $k^{(1)} = 5$ e $k^{(3)} = 2$.

Teorema 4.3.1 *Seja o conjunto de pares entrada-saída formado pelos pontos,*

$$(x_0, y_0), (x_1, y_1), (x_2, y_2), (x_3, y_3), (x_4, y_4) \tag{4.2}$$

onde $x_0 < x_1 < \cdots < x_4$. *Seja* $g(x)$ *a função dada pela equação 4.1 e que interpola* $f(x)$ *nos pontos dados (4.2). A derivada* $g'(x)$ *é contínua para* $k^{(j)} > 1 (j = 1, 2, 3)$.

Prova:

Para a análise de continuidade, será considerada a transição dos trechos $[x_1, x_2]$ para $[x_2, x_3]$, como ilustra a figura 4.9. Sejam $g_1(x)$ e $g_2(x)$ funções que interpolam $f(x)$ nos trechos $[x_1, x_2]$ e $[x_2, x_3]$ respectivamente, e dadas por:

$$g_1 = \frac{\sum_{j=1}^{3} [\mu_j(x)]^{k^{(j)}} \; Y_j(x)}{\sum_{j=1}^{3} [\mu_j(x)]^{k^{(j)}}} \tag{4.3}$$

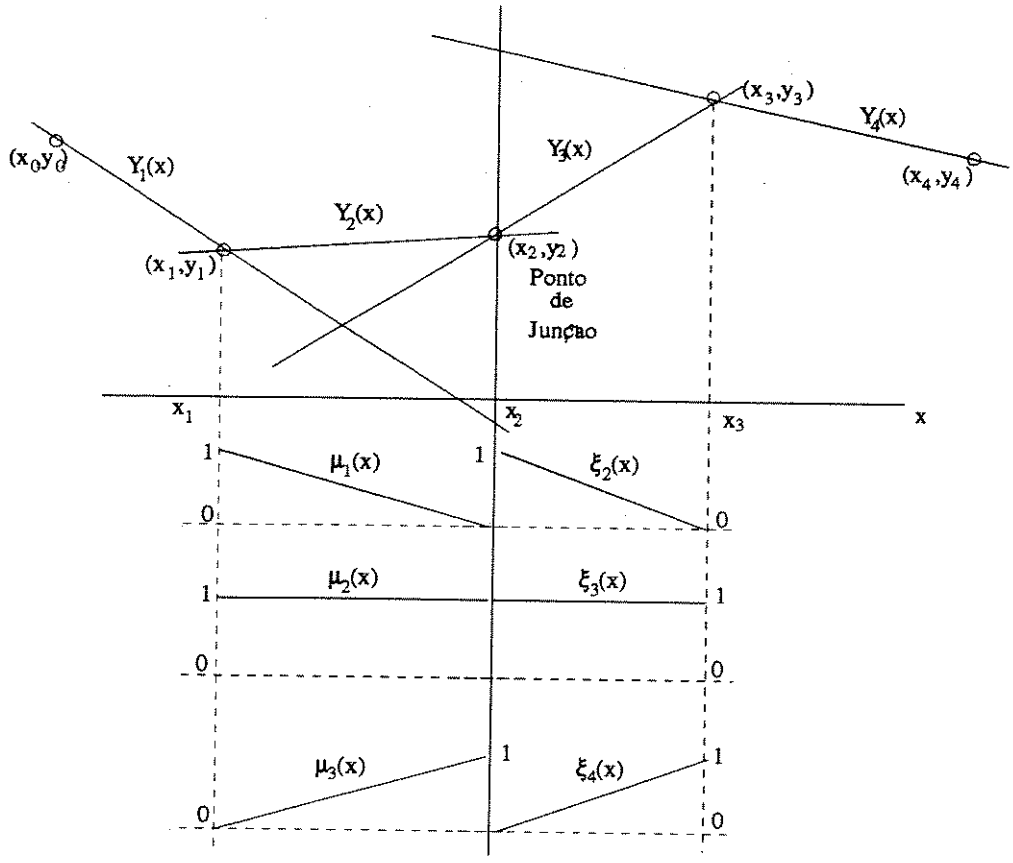


Figura 4.9: Interpolação em dois trechos subseqüentes: $[x_0, x_3]$ e $[x_1, x_4]$

$$g_2 = \frac{\sum_{j=2}^{4} [\xi_j(x)]^{k(j)} \, Y_j(x)}{\sum_{j=2}^{4} [\xi_j(x)]^{k(j)}}$$

onde $\mu_j(x)$ e $\xi_j(x)$ são funções de pertinência lineares[1]; $Y_j(x)$ é a reta que une os pontos (x_{j-1}, y_{j-1}) e (x_j, y_j); e $k^{(j)}$ é uma potência ≥ 1.

Primeiramente será analisada a continuidade da função g. Na transição dos intervalos de interpolação $[x_0, x_3]$ para $[x_1, x_4]$ (figura 4.9), verifica-se que os valores das retas $Y_2(x)$ e $Y_3(x)$ são iguais a y_2 para o ponto de junção (x_2, y_2). Os pontos de descontinuidade $Y_1(x_2)$ e $Y_4(x_2)$ são anulados pelas ponderações $\mu_1(x_2) = \xi_4(x_2) = 0$. Portanto, fica claro que os fatores $[\mu_j(x)]^{k(j)}$ e $[\xi_j(x)]^{k(j)}$ são fundamentais para se garantir a continuidade de g,

[1] A notação utilizada foi alterada a título de simplificação. As variáveis $\mu_j(x)$ e $\xi_j(x)$ indicam os valores de $\mu_{A^j_{i+1}}(x)$ e $\xi_{A^j_{i+1}}(x)$ no primeiro e segundo itervalo, respectivamente.

comprovada pela aplicação do limite,

$$\lim_{x \to x_2} g_1(x) = \lim_{x \to x_2} g_2(x) = y_2$$

Lema 4.3.1 *Os termos* $[\mu_j(x)]^{k(j)-1}$ *e* $[\xi_j(x)]^{k(j)-1}$ *garantem a continuidade da derivada* $g'(x)$ *na transição do trecho* $[x_1, x_2]$ *para* $[x_2, x_3]$.

Prova: Da equação 4.3 tem-se

$$g_1'(x) = -g_1 \frac{\sum_{j=1}^{3} k^{(j)}[\mu_j(x)]^{k(j)-1}\ \mu_j'(x)}{\sum_{j=1}^{3}[\mu_j(x)]^{k(j)}} + \frac{\sum_{j=1}^{3} k^{(j)}[\mu_j(x)]^{k(j)}\ Y_j'(x)}{\sum_{j=1}^{3}[\mu_j(x)]^{k(j)}} + \frac{\sum_{j=1}^{3} k^{(j)}[\mu_j(x)]^{k(j)-1}\ \mu_j'(x)\ Y_j(x)}{\sum_{j=1}^{3}[\mu_j(x)]^{k(j)}} \tag{4.4}$$

Rearranjando a equação 4.4, obtém-se:

$$g_1'(x) = \frac{\sum_{j=1}^{3} k^{(j)}[\mu_j(x)]^{k(j)-1}\ \mu_j'(x)\ (Y_j(x) - g_1)}{\sum_{j=1}^{3}[\mu_j(x)]^{k(j)}} + \frac{\sum_{j=1}^{3} k^{(j)}[\mu_j(x)]^{k(j)}\ Y_j'(x)}{\sum_{j=1}^{3}[\mu_j(x)]^{k(j)}} \tag{4.5}$$

e similarmente para a função g_2,

$$g_2'(x) = \frac{\sum_{j=2}^{4} k^{(j)}[\xi_j(x)]^{k(j)-1}\ \xi_j'(x)\ (Y_j(x) - g_2)}{\sum_{j=2}^{4}[\xi_j(x)]^{k(j)}} + \frac{\sum_{j=2}^{4} k^{(j)}[\xi_j(x)]^{k(j)}\ Y_j'(x)}{\sum_{j=2}^{4}[\xi_j(x)]^{k(j)}}. \tag{4.6}$$

Aplicando-se a mesma análise feita para g_1 e g_2, tem-se para o ponto de junção $x = x_2$,

$$Y_2(x_2) = g_1(x_2) = y_2$$
$$Y_3(x_2) = g_2(x_2) = y_2$$

e os termos de descontinuidade $\mu_1'(x_2)(Y_1(x_2) - g_1(x_2))$ e $\xi_4'(x_2)(Y_4(x_2) - g_2(x_2))$ são eliminados pelos fatores $[\mu_1(x_2)]^{k(1)-1}$ e $[\xi_4(x_2)]^{k(4)-1}$, respectivamente. Os termos mais à direita das equações 4.5 e 4.6 não apresentam descontinuidade, como ilustra a aplicação do limite,

$$\lim_{x \to x_2} g_1'(x) = \lim_{x \to x_2} g_2'(x) = \frac{Y_2'(x_2) + Y_3'(x_2)}{2}$$

Portanto, a função g' é contínua, visto que pela restrição inicial tem-se que $x_0 < x_1 < \cdots < x_4$ e conseqüentemente, $\frac{Y_2'(x_2)+Y_3'(x_2)}{2}$ será um valor finito.

Se, por outro lado $k^{(j)}$ for igual a 1, os termos $[\mu_j(x)]^{k^{(j)}-1}$ e $[\xi_j(x)]^{k^{(j)}-1}$ das equações 4.5 e 4.6 desaparecem e a aplicação do limite demonstra a descontinuidade da derivada g',

$$\begin{aligned} \lim_{x\to x_2} g_1'(x) &= \tfrac{1}{2}[\mu_1'(x_2)(Y_1(x_2)-y_2)] + \tfrac{Y_2'(x_2)+Y_3'(x_2)}{2} \\ \lim_{x\to x_2} g_2'(x) &= \tfrac{1}{2}[\xi_4'(x_2)(Y_4(x)-y_2)] + \tfrac{Y_2'(x_2)+Y_3'(x_2)}{2} \end{aligned}$$

Conseqüentemente, para se garatir a continuidade de g' e portanto, uma maior suavidade da função interpolada g, as regras devem ser não lineares, ou seja, $k^{(j)} > 1$, $(j = 1, 2, 3)$.

4.4 Análise de Convergência

Esta análise tem como objetivo verificar a convergência do método de interpolação INNL, descrito nas seção anterior, com base na análise de convergência de *splines* [ANW67].

A seguir serão definidos os conceitos de *módulo de continuidade de uma função* $f(x)$, *spline de deficiência 2* e dois lemas, necessários para a análise de convergência ao se interpolar a função $f(x)$, por $g(x)$ dada pela equação 4.1.

Módulo de Continuidade

Seja S um conjunto de pontos no espaço n-dimensional. A função $\mu(f;\delta)$, definida para $0 < \delta < \infty$ é chamada de módulo de continuidade da função $f(x)$ com domínio em S, se e somente se, para

$$x \text{ e } x' \in S$$

$$\| x - x' \| \leq \delta$$

tem-se

$$| f(x) - f(x') | \leq \mu(f;\delta),$$

onde $\| x \|$ define a norma de x como sendo o maior dos valores de $| x_i |$ $(i = 1, \cdots, n)$.

Spline de Deficiência 2

Define-se spline de deficiência 2 como sendo uma spline cúbica por partes $s(x)$ com primeira derivada $s'(x)$ contínua e para a qual, nada se sabe a respeito da segunda derivada $s''(x)$.

Lema 4.4.1 *Seja* $\{\Delta_k\}$ *uma seqüência de partições em* $[a,b]$, *com* $\| \Delta_k \| \rightarrow 0$ *para* $k \rightarrow \infty$. *Seja* $f(x) \in C[a,b]$. *Seja ainda* $\hat{S}_k(x)$ *uma spline de deficiência 2 interpolando* $f(x)$ *nos pontos da partição* Δ_k *e que possua derivada nula nestes pontos. Então, a seqüência* $\{\hat{S}_k(x)\}$ *converge uniformemente para* $f(x)$ *em* $[a,b]$, *pois*

$$| f(x) - \hat{S}_k(x) | \leq 2\mu(f; \| \Delta \| /2).$$

Lema 4.4.2 *Seja* $\{\Delta_k\}$ *uma seqüência de partições em* $[a,b]$, *com* $\| \Delta_k \| \rightarrow 0$ *quando* $k \rightarrow \infty$. *Sejam* $\hat{S}_k(x)$ *e* $\hat{T}_k(x)$ *duas splines em* Δ_k, *coincidentes nos pontos de partição. Se* $\max_j | \hat{S}'_k(x_{k,j}) | \| \Delta_k \| \rightarrow 0$ *e* $\max_j | \hat{T}'_k(x_{k,j}) | \| \Delta_k \| \rightarrow 0$, *então, em* $[a,b]$, $\hat{S}_k(x)$ *converge uniformemente para* $\hat{T}_k(x)$.

As provas destes lemas podem ser encontradas em [ANW67].

Considerando-se $g(x)$ como uma spline de deficiência 2 tem-se, pelo lema 4.4.2,

$$[g(x) - \hat{S}_k(x)] \rightarrow 0$$

e pelo lema 4.4.1

$$\hat{S}_k(x) \rightarrow f(x),$$

ou seja, $g(x)$ converge uniformemente para $f(x)$ quando $\Delta \rightarrow 0$.

4.5 Interpolação de Trajetórias Fechadas

A idéia de se aplicar o algoritmo INNL a trajetórias fechadas (não funções), motivou o desenvolvimento de outras modificações no algoritmo original INL. A condição necessária de que $x_i < x_{i+1} < x_{i+2} < x_{i+3}$ restringe , por exemplo, a aplicação para o caso onde se deseja interpolar uma trajetória fechada, dados quatro pontos diametralmente opostos.

Uma das formas de se interpolar a trajetória acima descrita seria dividí-la em partes que definissem funções distintas, rotacionando os eixos referenciais. Entretanto, seriam

necessários, no mínimo, dois pontos suplementares (um na parte inferior e um na parte superior), para se obter a interpolação desejada e satisfazer à condição:

$$x_i < x_{i+1} < x_{i+2} < x_{i+3}.$$

Desta maneira, a utilização do algoritmo INNL para recuperação de curvas, dado um conjunto mínimo de pontos originais, estaria comprometida. A solução alternativa pode ser obtida através de uma nova alteração na função de interpolação $g(x)$ que permite uma relaxação das condições iniciais de distribuição dos pontos dados.

Utilizando-se regras não lineares, desenvolvidas na seção 4.3, e considerando-se o conjunto de pares entrada-saída,

$$\{(x_0, y_0), \ldots, (x_n, y_n)\}$$

tal que $x_i \neq x_{i+1} \neq x_{i+2} \neq x_{i+3}$, tem-se para uma trajetória fechada, uma saída modificada dada por:

$$y_{i+1}(x) = Y^{(2)}_{i+2}(x) + y_{rel_{i+1}}(x)$$

onde $y_{rel_{i+1}}(x)$ indica a representação de $y_{i+1}(x)$ em relação ao eixo **x** transladado para $Y^{(2)}_{i+1}(x)$. Desta forma, o valor modificado da saída não nebulosa passa a ser dado por:

$$y_{i+1}(x) = Y^{(2)}_{i+2}(x) + \left[\frac{\sum_{i=1}^{3} [\mu_{A^{(j)}_{i+1}}(x)]^{k^{(j)}} \quad (-1)^{n^{(j)}} \quad Y^{(j)}_{rel_{i+1}}(x)}{\sum_{i=1}^{3} [\mu_{A^{(j)}_{i+1}}(x)]^{k^{(j)}}} \right] \tag{4.7}$$

onde $Y^{(j)}_{rel_{i+1}}(x) = Y^{(j)}_{i+1}(x) - Y^{(2)}_{i+1}(x)$ e

$$n^{(j)} \text{ é } \begin{cases} \text{par} & \text{se } x_{i+j-1} < x_{i+j} \\ \text{ímpar} & \text{se } x_{i+j-1} > x_{i+j} \\ \text{qualquer} & \text{se } j = 2 \end{cases} \tag{4.8}$$

As figuras 4.10(a) e 4.10(b) ilustram os casos dos pontos seqüenciais em x ($x_0 < x_1 < x_2 < x_3$) e não seqüencias em x ($x_0 < x_1 < x_2 > x_3$), respectivamente. Aplicando-se a equação 4.8 à disposição de pontos da figura 4.10(a) obtém-se,

$$n^{(1)} \text{ par}$$
$$n^{(3)} \text{ par}$$

e para a figura 4.10(b)

$$n^{(1)} \text{ par}$$
$$n^{(3)} \text{ ímpar}$$

A forma da equação 4.7 segue da idéia de possibilidade de variação de $y_{rel_{i+1}}(x)$. Verifica-se, por exemplo, que para a figura 4.10(a), este valor varia entre os pontos $Y^{(2)}_{rel_{i+1}}(x)$ e $Y^{(3)}_{rel_{i+1}}(x)$ onde o valor final é o resultado da ponderação por $\mu_{A^j_{i+1}}(x)$.

No caso onde os pontos não são seqüenciais figura 4.10(b), verifica-se que pela distribuição dos pontos, a curva interpolada não deve ultrapassar o eixo **x** ($y_{i+1}(x) \leq Y^2_{i+1}(x)$). Logo, $y_{rel_{i+1}}(x)$ deve ser ≤ 0 e conseqüentemente, este valor deve variar entre $Y^{(2)}_{rel_{i+1}}(x)$ e $-Y^{(3)}_{rel_{i+1}}(x)$.

Deste modo, com a introdução desta última modificação, a nova forma da função $g(x)$ permite a obtenção de curvas fechadas sem acréscimo de informação da base de dados ou rotação de eixos. A restrição inicial é substituída por uma condição menos restritiva de que os pontos originais devem obdecer à condição,

$$x_i \neq x_{i+1} \neq x_{i+2} \neq x_{i+3}$$

para $i = 0, \ldots, n$. Esta restrição é necessária uma vez que pontos adjacentes e coincidentes podem gerar valores infinitos na função g.

A figura 4.11 mostra a interpolação de uma curva fechada, onde as distribuições não seqüenciais ocorrem nos dois extremos da curva.

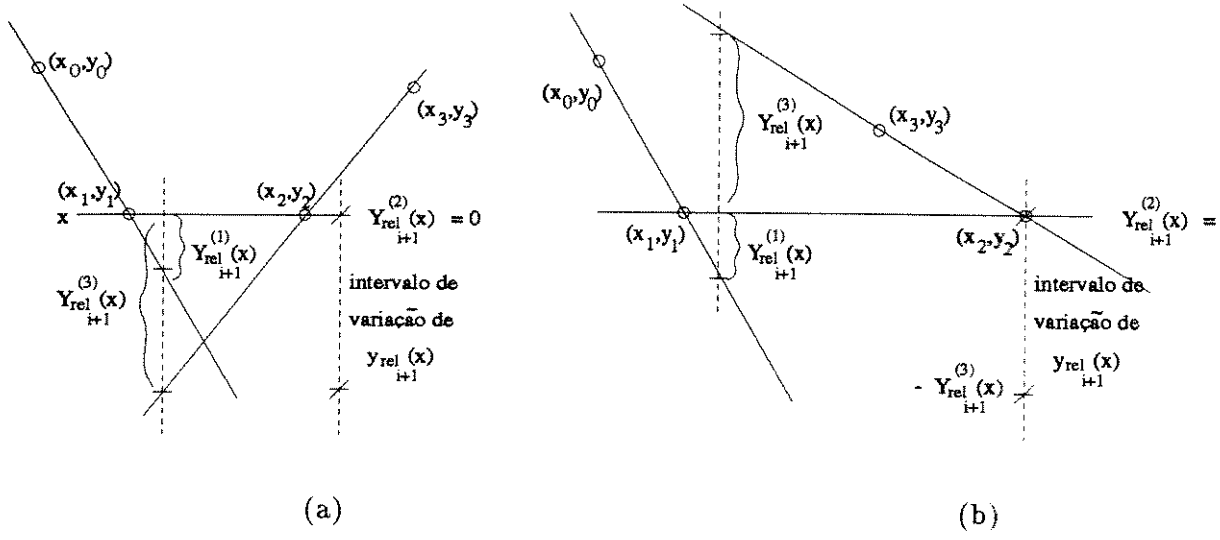


Figura 4.10: (a) Distribuição de pontos seqüenciais. (b) Distribuição de pontos não seqüenciais.

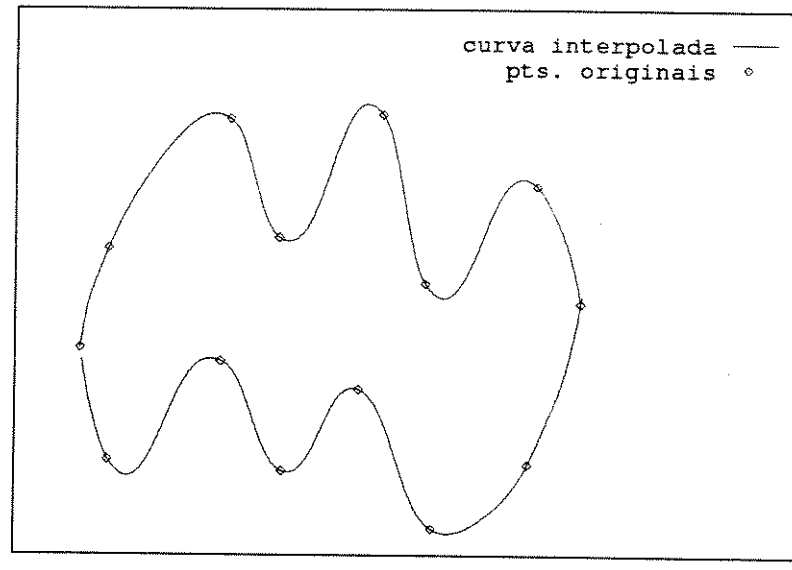


Figura 4.11: Exemplo de interpolação de curva fechada

Capítulo 5

Extração de Pontos Significativos

5.1 Introdução

A idéia de se representar uma curva por um conjunto pequeno de pontos (pontos significativos) podendo, posteriormente, recuperá-la através de um processo de interpolação motivou a concepção de paradigmas que solucionassem o problema da compressão.

Contudo, é importante salientar que todos os processos de compressão-descompressão requerem um compromisso entre a informação resultante e a mínima informação necessária para a recuperação, com um erro aceitável, dos dados originais.

Neste sentido, e através da análise do comportamento do método de interpolação nebulosa não linear, descrito na seção 4.3, foi possível se extrair o conhecimento de quais pontos seriam relevantes para o processo de interpolação. Esta análise, feita localmente para uma seqüência de cinco pontos como ilustra a figura 5.1, resultou num comportamento global (ao longo de toda a curva) bastante satisfatório.

Foram propostas duas soluções para a determinação dos padrões da figura 5.1. A primeira utiliza uma rede de processamento híbrido, como descrita na seção 2.4, cujo conhecimento já está contido na estrutura inicial da rede. A segunda proposta emprega uma rede neural artificial onde o modelo adotado é o "perceptron". Neste caso, o conhecimento é adquirido através de treinamento pelo algoritmo de aprendizado "back-propagation". As seções 5.2 e 5.3 descrevem os dois modelos propostos, assim como o processamento para a definição dos padrões de pontos significativos.

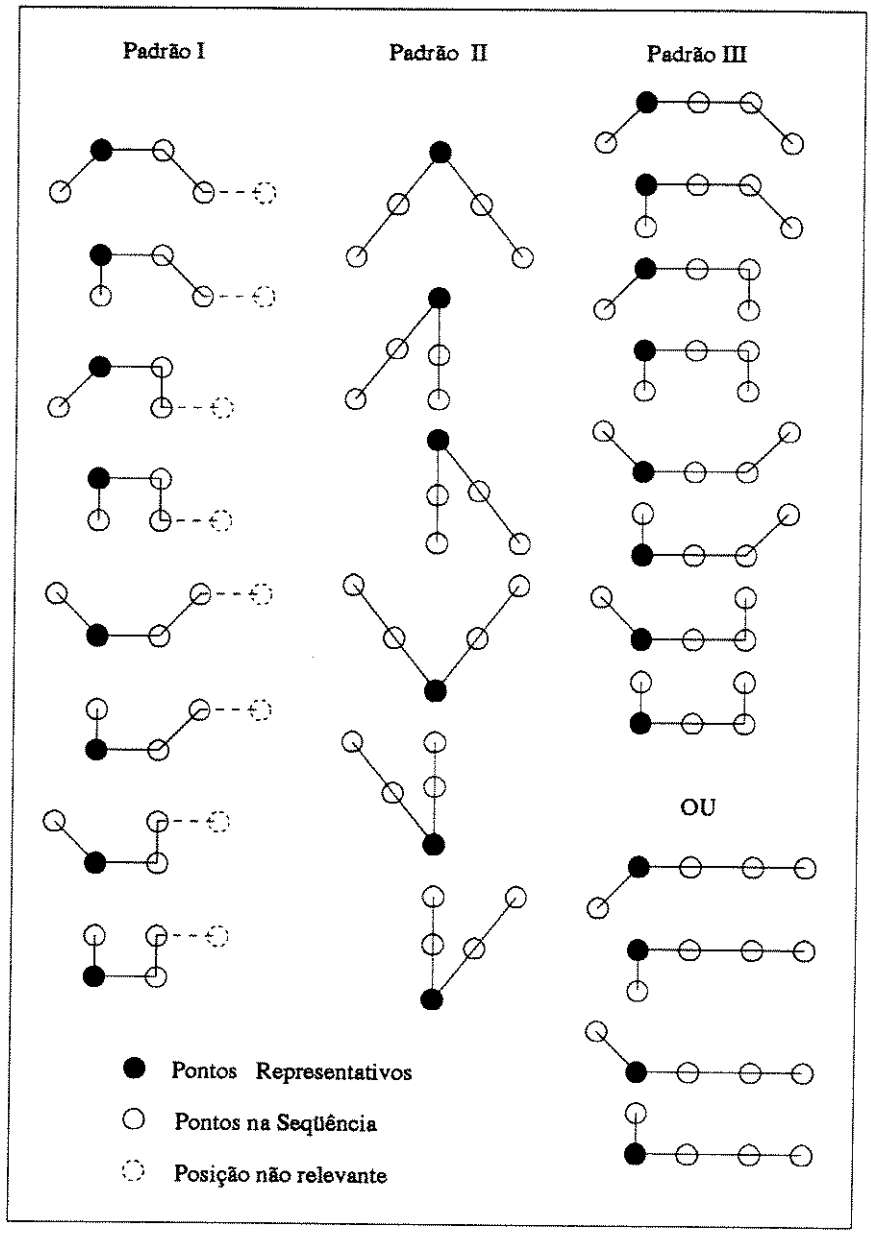


Figura 5.1: Classe de padrões que apresentam pontos significativos.

5.2 Sistema Neural para Compressão de Dados (SNCD)

O modelo do sistema neural para compressão de dados (SNCD), descrito em Regattieri *et al.* [RvZR93] e ilustrado na figura 5.2, foi desenvolvido para operar com dados de entrada em espaços bidimensionais. Supõe-se que a rede recebe, de forma seqüencial, os dados de entrada (x_i, y_i) $(i = 1, \cdots, n)$ do conjunto completo de dados originais e gera como saída o conjunto de dados comprimidos (pontos significativos). A rede realiza a tarefa de extração em quatro níveis:

- Nível 1: Dados dois pontos (x_i, y_i) e (x_{i+1}, y_{i+1}), determina-se uma direção (d_j) entre oito direções possíveis entre eles.
- Nível 2: Dadas duas direções adjacentes d_j e d_{j+1}, calcula-se a variação de direção vd_k como horária, anti-horária ou nula.
- Nível 3: Dadas três variações de direção seqüenciais vd_k, vd_{k+1}, vd_{k+2}, armazenadas no nível anterior, define-se a ocorrência ou não do ponto significativo (x_r, y_r) na seqüência - um ponto significativo é encontrado sempre que a seqüência assume uma das configurações específicas ilustradas na figura 5.1.
- Nível 4: Para uma memória que armazena os cinco últimos pontos da seqüência de entrada $S_e : (x_m, y_m)(m = i-3, i-2, \cdots, i+1)$, e caso exista um ponto representativo (x_r, y_r), este é dado pelo segundo ou terceiro ponto na seqüência $S_e : (r = i-2$ ou $r = i-1)$, de acordo com o mecanismo de decisão que será visto na seção a seguir.

5.2.1 Descrição do Processamento em SNCD

Em SNCD o princípio de computação do neurônio se baseia no acoplamento transmissor/receptor (t/r), que ativa um controlador c como mostra a relação a seguir:

$$t_i \wedge r_j \mapsto c_j.$$

Conforme foi visto na seção 2.4.3, o controlador c_j pode atuar nos neurônios pré e pós-sinápticos, e também nas células vizinhas. Por exemplo, eles podem operar como controladores, com o objetivo de realizar outros receptores pós-sinápticos r, para futuros acoplamentos com transmissores t (ver eq. 5.1); ou ainda, agindo no processo de decodificação, podem definir a quantidade de transmissores realizados pelo neurônio pós sináptico. A seguir, serão descritos os processamentos realizados em cada um dos quatro níveis da rede.

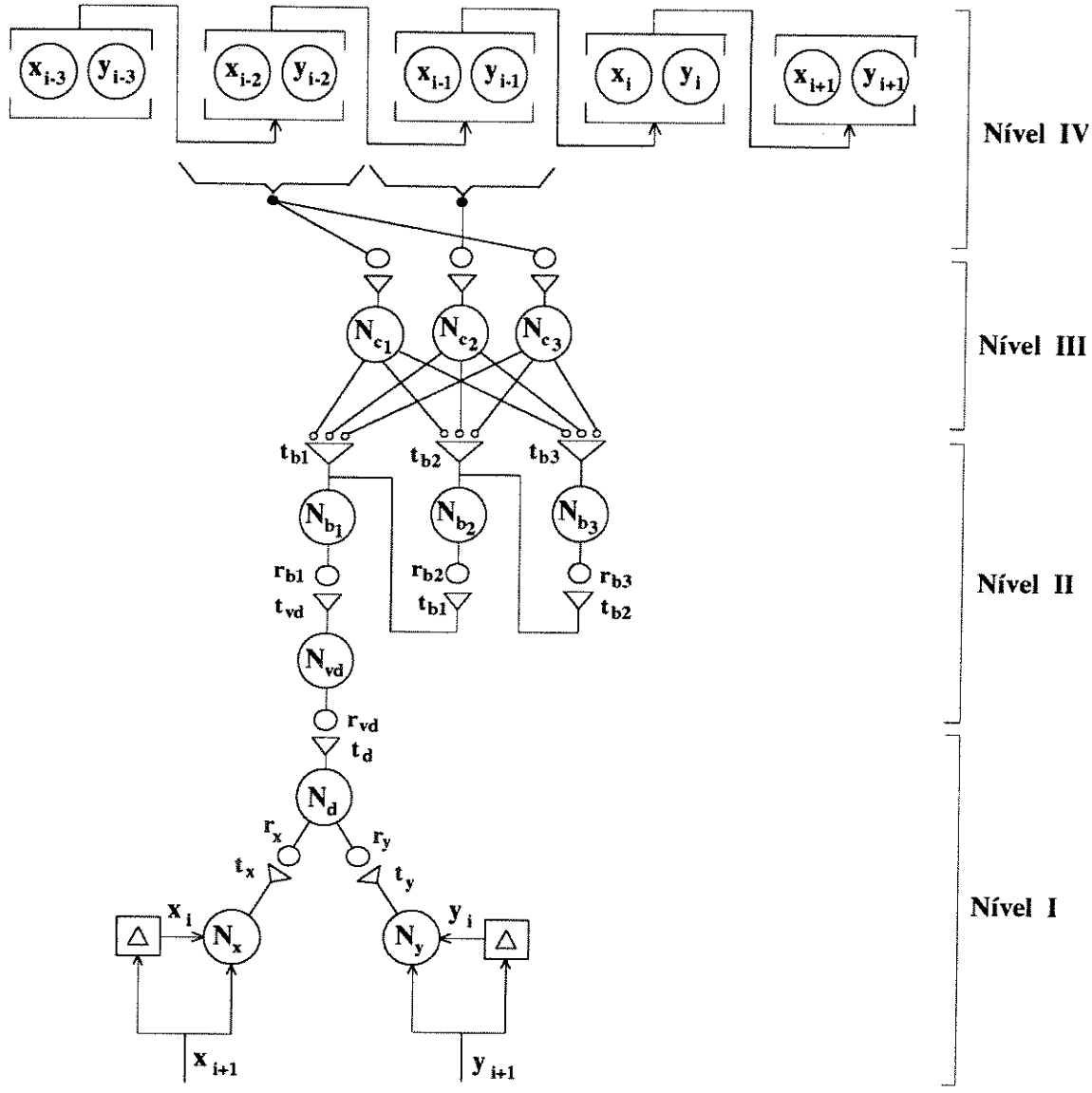


Figura 5.2: Modelo da rede para compressão de dados.

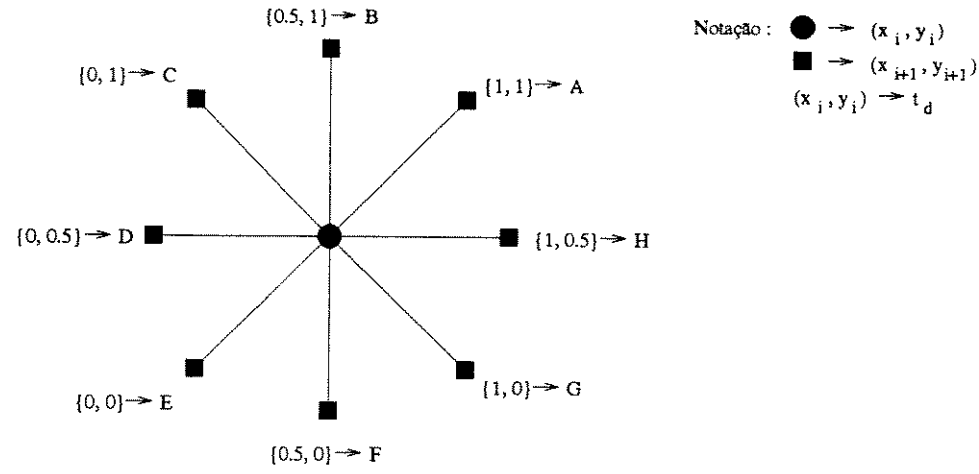


Figura 5.3: Posições relativas entre (x_i, y_i) e (x_{i+1}, y_{i+1}).

5.2.1.1 Nível I

O primeiro nível da rede é formada por três neurônios: dois neurônios sensoriais N_x e N_y e um neurônio de saída N_d. Este nível realiza a tarefa de determinar a posição de (x_{i+1}, y_{i+1}) em relação à entrada anterior (x_i, y_i), decidindo entre 8 posições relativas possíveis, como mostra a figura 5.3.

Os neurônios da primeira camada realizam os seguintes processamentos:

- A entrada anterior u_i ($u = x$ ou y) define os limiares α_1 e α_2 do neurônio N_u (N_x ou N_y), para computar a saída O_u, dependendo do valor da entrada atual u_{i+1}:

$$O_u = \begin{cases} 0 & \text{se} \quad u_{i+1} < \alpha_1 \\ 0.5 & \text{se} \quad \alpha_1 \leq u_{i+1} \leq \alpha_2 \\ 1 & \text{se} \quad u_{i+1} > \alpha_2 \end{cases}$$

  onde, $\alpha_1 = u_i - \varepsilon$; $\alpha_2 = u_i + \varepsilon$; $\varepsilon \to 0$.

  Esta computação é baseada no fato dos axônios pré-sinápticos possibilitarem diferentes contatos da entrada com os neurônios pós-sinápticos $N_u (u = x$ ou $y)$. Desta forma, a entrada anterior u_i ainda está disponível em um ramo do axônio quando a entrada u_{i+1} chega ao outro ramo.

- Os neurônios N_x e N_y enviam os transmissores t_x e t_y para se acoplarem aos receptores r_x e r_y, respectivamente, no neurônio N_d. De acordo com a eq. 2.18, a quantidade de transmissores realizadas por N_x e N_y são:

$$m_x = O_x \circ M(t_x); \qquad m_y = O_y \circ M(t_y);$$

para uma quantidade total de transmissores $M(t) = 1$. Assume-se que as quantidades de receptores no neurônio pós-sináptico N_d são iguais às quantidades total de transmissores dos neurônios pré-sinápticos. Então:

$$M(r_x) = M(t_x) = 1; \qquad M(r_y) = M(t_y) = 1.$$

As compatibilidades dos acoplamentos t_x/r_x e t_y/r_y são definidas como:

$$\mu(t_x, r_x) = 0.5; \qquad \mu(t_y, r_y) = 1;$$

De acordo com a eq. 2.19, tem-se:

$$v_x = m_x \wedge M(r_x) * \mu(t_x, r_x) \odot v_0$$

e

$$v_y = m_y \wedge M(r_y) * \mu(t_y, r_y) \odot v_0$$

Definindo-se $\circ$, $*$, $\wedge$ e $\odot$ como produto algébrico e $v_0 = 1$, tem-se:

$$v_x = 0.5\ O_x; \qquad v_y = O_y;$$

Então $$a_j = v_x + v_y = 0.5\ O_x + O_y$$

Seja a função de codificação g (eq. 2.16) dada pela função identidade; $\alpha_1 = 0$ e $\alpha_2 = 1.5$. Então,

$$O_d = \begin{cases} \text{erro} & \text{se} \quad a_j < 0 \\ \text{erro} & \text{se} \quad a_j > 1.5 \\ g(a_j) & \text{se} \quad 0 \leq a_j \leq 1.5 \end{cases}$$

Deste modo, o neurônio N_d realiza o processo de codificação, resultando em *oito* tipos diferentes de transmissores t_d. Cada um desses transmissores representa uma das posições relativas (direções) definidas na figura 5.3. A tabela 5.1 traz o mapeamento que transforma as saídas codificadas O_x e O_y no transmissor t_d realizado pelo neurônio N_d.

O_x \ O_y	1	0.5	0
1	A	H	G
0.5	B		F
0	C	D	E

Tabela 5.1: Mapeamento $(O_x, O_y) \rightarrow t_d$.

r_{vd} \ t_d	A	B	C	D	E	F	G	H
A	Z	P	P	P	-	N	N	N
B	N	Z	P	P	P	-	N	N
C	N	N	Z	P	P	P	-	N
D	N	N	N	Z	P	P	P	-
E	-	N	N	N	Z	P	P	P
F	P	-	N	N	N	Z	P	P
G	P	P	-	N	N	N	Z	P
H	P	P	P	-	N	N	N	Z
Φ	Φ	Φ	Φ	Φ	-	Φ	Φ	Φ

Tabela 5.2: Função de associação no neurônio N_{vd}: $t_d \wedge r_{vd} \mapsto c_{vd}$.

5.2.1.2 Nível II

Esta camada é composta por *quatro* neurônios: N_{vd}, N_{b_1}, N_{b_2} e N_{b_3} . O objetivo deste nível é calcular, através do neurônio N_{vd}, as variações de direção e armazenar, nos neurônios N_{b_k} $(k = 1, 2, 3)$, as três últimas variações calculadas. Este processo é realizado como se descreve a seguir:

- O transmissor $t_d = A$ ou B ou $\cdots$ ou H, enviado pelo neurônio N_d, se acopla ao receptor pós-sináptico $r_{vd} = \Phi$ ou A ou $\cdots$ ou H, localizado no neurônio N_{vd}. Desta forma o acoplamento t_d/r_{vd} ativa o controlador c_{vd}.
- O controlador c_{vd}, por sua vez, atua sobre o próprio neurônio N_{vd} realizando as seguintes ações:

  1. Ativa N_{vd} para realizar o transmissor t_{vd} de acordo com a tabela 5.2. A relação a seguir ilustra este processo:

  $$t_d \wedge r_{vd} \mapsto c_{vd} \Rightarrow t_{vd} = Z \text{ ou } N \text{ ou } P \text{ ou } \Phi$$

  onde

  Z → direção inalterada

  N → variação horária

  P → variação anti-horária

  Φ → informação insuficiente

2. Modifica o receptor r_{vd} de acordo com a relação:

$$t_d \wedge r_{vd} \mapsto r_{vd} = t_d \tag{5.1}$$

Assim o neurônio N_{vd} classifica a variação de direção vd_k no tipo do transmissor t_{vd}, e ainda, a informação sobre a última direção detectada d_k é armazenada no receptor r_{vd}.

- O transmissor t_{vd} realizado por N_{vd} se acopla ao receptor r_{b_1} do neurônio N_{b_1} e em conseqüência, o controlador c_{b_1} é ativado. Este controlador age tanto nos neurônios vizinhos N_{b_3} e N_{b_2}, como no neurônio pós-sináptico N_{b_1}.

$$t_{vd} \wedge r_{b_1} \mapsto c_{b_1} \Rightarrow \begin{matrix} N_{b_3} \\ N_{b_2} \\ N_{b_1} \end{matrix}$$

O controlador c_{b_1} tem por principais objetivos definir o transmissor realizado por N_{b_3} no instante t_{i+1} igual ao transmissor realizado por N_{b_2} no instante t_i; e de forma análoga, definir t_{b_2} igual a t_{b_1} no instante anterior, para somente depois, agir sobre N_{b_1}. Portanto, a ação de c_{b_1} se dá de três maneiras:

1. Age sobre o neurônio N_{b_3} de modo que o seu transmissor seja igual ao transmissor do neurônio N_{b_2} no intante anterior, ou seja,

$$t_{b_3} \mid_{t_{i+1}} = t_{b_2} \mid_{t_i} .$$

2. De forma análoga, age sobre N_{b_2} de modo que

$$t_{b_2} \mid_{t_{i+1}} = t_{b_1} \mid_{t_i} .$$

3. Age sobre o neurônio pós-sináptico N_{b_1} definindo o transmissor t_{b_1} igual ao transmissor pré-sináptico t_{vd}.

Esta dinâmica resulta na codificação das três últimas variações de direção pelos transmissores t_{b_1}, t_{b_2} e t_{b_3}. A ordem indica que t_{b_1} representa a mudança atual (vd_k em t_{i+1}), t_{b_2} a mudança com atraso de um tempo de processamento (vd_k em t_i) e finalmente, t_{b_3} determina a mudança com atraso 2 (vd_k em t_{i-1}).

5.2.1.3 Nível III

Esta camada, composta por três neurônios N_{c_1}, N_{c_2} e N_{c_3}, decide se a seqüência das coordenadas dos pontos de entrada contém um ponto significativo. Em caso afirmativo, define-se ainda qual o tipo (padrões dados na fig. 5.1). O processo de decisão e especificação do ponto significativo se realiza como será descrito a seguir:

Os transmissores $t_{b_k}(k = 1,2,3)$ dos neurônios N_{b_k} do nível II se acoplam aos receptores r_{c_j} dos neurônios $N_{c_j}(j = 1,2,3)$. Estes neurônios de codificação binária apresentam como saída o valor 1, para o caso de presença de ponto significativo e 0 caso contrário, segundo a equação,

$$o_{c_1} = \begin{cases} 1 & \text{se } t_{b_2} = t_{b_3} = N \text{ ou } P \\ 0 & \text{de outra maneira} \end{cases}$$

$$o_{c_2} = \begin{cases} 1 & \text{se } t_{b_1} = t_{b_3} = Z \text{ e } t_{b_2} \neq Z \\ 0 & \text{de outra maneira} \end{cases} \tag{5.2}$$

$$o_{c_3} = \begin{cases} 1 & \text{se} \begin{cases} (t_{b_2} = Z \text{ e } t_{b_1} = t_{b_3} \neq Z) \\ \text{ou} \\ (t_{b_3} \neq Z \text{ e } t_{b_2} = t_{b_1} = Z) \end{cases} \\ 0 & \text{de outra maneira} \end{cases}$$

Como mostra a equação 5.2, apenas um neurônio é ativado a cada tempo de processamento (ativação mutuamente exclusiva), onde se tem:

N_{c_1} ativo $\rightarrow$ padrão I
N_{c_2} ativo $\rightarrow$ padrão II
N_{c_3} ativo $\rightarrow$ padrão III
N_{c_1} N_{c_2} N_{c_3} inativos $\rightarrow$ ausência de ponto significativo.

A figura 5.4 ilustra a seqüência de cinco pontos de entrada, assim como o conjunto de transmissores realizados pelos neurônios $N_{b_j}(j = 1,2,3)$, no instante t_{i+1}.

De acordo com a equação 5.2, verifica-se que os valores de $t_{b_j}(j = 1,2,3)$ na fig. 5.4(a) ativam o neurônio N_{c_3}, indicando a ocorrência de um ponto significativo do tipo do padrão III. Para a fig. 5.4(b), com os valores de t_{b_j} nenhum neurônio $N_{c_j}(j = 1,2,3)$ é ativado indicando assim a ausência de ponto significativo.

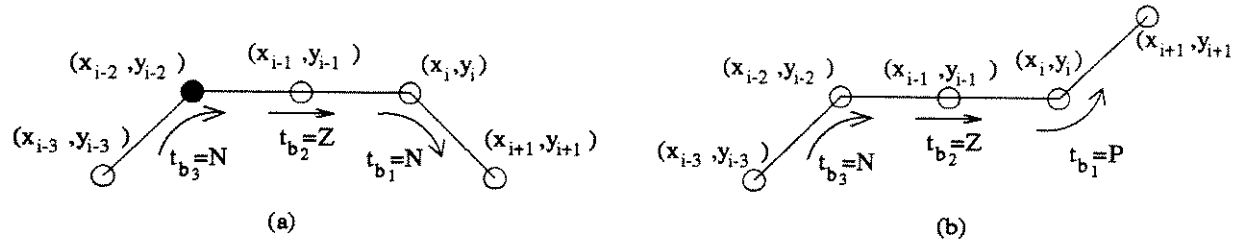


Figura 5.4: Rede simbólica (a) padrão que contém um ponto representativo; (b) padrão que **não** contém um ponto representativo.

5.2.1.4 Nível IV

Esta camada é formada por cinco pares de neurônios de memória $(N_{x_j}, N_{y_j})(j = i-3, i-2, \cdots, i+1)$ cuja função é armazenar os últimos cinco pares de entrada (x_j, y_j). Portanto, estes neurônios realizam um processamento semelhante ao realizado pelos neurônios $N_{b_j}(j = 1, 2, 3)$ na camada II.

As conexões entre os níveis III e IV definem o processo de decisão que seleciona qual a posição do ponto significativo associado ao padrão classificado no nível III, de acordo com a relação a seguir:

padrão I e III (neurônios N_{c_1} e N_{c_3} ativos) $\rightarrow (x_r, y_r) = (x_{i-2}, y_{i-2})$.
padrão II (neurônio N_{c_2} ativo) $\rightarrow (x_r, y_r) = (x_{i-1}, y_{i-1})$.

Este processo de codificação consiste, basicamente, da definição do grau de compatibilidade $\mu(t/r)$ entre os transmissores dos neurônios pré-sinápticos $N_{c_j}(j = 1, 2, 3)$ do nível anterior com os transmissores dos neurônios pós-sinápticos $(N_{x_j}, N_{y_j})(j = i-3, i-2, \cdots, i+1)$. Define-se:

$$\mu(t_{c_1}, r_{x_j}) = \mu(t_{c_1}, r_{y_j}) = \begin{cases} 1 & \text{se } j = i-2 \\ 0 & \text{de outro modo} \end{cases}$$

$$\mu(t_{c_2}, r_{x_j}) = \mu(t_{c_2}, r_{y_j}) = \begin{cases} 1 & \text{se } j = i-1 \\ 0 & \text{de outro modo} \end{cases}$$

$$\mu(t_{c_3}, r_{x_j}) = \mu(t_{c_3}, r_{y_j}) = \begin{cases} 1 & \text{se } j = i-2 \\ 0 & \text{de outro modo} \end{cases}$$

É interessante salientar que a tarefa de determinação da posição do ponto significativo representa o mapeamento de *três* entradas (padrão I, II e III) em *duas* saídas (posição 1 e 2). Desta forma, conclui-se que seriam necessários apenas *dois* neurônios $N_{c_k}(k = 1, 2)$, ou seja, os padrões I e III se fundiriam, já que representam a mesma posição. Entretanto, a título de explanação e com o intuito de simplificar a função de ativação dada pela eq. 5.2, foram mantidos os três neurônios $N_{c_j}(j = 1, 2, 3)$.

5.2.2 Dinâmica do Processamento

Considerando-se uma seqüência de entrada como a dos pontos na fig. 5.4(a), têm-se os senguintes passos na dinâmica do processamento do sistema SNCD:

PASSO 0: (Inicialização)

- todas as variáveis são inicializadas com Φ.
- entradas presentes nos axônios pré-sinápticos: Φ e (x_{i-3}, y_{i-3}).

PASSO 1:

- entrada: $(x_{i-3}, y_{i-3}), (x_{i-2}, y_{i-2})$
- O neurônio N_d realiza o transmissor $t_d = A$ (mapeamento da tabela 5.1).
- O acoplamento t_d/r_{vd} ativa o controlador c que age:

  1. definindo o transmissor t_{vd}, segundo a tabela 5.2, como:

     $A \wedge \Phi \mapsto c \Rightarrow t_{vd} = \Phi$

  2. definindo o receptor r_{vd} como:
     $A \wedge \Phi \mapsto c \Rightarrow r_{vd} = t_d = A$

- $t_{b_1} = t_{vd} = \Phi$
- $t_{b_2} = t_{b_1}(\text{PASSO } 0) = \Phi$
- $t_{b_3} = t_{b_2}(\text{PASSO } 0) = \Phi$
- $o_{c_1} = o_{c_2} = o_{c_3} = 0$
- $(x_r, y_r) = \Phi$

PASSO 2:

- entrada: $(x_{i-2}, y_{i-2}), (x_{i-1}, y_{i-1})$
- N_d realiza $t_d = H$.
- O acoplamento t_d/r_{vd} ativa o controlador c que age:

$$H \wedge A \mapsto c \Rightarrow \begin{cases} t_{vd} = N \\ r_{vd} = t_d = H \end{cases}$$

- $t_{b_1} = t_{vd} = N$
- $t_{b_2} = t_{b_1}(\text{PASSO } 1) = \Phi$
- $t_{b_3} = t_{b_2}(\text{PASSO } 1) = \Phi$
- $o_{c_1} = o_{c_2} = o_{c_3} = 0$
- $(x_r, y_r) = \Phi$

PASSO 3:

- entrada: $(x_{i-1}, y_{i-1}), (x_i, y_i)$
- N_d realiza $t_d = H$.
- O acoplamento t_d/r_{vd} ativa o controlador c que age:

$$H \wedge H \mapsto c \Rightarrow \begin{cases} t_{vd} = Z \\ r_{vd} = t_d = H \end{cases}$$

- $t_{b_1} = t_{vd} = Z$
- $t_{b_2} = t_{b_1}(\text{PASSO } 2) = N$
- $t_{b_3} = t_{b_2}(\text{PASSO } 2) = \Phi$
- $o_{c_1} = o_{c_2} = o_{c_3} = 0$
- $(x_r, y_r) = \Phi$

PASSO 4:

- entrada: $(x_i, y_i), (x_{i+1}, y_{i+1})$
- N_d realiza $t_d = G$.
- O acoplamento t_d/r_{vd} ativa o controlador c que age:

$$G \wedge H \mapsto c \Rightarrow \begin{cases} t_{vd} = N \\ r_{vd} = t_d = G \end{cases}$$

- $t_{b_1} = t_{vd} = N$
- $t_{b_2} = t_{b_1}(\text{PASSO } 3) = Z$
- $t_{b_3} = t_{b_2}(\text{PASSO } 3) = N$
- $o_{c_1} = o_{c_2} = 0; \quad o_{c_3} = 1;$
- $(x_r, y_r) = (x_{i-2}, y_{i-2})$

5.3 Rede Neural Artificial para Compressão de Dados

Esta proposta de solução via processamento numérico se baseia no treinamento de uma rede modelo "perceptron" (seção 2.3), de forma que, finalizado o processo de aprendizado, a rede seja capaz de classificar os padrões de entrada conforme a figura 5.1, no caso da ocorrência de ponto significativo. A rede deve ainda determinar a ausência de ponto significativo.

5.3.1 Estrutura da Rede

A rede neural artificial é composta por três camadas; uma cada de entrada C_1 composta por *quinze* elementos processadores que recebem as entradas de treinamento α_1, α_2 e α_3. Estas entradas representam as variações de direção como será mostrado na seção seguinte. A camada intermediária C_2 é formada por *dez* elementos processadores, todos conectados aos neurônios da camada de entrada C_1. Finalmente, a camada de saída C_3 é composta por apenas *uma* unidade processadora.

Esta estrutura foi obtida com base nos resultados de simulações com diversos modelos e parâmetros de rede. Observou-se o compromisso entre a topologia mais simples possível, que representasse um menor custo computacional, e a estrutura mais adequada para a extração dos pontos visando ao melhor conjunto de pontos representativos.

5.3.2 Padrões de Entrada para o Treinamento

Os padrões de entrada da rede são definidos para uma seqüência de cinco pares de coordenadas (x_i, y_i), $i = 1, \cdots, 5$. Para a obtenção final das variações de direção (α), primeiramente são calculadas as direções (ângulos) entre os pontos adjacentes (x_i, y_i) e (x_{i+1}, y_{i+1}), $i = 1, \cdots, 4$. Deste modo, para cada seqüência, obtêm-se quatro direções d_j, $j = 1, \cdots, 4$.

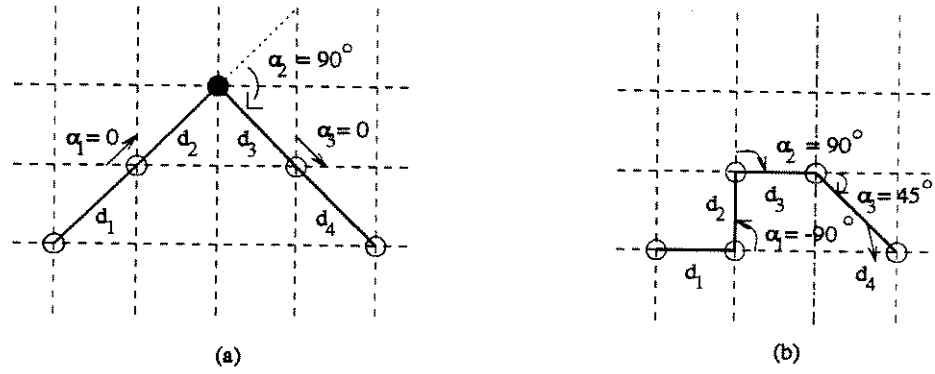


Figura 5.5: Rede numérica. (a) padrão que contém um ponto significativo; (b) padrão que não contém um ponto significativo.

As variações de direção $\alpha_k (k = 1, \cdots, 3)$ são então calculadas, com base na mudança de trajetória sofrida ao se percorrer a seqüência de pontos, ou seja, o quanto a direção d_j se desvia para atingir a direção d_{j+1}. A convenção adotada neste caso resulta em variações de direção positivas ($\alpha_k > 0$) para mudanças horárias e variações negativas para mudanças anti-horárias. As figuras 5.5(a) e (b) ilustram seqüências de cinco pontos, percorridos da esquerda para a direita, juntamente com os respectivos valores de $\alpha_k (k = 1, 2, 3)$, onde somente a seqüência da figura 5.5(a) define um ponto significativo.

A base de dados de treinamento foi montada a partir de um processo de geração de combinações para variações (α) de 45 em 45°, sujeitas a determinadas restrições, como por exemplo, a de que o módulo da soma dos ângulos ($\alpha_1 + \alpha_2 + \alpha_3$) não deve ultrapassar 270°. Estas restrições têm por objetivo gerar um conjunto menor para o treinamento da rede, levando-se em conta somente as entradas possíveis de ocorrerem nos padrões de imagem. Por exemplo, o padrão da figura 5.6(a) atende inteiramente à restrição anterior, já o padrão da fig. 5.6(b) atende à restrição na condição de limite, ou seja, qualquer variação maior que 135° em um dos extremos passa a excluí-lo da base de dados.

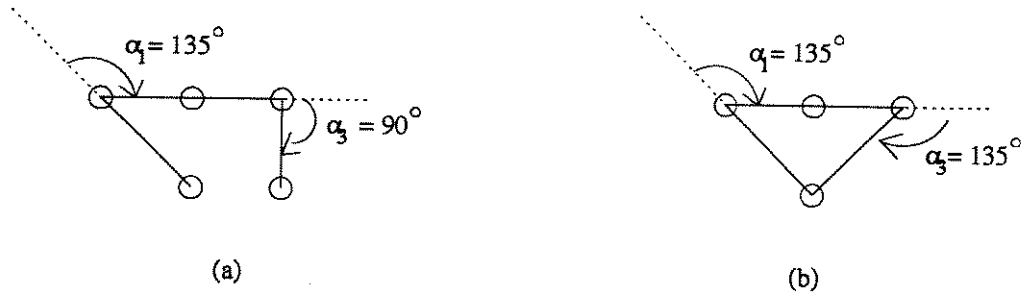


Figura 5.6: (a) Padrão pertencente à base de dados de treinamento, pois $\alpha_1 + \alpha_2 + \alpha_3 \leq 270$; (b) padrão limite pertencente à base de dados de treinamento, pois $\alpha_1 + \alpha_2 + \alpha_3 = 270$.

Deste modo são geradados 322 padrões de entrada, onde os valores de α_k são convertidos de graus para radianos e normalizados. Esta normalização é obtida fazendo-se com que a soma das entradas $\alpha_k(k = 1, 2, 3)$, ponderadas pelos respectivos pesos, se limite ao intervalo $[-4, 4]$. Desta forma eliminam-se, para os neurônios da camada 1, os efeitos de saturação uma vez que os valores das entradas destes neurônios estão fora dos intervalos de saturação da função sigmóide, como mostra a figura 5.8.

A figura 5.7 traz os padrões I e III definindo um único padrão de saída, ilustrado em 5.7(a). O padrão II é mostrado na figura 5.7(b) e 5.7(c) determina as entradas que não resultam em ponto significativo. Os padrões de entradas foram distribuídos em espaços tridimensionais de acordo com as variáveis α_1, α_2 e α_3, resultando em quatro regiões disjuntas. Portanto, de acordo com a seção 2.3.1.2, existe a necessidade de uma rede com três níveis de elementos processadores, capaz de gerar regiões mais complexas.

5.3.3 Codificação da Saída

A função de ativação sigmóide gera, como saída do neurônio, valores contidos dentro do intervalo $[0, 1]$. Desta forma, a saída foi dividida em três regiões (fig. 5.8), onde a primeira região $\in [0, 0.3)$ indica a presença de ponto significativo do tipo dos padrões I ou III; a segunda região $\in [0.3, 0.5]$ determina a ausência de ponto significativo; e por fim, a última região $\in (0.5, 1]$ representa as seqüências que contêm um ponto significativo do tipo do padrão II. Conforme visto na seção 5.2.1.4, os padrões I e III são redundantes do ponto de vista de posicionamento na seqüência de cinco pontos, uma vez que determinam a mesma posição relativa (segunda posição). Por isso, na fase de treinamento da rede "perceptron" proposta, os padrões I e III se tornam um só, visto que determinam a mesma saída desejada.

5.3.4 Treinamento do "Perceptron"

Utilizando-se o algoritmo *backpropagation*, os pesos da rede, após a inicialização com variações de $[-0.5, 0.5]$, são alterados segundo a equação:

$$W_{k+1} = W_k + \eta \delta_k X_k + \alpha \Delta w_k \tag{5.3}$$

onde, $\alpha = 0.1$ e $\eta = 2\mu$.

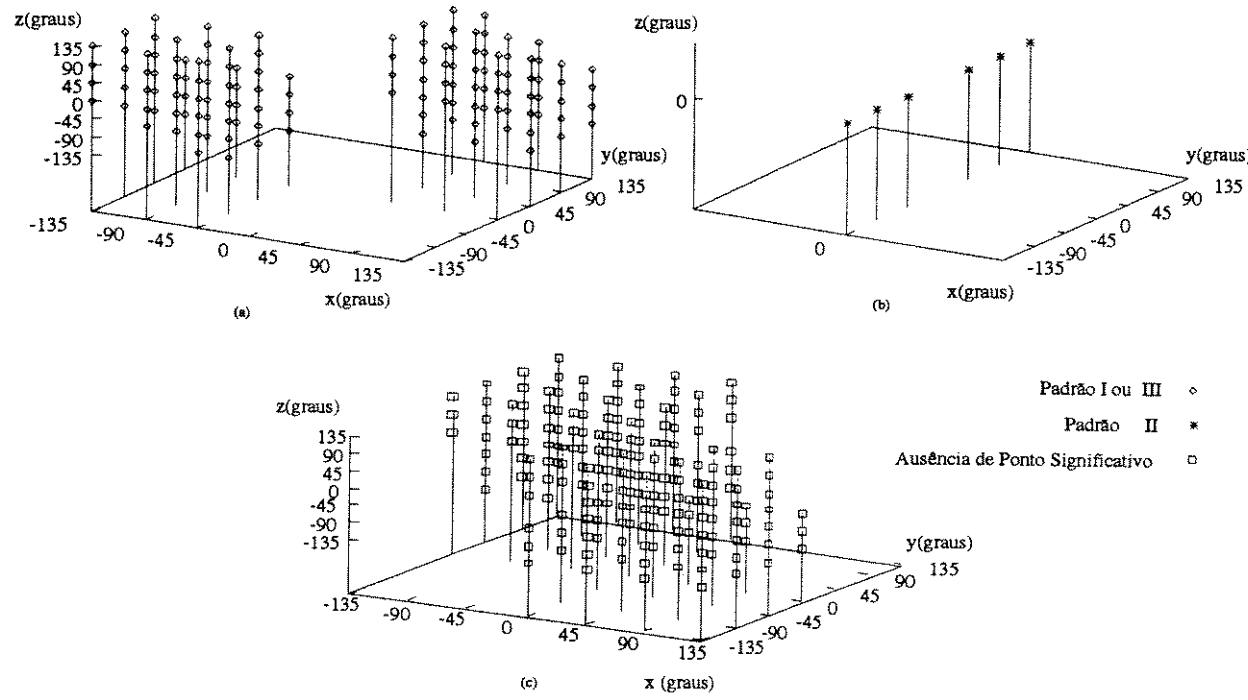


Figura 5.7: Distribuição espacial dos padrões de entrada; (a) padrão I e III (b) Padrão II; (c) padrões que não representam pontos significativos.

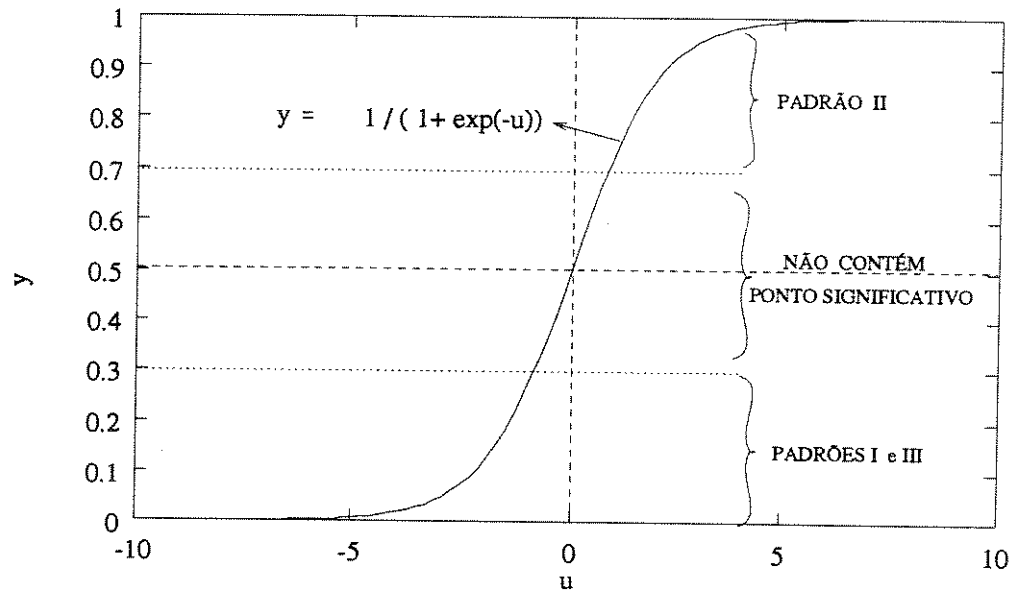


Figura 5.8: Codificação da saída.

A equação anterior difere da equação 2.4, pela introdução do termo de momento[1] [RM86].

Então, a cada passo de treinamento, um padrão k é apresentado à entrada da rede e caso o resultado (saída desejada - saída real) ultrapasse um limiar aceitável (limiar de treinamento), os pesos são corrigidos para o padrão k.

5.3.5 Exemplo de Aplicação

O exemplo ilustrado a seguir tem por objetivo mostrar a eficiência do método de extração de pontos significativo, assim como, comparar os resultados obtidos via processamento numérico e simbólico aos pontos extraídos por um especialista.

A figura 5.9(a) traz a curva original com os pontos significativos obtidos pelo conhecimento do especialista. É importante salientar que este conjunto de pontos significativos representa o conjunto ideal, uma vez que a curva original é obtida pelo processo inverso, ou seja, o especialista determina um conjunto de pontos não redundantes, com base no conhecimento do método de interpolação INNL, e obtém a curva original pela interpolação do conjunto de pontos escolhido. Portanto, para se obter uma curva interpolada com o mínimo de erro possível o método deve buscar um conjunto o mais próximo do ideal (figura 5.9(a)).

Para os dois casos (simbólico e numérico) as entradas são dadas pelos pares (x, y) da curva original. Através de uma grade sobreposta à curva original e com espaçamento horizontal e vertical iguais a 10, aplica-se um algoritmo de seqüenciamento dos pontos, utilizando o critério de busca por vizinhaça [GW87]. Deve-se salientar que este tratamento não tem por finalidade ordenar os pontos de forma restritiva (estritamente crescente por exemplo), mas apenas dar uma ordem para que estes sejam apresentados à rede. Conseqüentemente, obtém-se um conjunto de pares ordenados

$$(x_k, y_k) \qquad k = 1, \cdots, N$$

onde N é função da grade $(X \times Y)$ definida para o seqüenciamento. Portanto, cada seqüência de entrada (x_j, y_j) $(j = i-3, \cdots, i+1)$ é obtida fazendo-se i variar de 4 a $N-1$.

A figura 5.9(b) mostra o resultado da extração dos pontos significativos via processamento simólico e a curva interpolada pelo método INNL, e 5.9(c) traz a comparação com os dados do especialista. O resultado da interpolação dos pontos significativos obtidos pelo

[1] A introdução deste fator visa, fundamentalmente, evitar os efeitos oscilatórios, filtrando as variações de alta freqüência da superfície de erro.

método numérico está ilustrado em 5.9(d). E 5.9(e) traz a comparação com a curva original e conjunto de pontos de especialista. Verifica-se que os conjuntos de pontos extraídos pelos métodos simbólico e numérico são idênticos e bastante próximos do conjunto obtido pelo especialista.

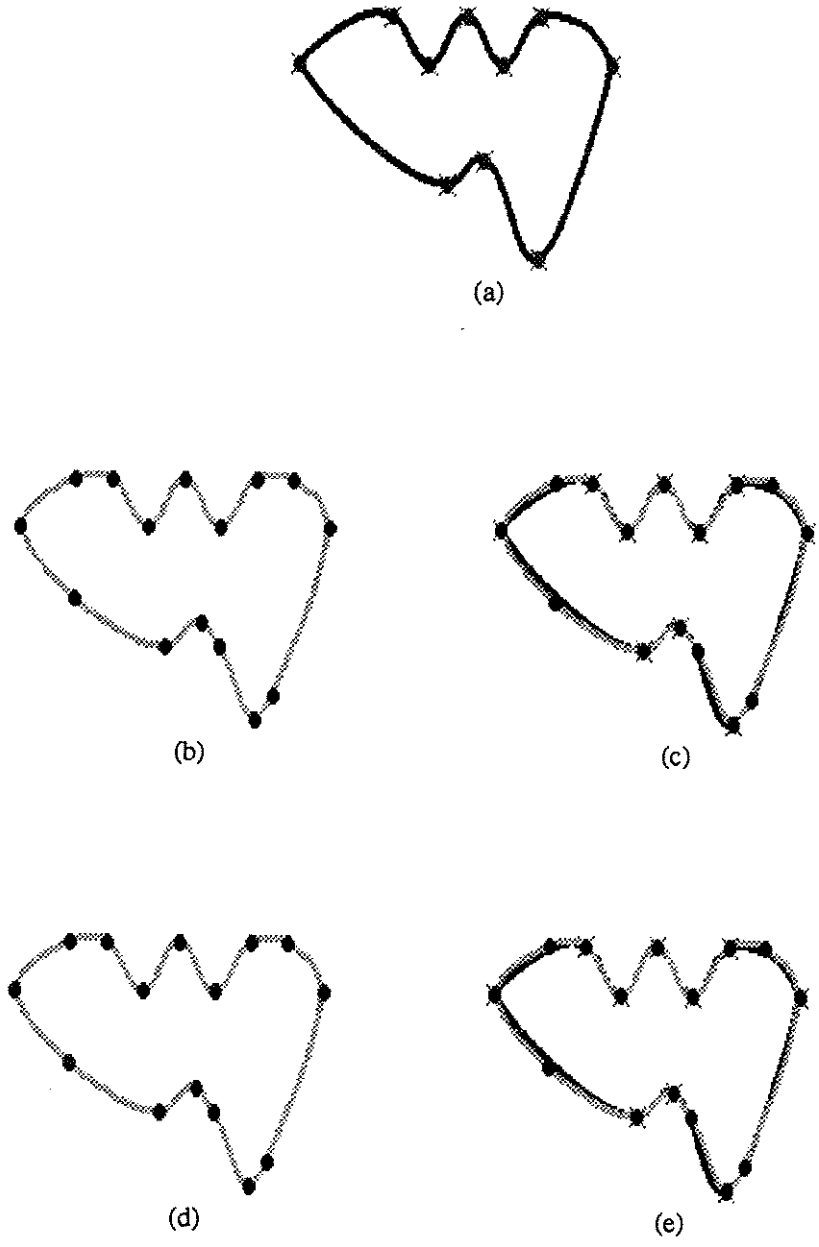


Figura 5.9: (a) Curva original; (b) curva interpolada pelo método INNL e pontos significativos definidos por SNCD; (c) sobreposição da curva original (a) e curva interpolada (b). (d) curva interpolada e pontos significativos extraídos pela rede numérica. (e) comparação do resultado das curvas original (a) e interpolada (d).

Capítulo 6

Análise de Resultados

6.1 Introdução

Este capítulo consiste dos resultados de simulações obtidos nas etapas de interpolação e extração de pontos significativos que definem este trabalho. A seção 6.2 traz os resultados da etapa de interpolação e a seção 6.3 mostra os resultados do processo de extração de pontos significativos.

Com o objetivo de comprovar a eficiência do método de interpolação INNL, a seção 6.2.1 apresenta uma comparação dos resultados da interpolação pelo método proposto por Uchino *et al.* [UYY90], com os resultados obtidos pelo método INNL. A seção 6.2.2 compara o método INNL com os métodos tradicionais de interpolação por partes - splines cúbicos e polinômios cúbicos.

A segunda parte, descrita na seção 6.3, traz a comparação dos resultados encontrados pelas duas soluções propostas para a extração de pontos significativos - solução via rede simbólica e solução via rede numérica. A seção 6.3.1 compara os conjuntos de pontos significativos obtidos pelos dois métodos, para diferentes grades. Nesta seção são testadas grades retangulares (valores diferentes nos eixos horizontal X e vertical Y).

Tendo em vista a comprovação do método de extração de pontos significativos, a seção 6.3.2 mostra os conjuntos de pontos extraídos de uma curva pelos métodos simbólico e numérico e comparados a um conjunto ideal. O conceito de ideal, neste caso, representa o melhor conjunto de pontos de forma a se obter uma curva interpolada exatamente igual à curva original.

6.2 Interpolação

6.2.1 Comparação dos Métodos de Interpolação INL e INNL

O principal objetivo desta seção é demonstrar a maior eficiência do método de Interpolação Nebulosa Não Linear, proposto neste trabalho, em relação ao método INL que utiliza regras lineares para a interpolação.

A figura 6.1(a) traz a função y interpolada por INL, para pontos igualmente espaçados com $\Delta = 1$, comparada com a função original $f(x) = x^3$ e 6.2(a) mostra a interpolação g obtida pelo método INNL. A tabela 6.1 ilustra os erros cometidos pelos dois métodos. Neste caso, obtêm-se melhores resultados utilizando-se o método INNL, tanto em relação ao erro médio quanto ao erro máximo.

Método de Interpolação	Função Original	Erro Médio %	Erro Máximo %
INL	$f(x) = x^3$	0.7	2.07
INNL	$f(x) = x^3$	0.09	0.3
INL	$f(x) = x^4$	2.09	7.66
INNL	$f(x) = x^4$	0.36	1.54
INL	$f(x) = 1/(1+x^2)$	2.82	16
INNL	$f(x) = x^3$	0.75	6.68

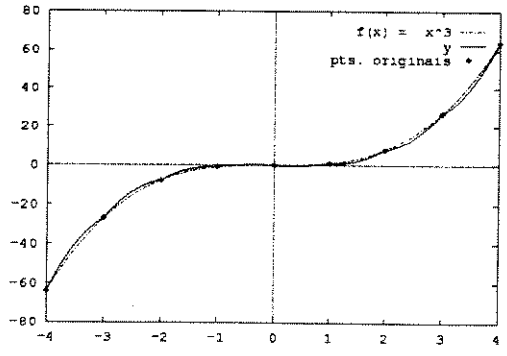
Tabela 6.1: Erro de interpolação.

Um outro resultado comprovado, conforme visto na seção 4.3.2, é uma maior suavidade da curva interpolada. Ao contrário dos saltos na derivada y' da curva obtida por INL (ver fig. 6.1(b)), a derivada g' da curva interpolada por INNL é contínua, como mostra a figura 6.2(b).

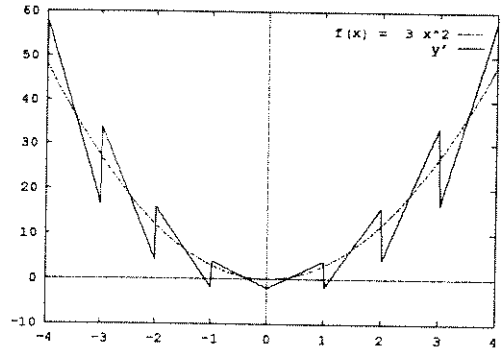
Resultados semelhantes são encontrados para a função $f(x) = x^4$, com espaçamento $\Delta = 1$. A figura 6.3(a) traz o resultado da interpolação por regras lineares INL. Novamente, além de menores erros de interpolação (ver tabela 6.1), a descontinuidade da primeira derivada é eliminada como mostram as figuras 6.3(b) e 6.4(b).

Uma nova simulação é feita para o caso de pontos não igualmente espaçados, com função original dada por $f(x) = 1/(1+x^2)$. As figuras 6.5 e 6.6 comparam os resultados encontrados para os métodos INL e INNL com erros médio e máximo dados na tabela 6.1. A

análise comparativa do erro e das derivadas, dadas nas figuras 6.5(b) e 6.6(b), demonstram a eficiência do método INNL também para as situações de pontos não igualmente espaçados.

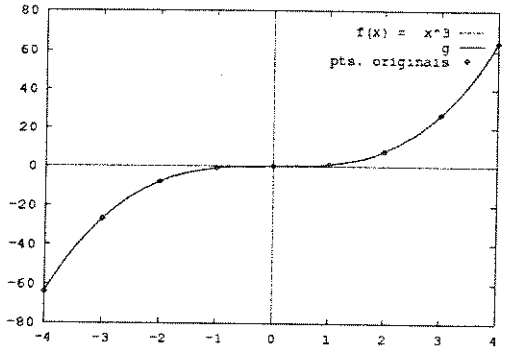


(a)

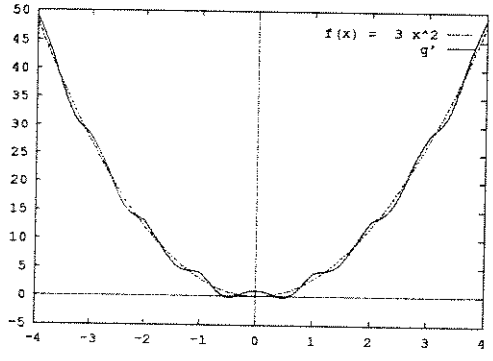


(b)

Figura 6.1: (a) Função original $f(x) = x^3$ e função interpolada y pelo método INL. (b) Derivada $f'(x) = 3x^2$ e a derivada y' da função interpolada.

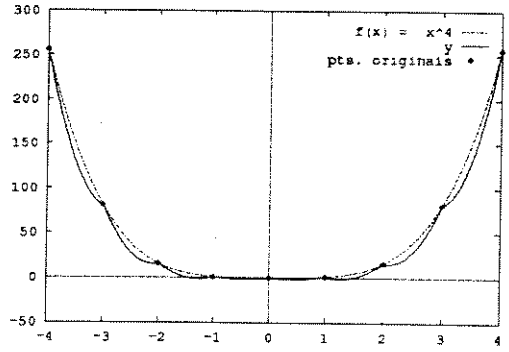


(a)

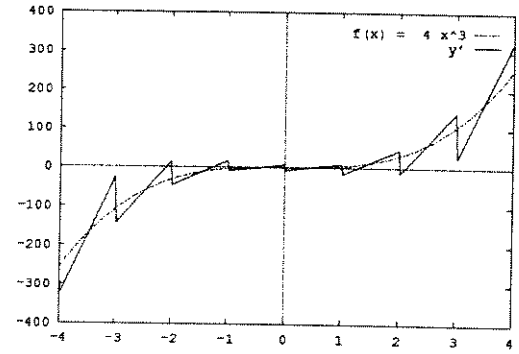


(b)

Figura 6.2: (a) Função original $f(x) = x^3$ e função interpolada g pelo método INNL. (b) Derivada $f'(x)' = 3x^2$ e a derivada g' da função interpolada.

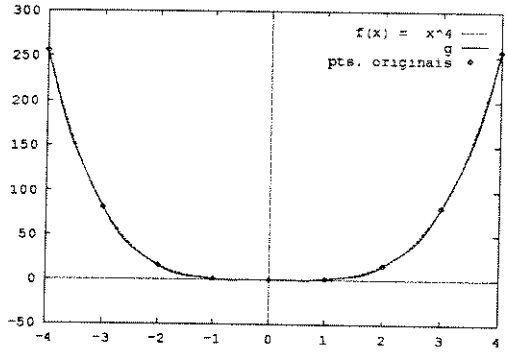


(a)

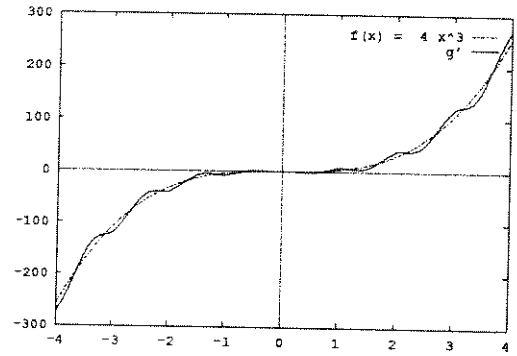


(b)

Figura 6.3: (a) Função original $f(x) = x^4$ e função interpolada y pelo método INL. (b) Derivada $f'(x) = 4x^3$ e a derivada y' da função interpolada.

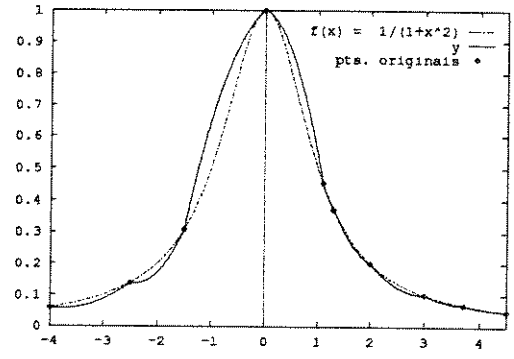


(a)

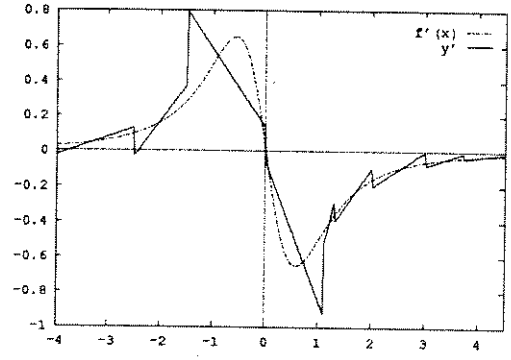


(b)

Figura 6.4: (a) Função original $f(x) = x^4$ e função interpolada g pelo método INNL. (b) Derivada $f'(x) = 4x^3$ e a derivada g' da função interpolada.

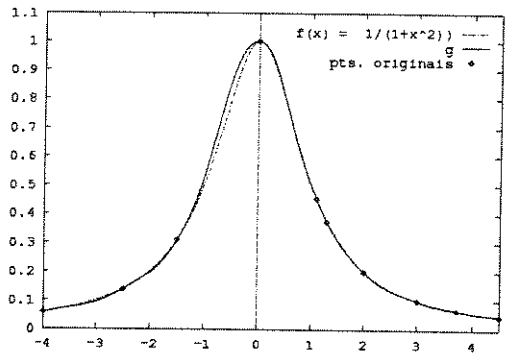


(a)

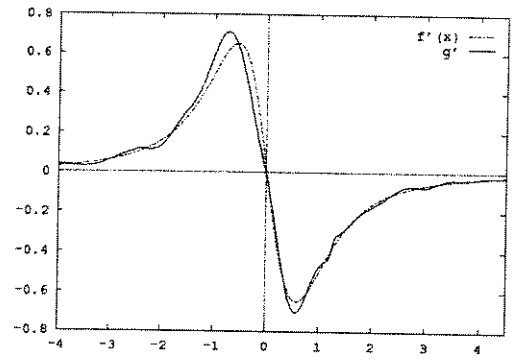


(b)

Figura 6.5: (a) Função original $f(x) = 1/(1+x^2)$ e função interpolada y pelo método INL. (b) Derivada $f'(x)$ e a derivada y' da função interpolada.



(a)



(b)

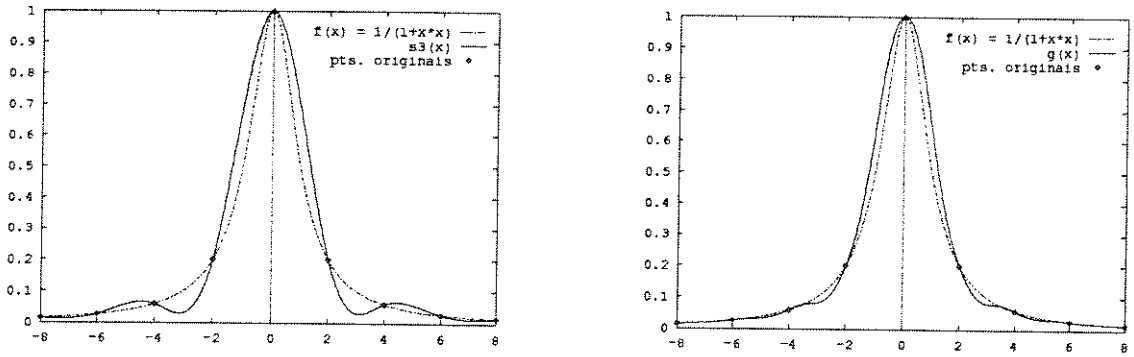
Figura 6.6: (a) Função original $f(x) = 1/(1+x^2)$ e função interpolada g pelo método INNL. (b) Derivada $f'(x)$ e a derivada g' da função interpolada.

6.2.2 Comparação do Método INNL com os Métodos Tradicionais de Interpolação por Partes

Nesta seção, os resultados para o método INNL serão apresentados, a nível de precisão e suavidade da curva interpolada, e comparados aos obtidos pelos métodos tradicionais de interpolação por partes - splines cúbicos e polinômios cúbicos por partes.

A tabela 6.2 traz os erros cometidos ao se interpolarem funções diversas pelos métodos tradicionais. Estes erros são comparados ao erro resultante da interpolação por INNL. Verifica-se que, na maioria dos casos, tanto o erro médio quanto o máximo estão bastante próximos para os três métodos. Através da análise da tabela 6.2, nota-se que o método INNL apresentou melhores resultados que a interpolação por splines para os casos das funções $1/(1+x^2)$, seno, cosseno, seno hiperbólico e cosseno hiperbólico. Para as funções sigmóide e tangente hiperbólica, os métodos tradicionais se mostraram mais eficientes.

A figura 6.7 ilustra a interpolação da função $1/(1+x^2)$ por splines cúbicos (fig. 6.7(a)) e pelo método INNL (fig. 6.7(b)). Conforme a tabela 6.2, os pontos estão igualmente espaçados com $\Delta = 2$.



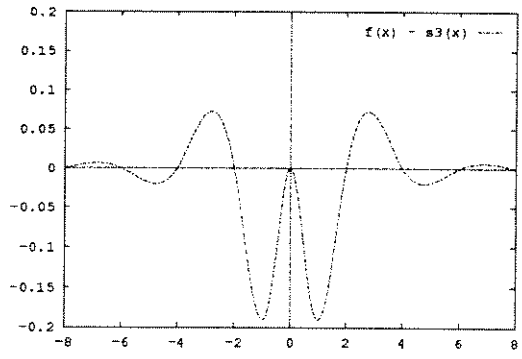
(a) (b)

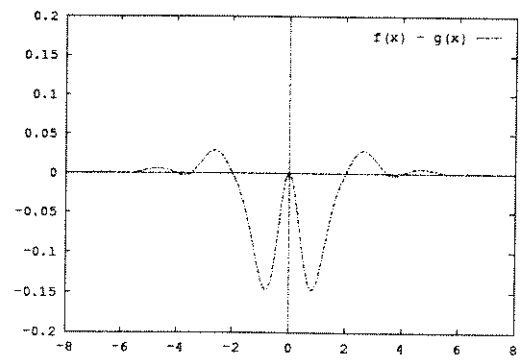
Figura 6.7: (a) Função original $f(x) = 1/(1+x^2)$ e função interpolada pela spline cúbica $s_3(x)$. (b) Função original $f(x) = 1/(1+x^2)$ e função interpolada g pelo método INNL.

Método	Função	Intervalo de Interpolação	*Delta*(Δ)	Erro Médio %	Erro Máximo %
INNL	$1/(1+x^2)$	[-8,8]	2	2.28	14.82
SPLINE				3.4	19.19
CÚBICA				2.02	16.15
INNL	$1/(1+e^{-x})$	[-8,8]	2	0.49	2.42
SPLINE				0.29	0.15
CÚBICA				0.38	2.42
INNL	$\tanh(x)$	[-10,10]	2	1.74	12.60
SPLINE				1.42	9.78
CÚBICA				1.45	11.70
INNL	$\sinh(x)$	[-10,10]	2	1.75	15.65
SPLINE				1.98	14.0
CÚBICA				1.22	11.93
INNL	$\cosh(x)$	[-10,10]	2	3.5	31.31
SPLINE				3.95	28.01
CÚBICA				2.44	23.87
INNL	$\sin(x)$	$[-7\pi/2, 7\pi/2]$	π	1.60	2.87
SPLINE				1.60	8.73
CÚBICA				0.45	1
INNL	$\cos(x)$	$[-4\pi, 4\pi]$	π	1.60	2.87
SPLINE				1.50	8.73
CÚBICA				0.46	1

Tabela 6.2: Erro da interpolação dada por INNL, polinômios splines cúbicos e polinômios cúbicos por partes

Os erros de interpolação resultantes da aplicação dos métodos spline cúbicos e INNL, estão ilustrados nas figuras 6.8(a) e 6.8(b), respectivamente.

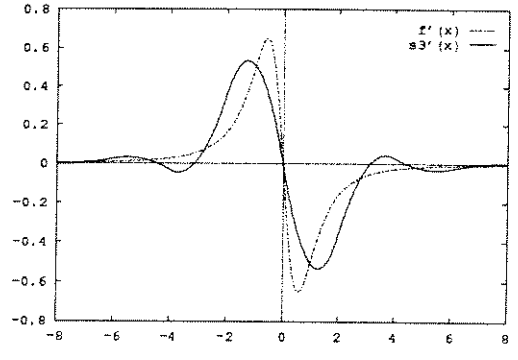


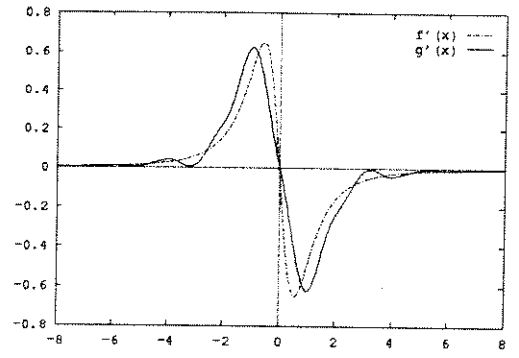


(a) (b)

Figura 6.8: (a) Erro: $(f(x) - s_3(x))$. (b) Erro: $(f(x) - g(x))$.

As figuras 6.9(a) e 6.9(b) comparam as derivadas s_3' e g' com a derivada $f'(x)$ da função original $f(x) = 1/(1+x^2)$.

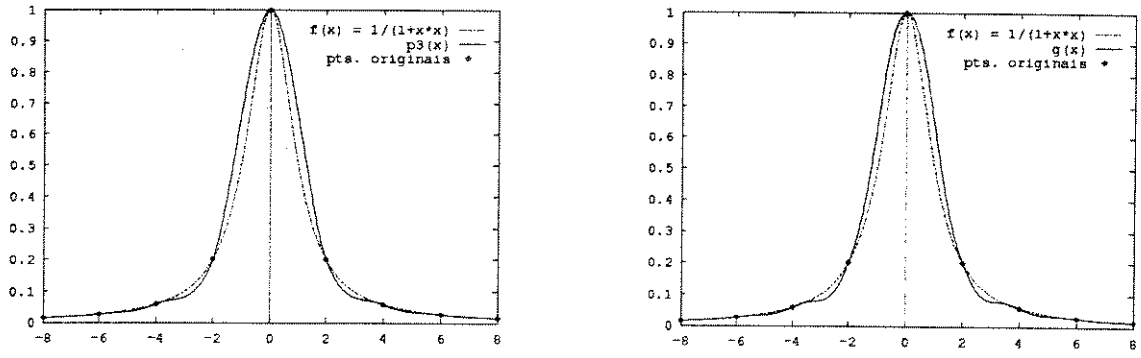




(a) (b)

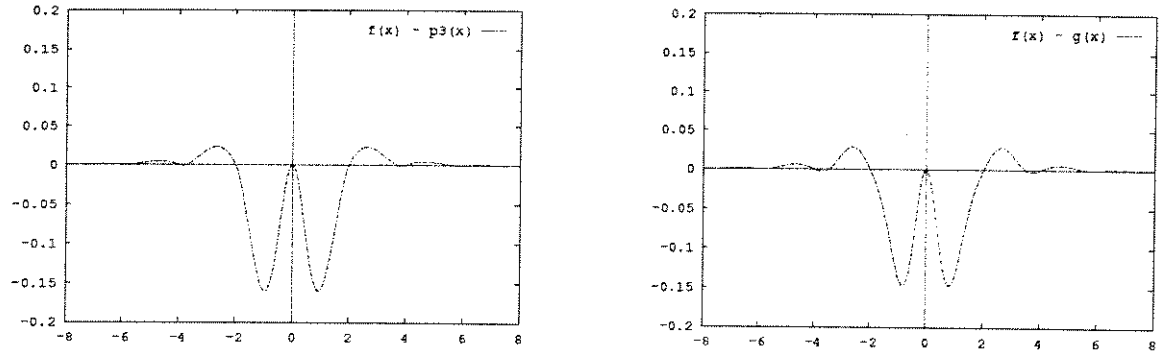
Figura 6.9: (a) Derivada $f'(x)$ da função original e a derivada $s_3'(x)$ da função spline interpolada. (b) Derivada $f'(x)$ da função original e a derivada g' da função interpolada por INNL.

De forma similar à comparação anterior, os resultados da interpolação pelos métodos polinômios cúbicos por partes e INNL estão ilustrados na figura 6.10, com erro de interpolação dado em 6.11.



(a) (b)

Figura 6.10: (a) Função original $f(x) = 1/(1+x^2)$ e função interpolada pelo polinômio $p_3(x)$. (b) Função original $f(x) = 1/(1+x^2)$ e função interpolada g pelo método INNL.



(a) (b)

Figura 6.11: (a) Erro: $(f(x) - p_3(x))$. (b) Erro: $(f(x) - g(x))$.

6.3 Extração de Pontos Significativos

6.3.1 Definição dos Pontos Significativos para Grades Retangulares

Esta seção objetiva o teste comparativo dos dois métodos para extração de pontos significativos (simbólico e numérico), para grades retangulares. Isto é necessário para se avaliar, principalmente, a capacidade de abstração da rede numérica, tendo em vista que as situações treinadas foram sempre de grades quadradas (10×10), ou seja, ângulos de mudança de direção múltiplos de 45°.

A figura 6.12 mostra o resultado do processo de extração de pontos significativos e interpolação da curva a partir destes pontos, para uma grade $X = 10$ e $Y = 15$. A curva original 6.12(a) é redefinida para uma nova seqüência de pontos, através da grade $X \times Y$, conforme mostra a figura 6.12(b). Como descrito na seção 5.3.5, as seqüências de entrada das redes (simbólica ou numérica) são formadas por cinco pontos adjacentes, obtidos da seqüência ilustrada em 6.12(b). Neste caso, os pontos significativos extraídos para os dois sistemas são idênticos (ver fig. 6.12(c)). A interpolação da curva pelo método INNL, a partir dos pontos significativos, é comparada em 6.12(d), à curva curva original.

A tabela 6.3 ilustra o resultado do processo de extração de pontos significativos da curva original dada em 6.12(a), para diferentes grades $X \times Y$. Esta tabela traz apenas situações onde os resultados obtidos pelos dois métodos são idênticos. Alguns casos onde isto não ocorre são mostrados na tabela 6.4.

A figura 6.13 representa a seqüência para uma grade 10×20, dada na tabela 6.4. A parte em destaque ilustra a i-ésima seqüência de cinco pontos de entrada. Verifica-se que a rede simbólica (fig. 6.13(a)) reconhece esta seqüência como um padrão do tipo I definido na seção 5.1 (fig. 5.1), onde o ponto indicado pela seta representa o ponto significativo. Já para a rede numérica nenhum ponto significativo é detectado, ou seja, o padrão é classificado como tendo ausência de ponto significativo, conforme mostra a figura 6.13(b).

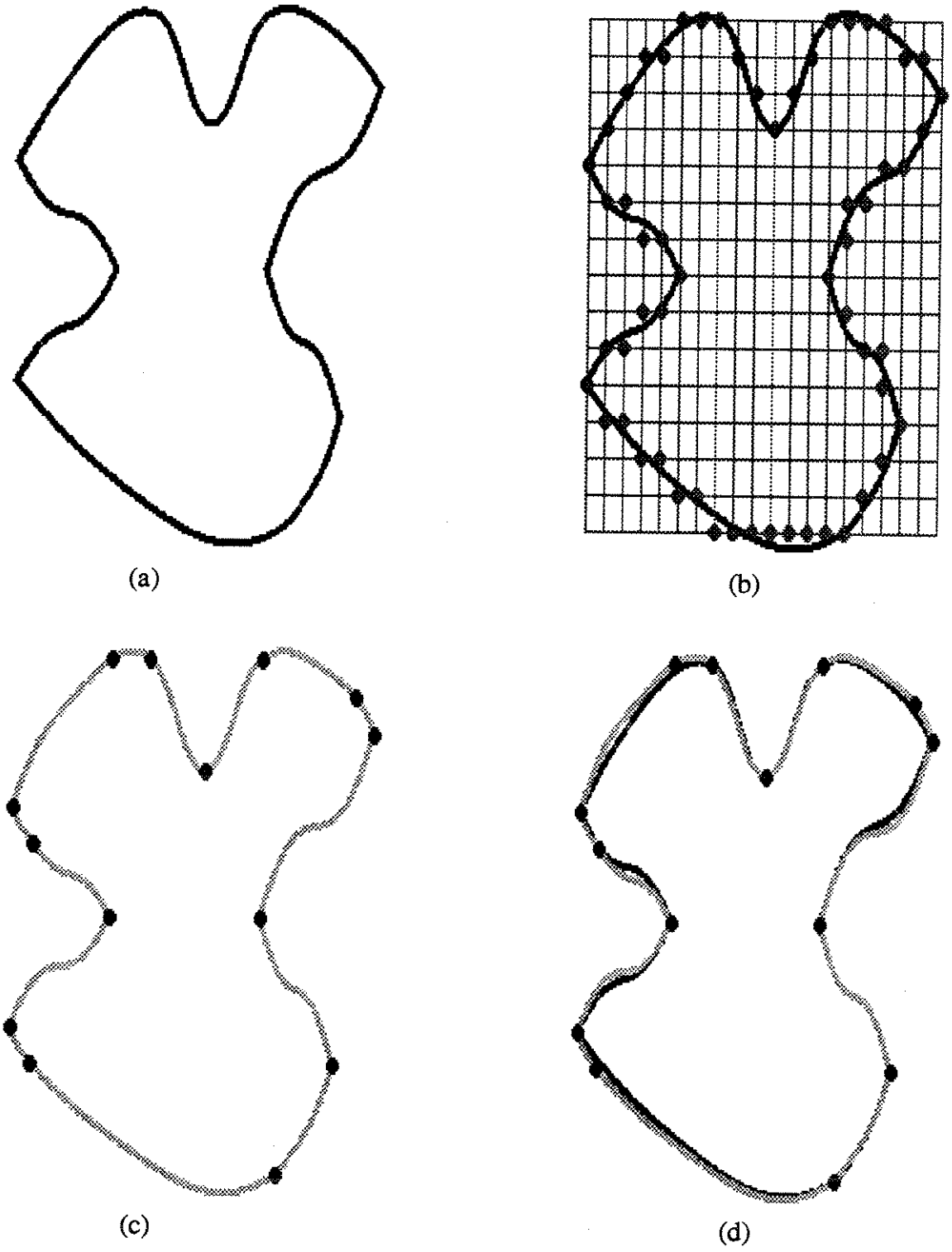


Figura 6.12: (a) Curva original; (b) Seqüenciamento dos pontos através da grade 10×15 (c) Curva interpolada a partir dos pontos significativos (d) Comparação da curva original e curva interpolada.

Método	Grade ($X \times Y$)	Número de Pontos Significativos	Erro Médio %	Erro Máximo %
Simb = Num	10×15	14	0.55	2.73
Simb = Num	5×10	21	1.14	4.56
Simb = Num	10×5	23	1.40	12.33
Simb = Num	40×10	12	3.80	15
Simb = Num	15×10	13	4.08	32.42
Simb = Num	10×30	17	4.11	22.64
Simb = Num	10×40	15	6.42	33.33

Tabela 6.3: Resultados idênticos dos métodos simbólico e numérico para a figura 6.12

Método	Grade ($X \times Y$)	Número de Pontos Significativos	Erro Médio %	Erro Máximo %
Simbólico	10×20	18	2.03	19.63
Numérico		17	2.10	19.63
Simbólico	25×10	15	2.56	17.35
Numérico		14	2.69	17.35
Simbólico	10×25	16	2.78	13.69
Numérico		15	3.16	18.26
Simbólico	30×10	14	3.95	14.15
Numérico		13	3.98	14.61

Tabela 6.4: Resultados diferentes dos métodos simbólico e numérico para a figura 6.12

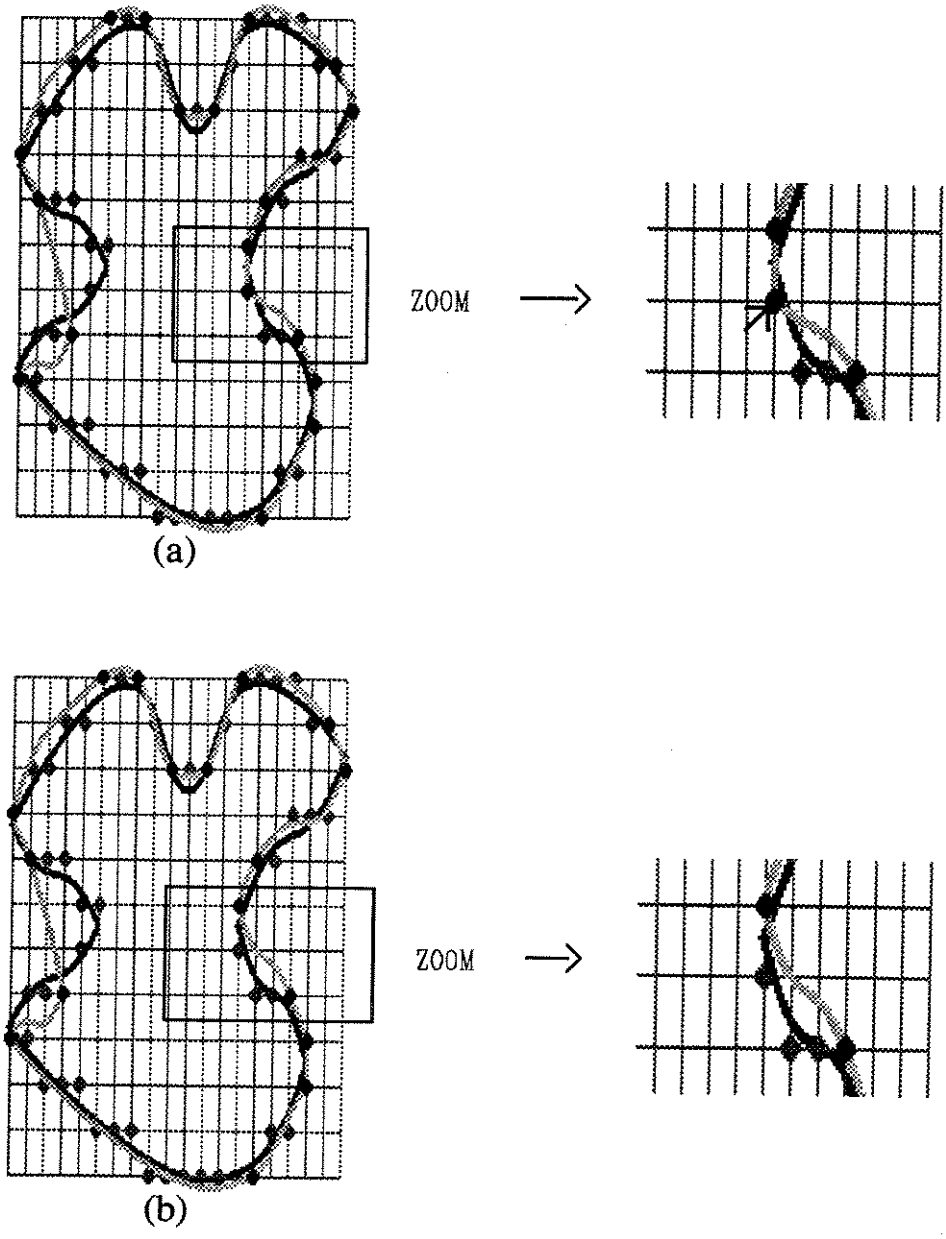


Figura 6.13: (a) Resultado da extração pela rede simbólica e interpolação, com destaque para a i-ésima seqüência que apresenta ponto significativo. Grade 10×20. (b) Extração pela rede numérica, para grade 10×20 e interpolação. A seqüência em destaque não apresenta ponto significativo.

A figura 6.14 mostra o processo de extração de pontos significativos para a grade retangular 15×10, com resultados idênticos para os dois métodos. Novamente, a interpolação passando pelos pontos significativos extraídos (ver fig. 6.14(c)) é comparada, em 6.14(d), à curva original.

A tabela 6.5 traz os resultados do processo de extração (número de pontos) e interpolação (erro de interpolação) da curva dada na figura 6.14(a). Diferentes grades são testadas e os resultados são idênticos.

Uma última simulação é feita para a curva original ilustrada na figura 6.15(a). As tabelas 6.6 e 6.7 trazem os resultados idênticos e diferentes, respectivamente, para os dois métodos de extração de pontos significativos.

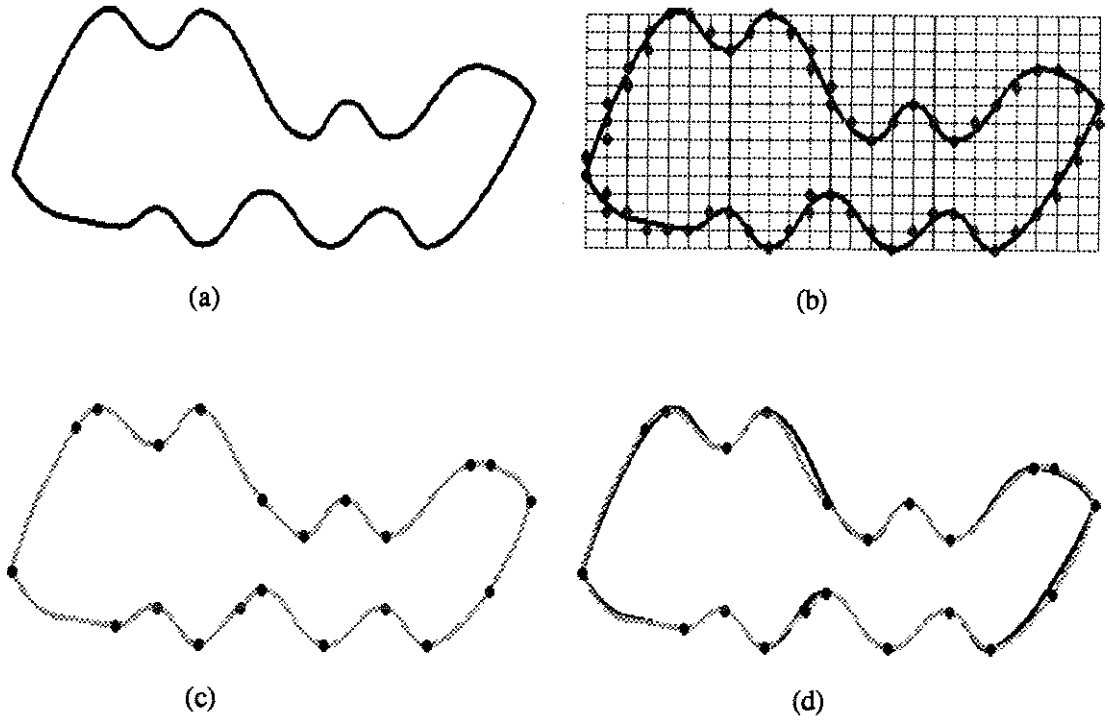


Figura 6.14: (a) Curva original; (b) Seqüenciamento dos pontos através da grade 15×10 (c) Curva interpolada a partir dos pontos significativos (d) Comparação da curva original e curva interpolada.

Método	Grade $(X \times Y)$	Número de Pontos Significativos	Erro Médio %	Erro Máximo %
Simb = Num	15×10	21	0.80	7.57
Simb = Num	10×5	37	1.47	6.06
Simb = Num	5×10	24	2.67	7.57
Simb = Num	20×10	23	3.37	18.18
Simb = Num	15×20	19	3.66	14.39
Simb = Num	10×15	16	4.15	23.48
Simb = Num	20×15	15	4.48	17.42
Simb = Num	10×20	15	5.59	22.72
Simb = Num	10×25	17	6.69	22.72
Simb = Num	10×40	10	10.91	49.24

Tabela 6.5: Resultados idênticos dos métodos simbólico e numérico para a figura 6.14

Método	Grade ($X \times Y$)	Número de Pontos Significativos	Erro Médio %	Erro Máximo %
Simb = Num	15×20	17	0.39	2.31
Simb = Num	5×10	31	0.87	16.56
Simb = Num	10×5	33	1.07	14.5
Simb = Num	15×10	26	1.13	3.64
Simb = Num	10×15	27	1.31	6.62
Simb = Num	20×15	23	1.83	15.43

Tabela 6.6: Resultados idênticos dos métodos simbólico e numérico para a figura 6.15

Método	Grade ($X \times Y$)	Número de Pontos Significativos	Erro Médio %	Erro Máximo %
Simbólico	25×10	19	2.55	21.52
Numérico		18	2.55	21.80
Simbólico	30×10	17	3.41	38.74
Numérico		16	3.61	38.74
Simbólico	10×25	19	3.62	15.89
Numérico		16	3.96	17.22
Simbólico	10×30	21	3.9	14.23
Numérico		19	3.96	14.23

Tabela 6.7: Resultados diferentes dos métodos simbólico e numérico para a figura 6.15

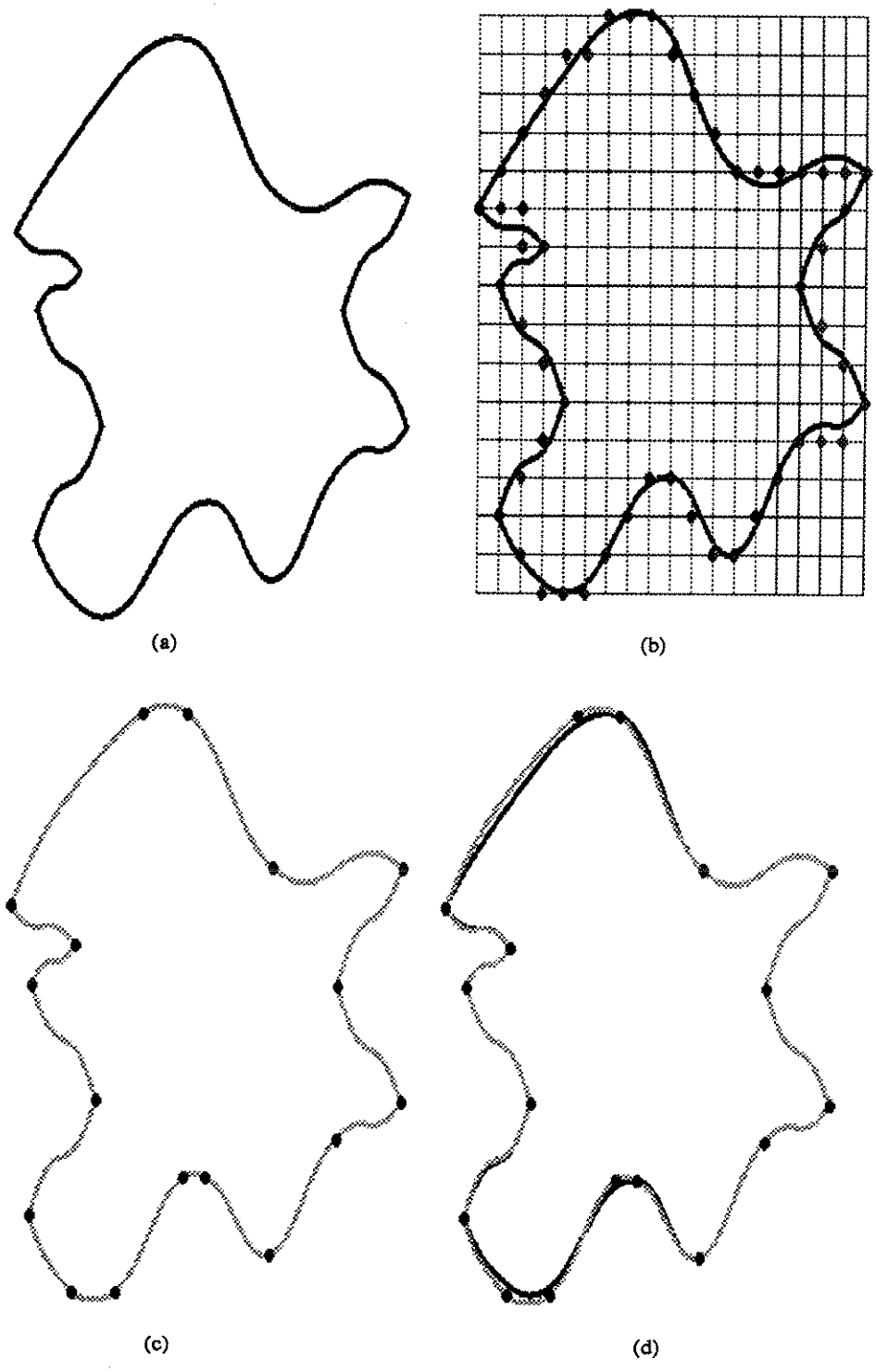


Figura 6.15: (a) Curva original; (b) Seqüenciamento dos pontos através da grade 15×20 (c) Curva interpolada a partir dos pontos significativos (d) Comparação da curva original e curva interpolada.

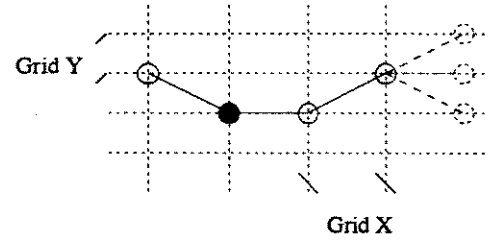


Figura 6.16: padrão do tipo I generalizado

Através da análise das tabelas 6.4 e 6.7, verifica-se que a rede numérica deixa de reconhecer algumas seqüências de entrada (cinco pontos subseqüentes), como padrões de pontos significativos. Isto se observa visto que, em todos os casos, o número de pontos significativos extraídos pela rede numérica é menor. Conforme mostrado na fig. 6.13, nota-se que a rede numérica é, neste caso, incapaz de reconhecer padrões do tipo I. A figura 6.16 traz uma representação mais geral deste tipo de padrão, decorrente da utilização de grades retangulares. A seqüência em destaque na figura 6.13 representa uma rotação do padrão ilustrado em 6.16 no sentido horário[1].

O fato de se obterem resultados diferentes para os padrões do tipo I pode ser entendido pela dificuldade de se treinarem situações de posição irrelevante em redes neurais artificiais. Estas situações exigem uma maior capacidade de abstração por parte da rede. Esta exigência se deve porque, além das diferenças de ângulos entre as situações treinadas e testadas, existe um grau de liberdade extra provocado pela posição irrelevante. A rede simbólica, por sua vez, não enfrenta este tipo de problema pois utiliza informações referentes apenas ao sinal das mudança de direção (positiva, negativa ou nula) e as posições irrelevantes não são testadas. Em todos os casos exemplificados nas tabelas 6.4 e 6.7, os erros da interpolação que passa pelos pontos obtidos pela rede simbólica são menores. Deste modo, demonstra-se a maior eficiência da rede simbólica no processo de extração de pontos significativos, quando se utilizam grades retangulares.

Um outro resultado a ser observado é a dependência da interpolação em relação à escolha da grade. As tabelas de 6.3 a 6.7 mostram que pequenas mudanças nos valores das grades podem provocar grandes alterações, tanto no número de pontos extraídos como nos erros de interpolação. Isto, porque as alterações na seqüência dos pontos definidos pela grade podem resultar em seqüências de entrada diferentes. Um exemplo destas diferenças pode ser observado na comparação das figuras 6.12 e 6.13.

[1]Os efeitos de rotação não alteram os resultados na saída das redes, visto que, tanto no caso simbólico quanto no numérico, analisam-se apenas as mudanças de direções.

6.3.2 Comparação do Conjunto de Pontos Extraídos pelos Métodos Simbólico e Numérico com um Conjunto de Pontos Ideal

Nesta seção serão mostrados os resultados dos processos de extração de pontos significativos para grades quadradas (10×10). Estes resultados serão comparados aos resultados obtidos por um sistema ideal de extração de pontos significativos. Este sistema, que define um conjunto de pontos para o qual a curva interpolada é igual à curva original, opera do seguinte modo:

- Define-se um conjunto P de pontos (x, y) quaisquer como sendo o conjunto de dados de entrada para o sistema de interpolação INNL.
- Obtém-se então, uma curva interpolada fechada passando pelos pares de coordenadas definidos por P.
- Esta curva interpolada passa a representar a curva original e o conjunto de pontos P define o conjunto ideal de pontos de modo a se obter, na etapa de interpolação, exatamente a curva original.

Um outro resultado a ser apresentado é a capacidade de compressão dos sistemas de extração de pontos significativos. A relação número de pontos extraídos em relação à quantidade total de pontos da curva original é comparada com a capacidade do sistema ideal de extração de pontos.

A figura 6.17(a) representa a curva original, onde os pontos assinalados (marcados por x) indicam os pontos pertencentes ao conjunto ideal P. A interpolação passando pelos pontos significativos obtidos pela rede simbólica é mostrado em 6.17(b), e comparada, em 6.17(c), à curva original. A figura 6.17(d) traz o resultado da extração dos pontos pela rede numérica e interpolação da curva fechada. Em 6.17(e), o resultado anterior é comparado à curva original.

Da mesma forma descrita anteriormente, as figuras 6.18 e 6.19 trazem, para diferentes curvas, os resultados dos processos de extração e interpolação, comparados aos resultados obtidos pelo sistema ideal.

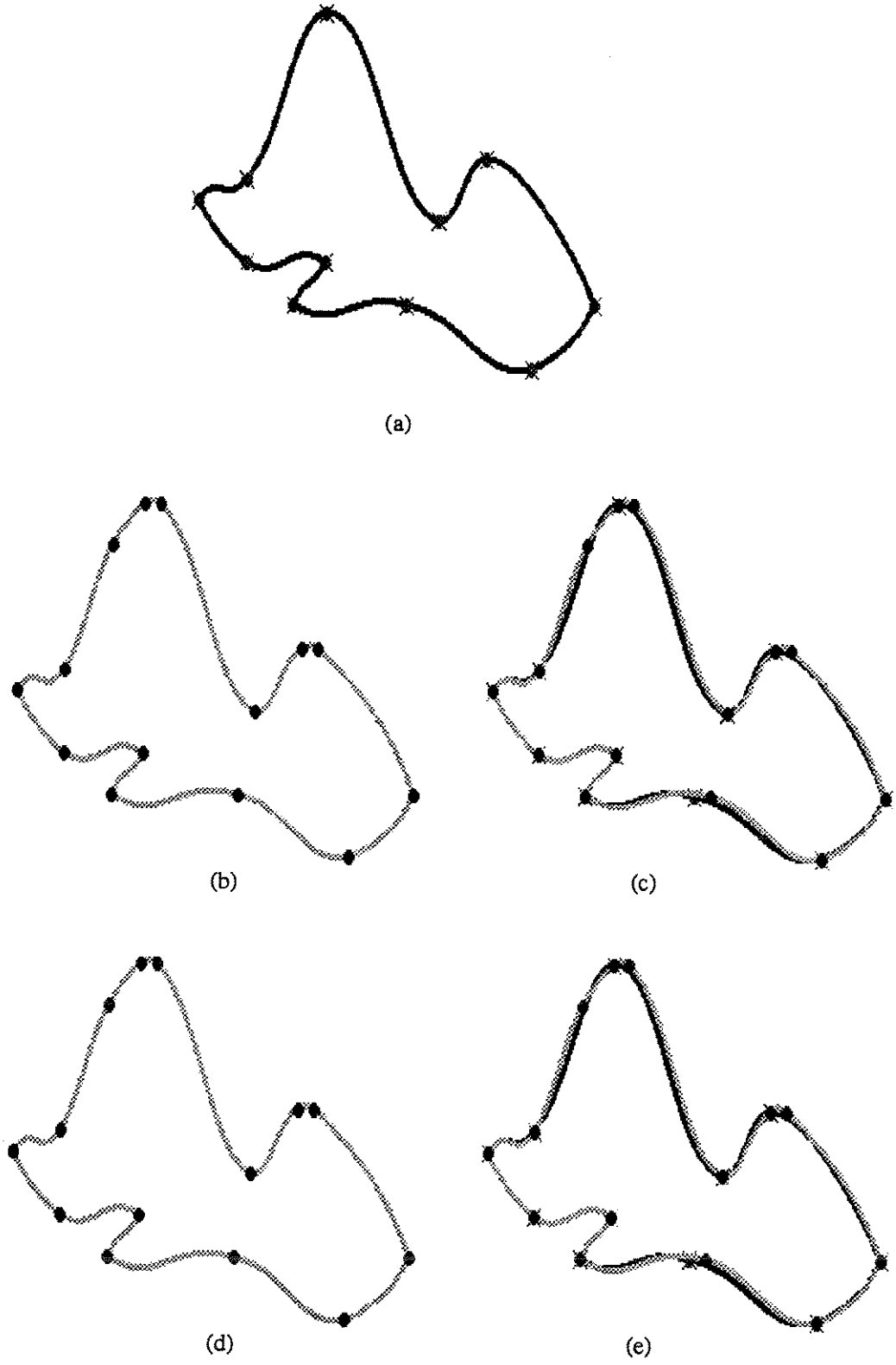


Figura 6.17: (a) Curva original; (b) curva interpolada e pontos significativos definidos por SNCD; (c) Comparação das curvas original e interpolada (d) curva interpolada e pontos significativos extraídos pela rede numérica. (e) comparação das curvas original e interpolada

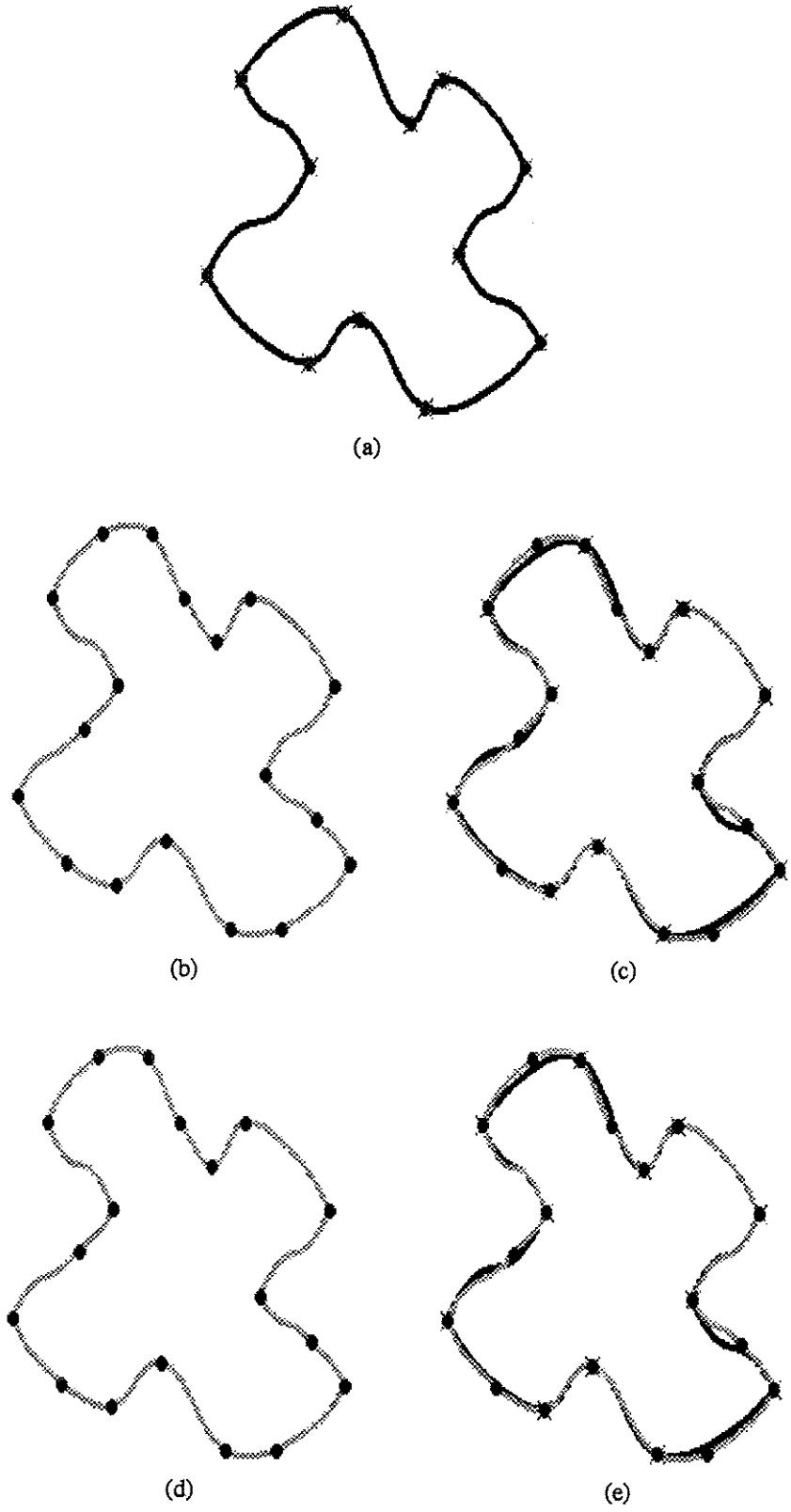


Figura 6.18: (a) Curva original; (b) curva interpolada e pontos significativos definidos por SNCD; (c) Comparação das curvas original e interpolada (d) curva interpolada e pontos significativos extraídos pela rede numérica. (e) comparação das curvas original e interpolada

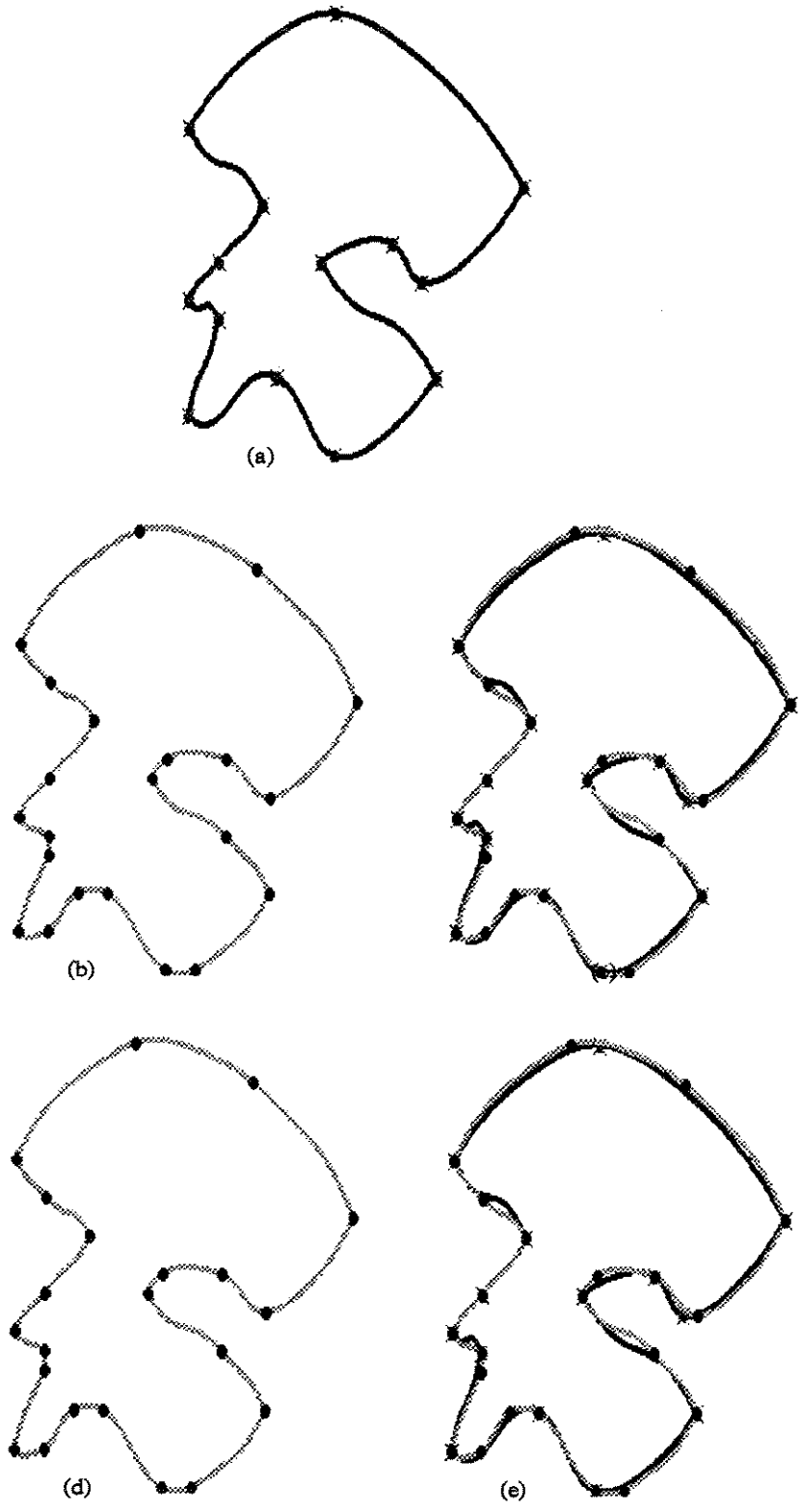


Figura 6.19: (a) Curva original; (b) curva interpolada e pontos significativos definidos por SNCD; (c) Comparação das curvas original e interpolada (d) curva interpolada e pontos significativos extraídos pela rede numérica. (e) comparação das curvas original e interpolada

A capacidade de compressão, que varia no intervalo $[0, 1]$, é definida do seguinte modo:

$$cc = \frac{n - ns}{n} = 1 - \frac{ns}{n},$$

onde n define o total de pontos da curva original e ns determina o número de pontos significativos obtidos. A tabela 6.8 ilustra os valores dos erros e capacidade de compressão para as figuras 6.17, 6.18 e 6.19.

Método	Curva	Grade $(X \times Y)$	Tot. de Pts. Signf.	Capacidade de Compressão	Erro Médio %	Erro Máximo %
Ideal	fig. 6.17	10×10	11	$1 - 11/859 = 0.987$	0	0
Simb.			14	$1 - 14/859 = 0.983$	0.70	4.09
Num.			14	$1 - 14/859 = 0.983$	0.70	4.09
Ideal	fig. 6.18	10×10	12	$1 - 12/914 = 0.986$	0	0
Simb.			18	$1 - 18/914 = 0.980$	0.51	3.82
Num.			18	$1 - 18/914 = 0.980$	0.51	3.82
Ideal	fig. 6.19	10×10	14	$1 - 14/1174 = 0.988$	0	0
Simb.			22	$1 - 22/1174 = 0.981$	0.79	3.03
Num.			22	$1 - 22/1174 = 0.981$	0.79	3.03

Tabela 6.8: Resultados com relação à capacidade de compressão cc.

A análise da tabela anterior comprova os resultados idênticos dos métodos simbólico e numérico para a situação de treinamento (grade quadrada com ângulos múltiplos de 45°).

Outro resultado importante refere-se à ótima capacidade de compressão dos dois métodos. Verifica-se que o total de pontos extraídos é bastante próximo do número mínimo (conjunto ideal) de pontos, necessários para se obter uma interpolação perfeita; e este total representa uma parte muito pequena comparada ao total de pontos da curva original.

Um último fato a ser salientado é a comprovação da eficiência do método INNL na interpolação de curvas fechadas. Isto pode ser observado, uma vez que, todos os resultados de interpolação das curvas na seção 6.3 foram obtidos pela alteração do método INNL proposta na seção 4.5.

Capítulo 7

Conclusão

O problema de se representar um sistema de entrada-saída por meio de uma base de dados pode tornar-se intratável, dependendo da dimensão desta base e dos recursos disponíveis. Neste sentido, a idéia de se implementar um sistema capaz de comprimir e posteriormente recuperar, com um erro pequeno, os dados de uma curva plana por exemplo, se apresenta como uma solução bastante interessante.

A utilização da teoria de redes neurais e lógica nebulosa, cujos fundamentos básicos foram descritos no capítulo 2, possibilitou a realização do processo de compressão através da extração dos pontos significativos e a simplificação do método de interpolação, utilizado na etapa de descompressão.

Em contrapartida à complexidade de alguns métodos tradicionais de interpolação, como visto no capítulo 3, a estruturação, por meio de regras nebulosas, do método proposto permitiu a obtenção de um algoritmo de interpolação mais simples, sem que isto representasse menor precisão. A modificação do algoritmo original INL através da introdução da não linearidade garantiu uma maior suavidade da curva interpolada. A prova da necessidade de funções de pertinência não lineares, mostrada no teorema 4.3.1, foi ilustrada por meio de simulações, onde os saltos na derivada das curva interpolada foram eliminados sem a necessidade de se recorrer aos dados da curva original. A outra alteração proposta permitiu a aplicação do método INNL à interpolação de curvas genéricas.

Comprovada a eficiência do método de interpolação INNL, a outra etapa do trabalho consistiu em se definir um sistema de determinação de pontos significativos de uma curva plana, de modo que esta curva pudesse ser recuperada através da interpolação passando por estes pontos. Após serem definidas as classes de padrões que representavam os pontos signi-

ficativos, foram propostas duas soluções para o processo de extração destes pontos. Tanto o modelo para processamento simbólico quanto o de processamento numérico receberam, como entrada, uma seqüência de pontos obtida pela sobreposição de uma grade $(X \times Y)$ à curva original. A situação de treinamento da rede numérica consistiu de combinações de ângulos de variação de direção nas seqüências de entrada, decorrentes da aplicação de grades (10×10). Verificou-se um comportamento idêntico para os dois sitemas nos casos das grades quadradas $(X = Y)$. Observou-se ainda que a rede simbólica apresentou melhores resultados para os casos não treinados, ou seja, grades retangulares.

Finalmente, comprovou-se a excelente capacidade de compressão dos métodos comparando-os aos resultados obtidos por um sistema ideal, onde um conjunto mínimo de pontos leva à interpolação com erro nulo.

Dentro da linha de pesquisa deste trabalho, são sugeridas como desenvolvimentos futuros as seguintes propostas:

- Realizar novas alterações que garantam a continuidade da segunda derivada da curva interpolada.
- Generalizar as bases de regras do método de interpolação e o processo de extração dos pontos significativos para aplicações em espaços N-dimensionais.
- Modificar o processo de extração de pontos significativos de forma que seja feita uma análise global, ao contrário da análise local feita para as seqüências de cinco pontos que definem os padrões de pontos significativos. Esta análise global poderia ser realizada, tendo como base o conhecimento do especialista que determinou a implementação do sistema ideal de extração de pontos significativos.
- Finalmente, aplicar o método de compressão e descompressão para problemas de reconhecimento de padrões.

Bibliografia

[ANW67] J. Ahlberg, E. Nilson, and J. Walsh. *The Theory of Splines and Their Applications.* Academic Press, 1967.

[Atk89] K. Atkinson. *An Introduction to Numerical Analysis.* Wiley & Sons, 1989.

[Bar89] A. G. Barto. Connectionist learning for control : An overview. COINS Technical Report 89-89, Department of Computer and Information Science // University of Massachusetts, Amherst MA 01003, September 1989.

[Boo78] C. de Boor. *A Practical Guide to Splines.* Springer-Verlag, 1978.

[GW87] R. C. Gonzales and P. Wintz. *Digital Image Processing.* Addison-Wesley, 1987.

[Hol57] J. C. Holladay. Smoothest curve approximation. *Mathematical Tables Aids Computation*, 11:233–243, 1957.

[Iiz90] *Proceedings of The International Conference on Fuzzy Logic and Neural Networks*, Iizuka - Japan, July 1990.

[IK66] E. Isaacson and H. Keller. *Analysis of Numerical Methods.* Wiley & Sons, 1966.

[Kau75] A. Kaufmann. *Introduction to the Theory of Fuzzy Subsets.* Academic Press, 1975.

[Lee90] C. C. Lee. Fuzzy logic in control systems : Fuzzy logic controller. *IEEE Transactions on systems, man, and cybernetics*, 20(2):404–435, March/April 1990. Parts I and II.

[Lip87] R. P. Lippmann. An introduction to conputin with neural nets. *IEEE ASSP Magazine*, 1:4–22, 1987.

[MP69] M. L. Minsky and S. A. Papert. *Perceptron : an Introduction to Computational Geometry.* MIT Press, 1969.

[Ped89] W. Pedrycz. *Fuzzy Control and Fyzzy Systems.* Springer-Verlag, 1989.

[Pre75] P. M. Prenter. *Splines and Variational Methods.* Wiley & Sons, 1975.

[RM86] D. E. Rumelhart and J. L. McClelland. *Parallel Distributed Processing : Explorations in the Microstructure of Cognition*, volume 1 : Foundations. Bradford Books/MIT Press, 1986.

[Roc92] A. F. Rocha. *NEURAL NETS - A Theory for Brains and Machines*, volume 638 of *Lecture Notes in Artificial Intelligence*. Springer-Verlag, 1992.

[RvZR93] M. Regattieri, F. J. von Zuben, and A. F. Rocha. Neurofuzzy interpolation II - reducing complexity of description. In *Proceedings of the 2th International Conference on Neural Networks and Fuzzy Logic*, pages 1835–1839, San Francisco - USA, March/April 1993. IEEE.

[Sok56] I. S. Sokolnikoff. *Mathematical Theory of Elasticity*. McGraw-Hill, 1956.

[Tak90] H. Takagi. Fusion technology of technology of fuzzy theory and neural networks - survey and future directions. In Iizuka [Iiz90], pages 13 – 26.

[TS83] T. Takagi and M. Sugeno. Derivation of fuzzy control rules from human operato's control actions. In *Proceedings of Symposium on Fuzzy Information Knowledge Representation and Decision Analysis*, pages 55–60, Marseilles- France, 1983.

[UYY90] E. Uchino, T. Yamakawa, and T. Yanaru. How to find out the supplementary rules representing an uncertain system. In Iizuka [Iiz90], pages 533 – 536.

[WL90] B. Widrow and Michael A. Lehr. 30 years of adaptive neural networks : Perceptron, madaline, and backpropagation. *Proceedings of the IEEE*, 78:1415–1442, 1990.

[Zad73] L. A. Zadeh. The concept of a linguistic variable and its application to approximate reasoning. *IEEE Transactions on systems, man, and cybernetics*, SMC-3:28–44, 1973.

[Zad75] L. A. Zadeh. Outline of a new approach to the analysis complex systems and decision processes. *Information Sciences*, 8:199 – 249, 1975.

[ZRR92] F. J. Zuben, M. Regattieri, and A. F. Rocha. Neurofuzzy interpolation: I - the theoretical background. In *Proceedings of The 2ND International Conference on Fuzzy Logic and Neural Networks*, pages 229 – 232, Iizuka - Japan, July 1992.