THIAGO FERNANDO FERREIRA

Sistemas p-Fuzzy Modificados para Dinâmicas Populacionais: modelagens e simulações

UNIVERSIDADE FEDERAL DE UBERLÂNDIA
FACULDADE DE MATEMÁTICA
2012

THIAGO FERNANDO FERREIRA

Sistemas p-Fuzzy Modificados para Dinâmicas Populacionais: modelagens e simulações

Dissertação apresentada ao Programa de Pós-Graduação em Matemática da Universidade Federal de Uberlândia, como parte dos requisitos para obtenção do título de **MESTRE EM MATEMÁTICA.**

Área de Concentração: Matemática.
Linha de Pesquisa: Análise Numérica.

Orientadora: Profa. Dra. Rosana Sueli da Motta Jafelice.

UBERLÂNDIA - MG
2012

Dados Internacionais de Catalogação na Publicação (CIP)
Sistema de Bibliotecas da UFU , MG, Brasil

F383s 2012

Ferreira, Thiago Fernando, 1983-
Sistemas p-Fuzzy modificados para dinâmicas populacionais : modelagens e simulações / Thiago Fernando Ferreira. - 2012.
138 f. : il.

Orientadora: Rosana Sueli da Motta Jafelice.

Dissertação (mestrado) – Universidade Federal de Uberlândia, Programa de Pós-Graduação em Matemática.
Inclui bibliografia.

1. Matemática - Teses. 2. Conjuntos difusos - Teses. 2. Equações diferenciais - Teses. 3. Equações diferenciais ordinárias - Teses. I. Jafelice, Rosana Sueli da Motta. II. Universidade Federal de Uberlândia. Programa de Pós-Graduação em Matemática. III. Título.

CDU: 51

UNIVERSIDADE FEDERAL DE UBERLÂNDIA
FACULDADE DE MATEMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM MATEMÁTICA
Av. João Naves de Ávila, 2121, Bloco 1F, Sala 1F 152
Campus Santa Mônica, Uberlândia - MG, CEP 38400-902

ALUNO: Thiago Fernando Ferreira.

NÚMERO DE MATRÍCULA: 11012MAT003.

ÁREA DE CONCENTRAÇÃO: Matemática.

LINHA DE PESQUISA: Análise Numérica.

PÓS-GRADUAÇÃO EM MATEMÁTICA: Nível Mestrado.

TÍTULO DA DISSERTAÇÃO: Sistemas p-Fuzzy Modificados para Dinâmicas Populacionais: modelagens e simulações.

ORIENTADORA: Profa. Dra. Rosana Sueli da Motta Jafelice.

Esta dissertação foi **APROVADA** em reunião pública realizada na Sala Multiuso da Faculdade de Matemática, Bloco 1F, Campus Santa Mônica, em 20 de abril de 2012, às 14h00min, pela seguinte Banca Examinadora:

NOME	ASSINATURA
Profa. Dra. Rosana Sueli da Motta Jafelice UFU - Universidade Federal de Uberlândia	Rosana Sueli da Motta Jafelice
Prof. Dr. Laécio Carvalho de Barros UNICAMP - Universidade Estadual de Campinas	Laécio C. Barros
Profa. Dra. Sezimária de Fátima Pereira Saramago UFU - Universidade Federal de Uberlândia	Saramago

Uberlândia-MG, 20 de abril de 2012.

Dedicatória

À minha mãe, Miriam, pelo amor e exclusividade que tenho em sua vida.

À minha filha, Beatriz, que é o maior motivo de todas as minhas vitórias, e a quem dedico este trabalho, meus dias, meus sonhos, e minha vida.

Agradecimentos

Agradeço:

- Em primeiro lugar, a Deus.
- Aos amigos da UNESP - Universidade Estadual Paulista "Júlio de Mesquita Filho- Câmpus de Ilha Solteira, com quem dividi minhas primeiras experiências acadêmicas, em especial a Profa. Dra. Roseli Arbach Fernandes de Oliveira e Prof. Dr. Luis Antonio Fernandes de Oliveira, por todos os conselhos, desde o primeiro ano da minha Graduação.
- Aos amigos de mestrado Giselle, Franciele, Adenilce, Fabrícia, Fabiana, e em especial ao meu grande amigo Edir, pessoas que muito contribuiram para meu crescimento acadêmico e pessoal.
- A todos os membros da FAMAT por toda ajuda e acompanhamento, do início ao fim, professores e fucionários que se mostraram grandes amigos, conselheiros e cúmplices em minha caminhada.
- Aos professores Laécio Carvalho de Barros e Sezimária de Fátima Pereira Saramago, por terem aceito o convite de participar da banca examinadora.
- Aos docentes do Programa de Mestrado em Matemática da UFU, pela dedicação no esclarecimento de todas as dúvidas, em especial ao coordenador Prof. Dr. Edson Agustini.
- De um modo muito especial, quero agradecer a minha orientadora, a Profa. Dra. Rosana Sueli da Motta Jafelice, pela paciência e conhecimento transmitidos, e pela confiança e empenho depositadas na realização deste trabalho, não medindo esforços para sanar todas as dúvidas e direcionar de maneira exímia meus estudos.
- A agência financiadora CAPES, pelo apoio financeiro indispensável para desenvolvimento e conclusão do trabalho, bem como sustento não só meu, como o da minha família.
- A meu pai João Célio, por acreditar em mim e me apoiar, e a minha avó Flora pelo carinho e afeto.
- A pessoas especiais como minha "madrinha"tia Artêmis, tio Amilton, tia Priscila, tio Maurício, tia Cássia, tia Mara, Tia Cirene, Tia Damares, que estiveram o tempo todo ao meu lado, mesmo não podendo segurar em minha mão.
- Aos amigos da UFTM, Profa. Marcela Vilela e Prof. Osmar Aléssio, pelo incentivo e apoio.
- A Aline Carrijo pela disponibilidade e colaboração na fase final da pesquisa.
- A Thiago de Freitas C. Costa, por estar ao meu lado, me apoiar incondicionalmente, e mostrar que juntos somos mais fortes.

Muito Obrigado!

FERREIRA, Thiago F., *Sistemas p-Fuzzy Modificados para Dinâmicas Populacionais: modelagens e simulações.* 2012. 138 p., Dissertação de Mestrado, Universidade Federal de Uberlândia, Uberlândia-MG.

Resumo

O grande desafio da modelagem matemática é descrever fenômenos naturais para fazer previsões e tomar decisões. O uso de equações de diferenças e equações diferenciais ordinárias tem sido ferramenta essencial na modelagem de fenômenos populacionais. A viabilidade e aplicabilidade de sistemas parcialmente fuzzy (p-fuzzy) na modelagem de fenômenos descritos por equações diferenciais já foi estudada anteriormente. A proposta deste trabalho é apresentar sistemas p-fuzzy modificados (cujas funções de pertinência são alteradas por uma potência) e sistemas p-fuzzy modificados no tempo (em que as funções de pertinência são alteradas a cada iteração, no tempo) que descrevem uma melhor aproximação às soluções determinísticas para alguns modelos de dinâmica populacional, do que os sistemas p-fuzzy sem modificações. Os sistemas p-fuzzy modificados foram obtidos numericamente por meio de programas no Matlab para determinar a potência e a função que altera as potências adequadamente, calculando o menor erro entre a solução analítica e os sistemas p-fuzzy modificados para alguns modelos de dinâmica populacional. Constatamos que o sistema p-fuzzy modificado no tempo para o modelo do controle de pragas é mais conveniente em relação ao sistema p-fuzzy sem modificações; e modificado por potência, pois neste sistema o resultado final é a menor população de pragas sobrevivente ao controle. Analisando os resultados alcançados, pode-se concluir que com a utilização de sistemas p-fuzzy modificados por potência, com modificações tanto restritivas (potência > 1) como expansivas ($0 <$ potência < 1), obtemos uma melhor aproximação à solução analítica de cada modelo. Para o modelo do controle de pragas essa aproximação torna-se ainda maior quando a modificação é feita a cada iteração, ou seja, com sistemas p-fuzzy modificados no tempo.

Palavras-chave: Teoria dos Conjuntos Fuzzy, Equações de Diferenças, Equações Diferenciais Ordinárias, Sistemas p-Fuzzy Modificados, Sistemas p-Fuzzy Modificados no Tempo.

FERREIRA, Thiago F., *Modified p-Fuzzy Systems for Population Dynamics: modeling and simulation.* 2012. 138 p. M. Sc. Dissertation, Federal University of Uberlândia, Uberlândia-MG.

Abstract

The great challenge of mathematical modeling is to describe natural phenomena in order to make predictions and take decisions. The use of difference equations and ordinary differential equations has been an essential tool in modeling population-related phenomena. The feasibility and applicability of partially fuzzy systems (p-fuzzy) in modeling phenomena described by differential equations have already been studied. The proposal in this work is to present modified p-fuzzy systems (which membership functions are altered by a potency) and in-time modified p-fuzzy systems (in which the membership functions are altered at each iteration in time) which describe a better approximation to the deterministic solutions for some dynamic population models, than the p-fuzzy systems without modifications. The modified p-fuzzy systems were obtained numerically by means of programs in Matlab software, in order to determine the potency and function that adequately alter the potency, calculating the smallest error between the analytic solution and the modified p-fuzzy systems for some models in population dynamics. We verified that in-time modified p-fuzzy system for the pest control model is more convenient in relation to both the p-fuzzy without modifications and the one modified by the potency, since in this system the final result is the smallest pest population that survives to the control. After analyzing the attained results, it can be concluded that when the potency modified p-fuzzy system with either restrictive modifications (potency > 1) or expansive ($0 <$ potency < 1), we accomplish a better approximation for the analytical solution for each model. For the pest control model, this approximation becomes greater when the modification is made to each and a very iteration, i.e., with in-time modified p-fuzzy system.

Keywords: Fuzzy Set Theory, Difference Equations, Ordinary Differential Equations, Modified p-Fuzzy System, In-time Modified p-Fuzzy Systems.

Lista de Figuras

Sumário

Introdução

A *teoria dos conjuntos fuzzy* foi apresentada em 1965 por Lofti A. Zadeh [27], professor do departamento de engenharia elétrica e ciências da computação da Universidade da Califórnia, em Berkeley, quando ele trabalhava com problemas sobre a classficação de conjuntos que não possuíam fronteiras bem definidas.

Em muitos problemas de física e matemática não temos dificuldade em classificar elementos como pertencentes ou não a um dado conjunto clássico, ou seja, dado um conjunto A e um elemento x do conjunto universo U conseguimos dizer: se $x \in A$ ou se $x \notin A$. Porém, existem inúmeras situações em que a relação de pertinência não é bem definida e, nestes casos, não sabemos dizer se o elemento pertence ou não a um dado conjunto. A intenção de Zadeh foi flexibilizar a pertinência de elementos aos conjuntos criando a ideia de *grau de pertinência* e, dessa forma, um elemento poderia pertencer parcialmente a um dado conjunto. Esta ideia foi publicada em 1965, representando o marco do nascimento da teoria dos conjuntos fuzzy [27].

Apesar da forte resistência à teoria dos conjuntos fuzzy, muitos pesquisadores vislumbraram as possibilidades que esta teoria oferecia e Zadeh encontrou seguidores em todo mundo, principalmente no Japão, onde esta teoria encontrou um solo fértil para desenvolver-se rapidamente. Em 1976 temos a primeira aplicação industrial da teoria dos conjuntos fuzzy, desenvolvido pelo Circle Cement e SIRA, na Dinamarca, constituída de um Sistema Baseado em Regras Fuzzy incorpora o conhecimento e a experiência dos operários para controlar os fornos das fábricas. Podemos notar um crescente interesse pela teoria fuzzy por profissionais e pesquisadores das mais diversas áreas, dado a capacidade de explorar variáveis linguísticas, da possibilidade de desenvolver raciocínio mais próximo ao humano, da diversidade de operações e da sua potencialidade em aplicações [17].

A característica essencial da modelagem matemática de processos variacionais, utilizando equações determinísticas é a exatidão obtida nas previsões do fenômeno estudado. Nos modelos

clássicos da biomatemática, particularmente nos modelos de dinâmica populacionais que lidam com incertezas, as soluções são processos estocásticos cujas médias podem ser obtidas quando se tem as densidades de distribuição das variáveis e/ou dos parâmetros envolvidos no modelo referente ao fenômeno analisado. Porém, se numa população, além de quantificar seus elementos, pretendemos levar em conta certas qualidades dos indivíduos, as variáveis devem captar tais incertezas [2].

Em diversas áreas de ciências aplicadas podemos encontrar modelagens com a teoria dos conjuntos fuzzy. Em particular, Cecconelo [5], Peixoto [18] e Silva [22] utilizaram um Sistema Baseado em Regras Fuzzy (SBRF) para modelar a densidade populacional de uma espécie, utilizando regras de inferência do tipo "Se ... então ...", sendo que [18] foi o primeiro trabalho que apresentou os sistemas p-fuzzy. Nos trabalhos citados anteriormente, as equações diferenciais ordinárias são substituídas por um Sistema Baseado em Regras Fuzzy (SBRF) no qual a relação que descreve as variações com as suas respectivas variáveis de estado é descrita por uma base de regras fuzzy no lugar de equações. Esses tipos de sistemas são chamados de Parcialmente Fuzzy (p-fuzzy) e foram construídos usando o método de inferência de Mamdani [16].

Em [7] e [14] foram feitos estudos sobre aplicação de sistemas p-fuzzy modelando equações diferenciais parciais, sendo que no primeiro foi utilizado o método de inferência de Takagi-Sugeno-Kang e no segundo o método de inferência de Mamdani. Vale ressaltar que em [7] os sistemas p-fuzzy foram obtidos utilizando a função *ANFIS (Adaptive Neuro-Fuzzy Inference Systems)* do Matlab, que a partir de um conjunto de dados identifica as funções de pertinência e os parâmetros do sistema baseado em regras fuzzy. As diferenças básicas entre o método de inferência de Takagi-Sugeno-Kang (TSK) e o método de Mamdani estão na forma de escrever o consequente de cada regra e no procedimento de defuzzificação para obter-se a saída geral do sistema [2]. Com o método TSK, o consequente de cada regra é dado explicitamente por uma função dos valores de entrada desta regra. Em relação ao método de Mamdani, a descrição encontra-se neste trabalho.

Os modelos variacionais fuzzy podem comportar vários tipos de incertezas, traduzidas por coeficientes, condições iniciais ou pelas próprias variáveis de estado. Aplicamos a teoria dos conjuntos fuzzy para modelos de dinâmicas populacionais, considerando as variáveis de estado e as variações em termos linguísticos [2].

O objetivo desta dissertação é estudar os sistemas p-fuzzy, com aplicação da teoria dos modificadores linguísticos. Como o próprio nome sugere, os modificadores linguísticos são frequentemente utilizados para alterar atributos, ou seja, modelar advérbios. As funções que determinam o grau de pertinência dos elementos aos subconjuntos fuzzy serão alteradas por modificadores do tipo potência que elevam a função de pertinência a potências diferentes de um. Se o valor da potência está entre zero e um, o modificador é chamado expansivo e se o valor for maior do que um, é chamado restritivo. Dessa forma, trabalhamos com sistemas p-fuzzy modificados e verificamos se existe vantagem na utilização destes, em relação aos sistemas p-fuzzy sem modificações. Finalmente, verificamos a possibilidade de aplicação da teoria dos modificadores, a cada iteração de alguns modelos de fenômenos biológicos. Esse sistema construído é chamado sistema p-fuzzy modificado no tempo, tema não encontrado na literatura conhecida. Verificamos que existe vantagem na utilização dos sistemas p-fuzzy modificados e p-fuzzy modificados no tempo, modelando fenômenos biológicos sem a utilização de equações de diferenças e equações diferenciais ordinárias.

O trabalho está dividido em quatro capítulos, considerações finais, referências bibliográficas e Apêndices A e B.

No Capítulo 1, apresentamos uma breve introdução à Teoria dos Conjuntos Fuzzy, onde constam os conceitos básicos para desenvolvimento do nosso trabalho.

O Capítulo 2 descreve métodos numéricos para a resolução de equações diferenciais ordinárias e métodos de integração numérica utilizada nos capítulos seguintes. No Capítulo 3 introduzimos os conceitos a respeito do nosso objeto de estudo, os sistemas p-fuzzy, que são essenciais para o desenvolvimento do capítulo subsequente.

O Capítulo 4 é o mais importante deste trabalho, pois nele serão discutidos os sistemas p-fuzzy modificados e modificados no tempo de alguns modelos de dinâmica populacional, onde os resultados são apresentados, bem como os gráficos referentes aos resultados alcançados. É neste Capítulo que são descritas as metodologias utilizadas para a busca da melhor potência para o sistema p-fuzzy modificado e da função de alteração de potências para o sistema p-fuzzy modificado no tempo.

No Capítulo 5 apresentamos as considerações finais sobre os estudos realizados e a discussão sobre a vantagem da utilização dos sistemas p-fuzzy modificados e modificados no tempo para dinâmicas populacionais, em relação a sistemas p-fuzzy sem modificações. Além disso, elencamos sugestões para possíveis trabalhos futuros.

No Apêndice A estão os programas e rotinas computacionais das funções de pertinência que foram modificadas a partir dos códigos fonte no toolbox *fuzzy* do programa *Matlab* e utilizadas neste trabalho. No Apêndice B os programas e rotinas computacionais de cada um dos sistemas p-fuzzy descritos para os modelos computacionais dos Capítulos 3 e 4.

Thiago Fernando Ferreira

Uberlândia-MG, 20 de abril de 2012.

Capítulo 1

Conjuntos Fuzzy

Serão apresentados neste capítulo conceitos e ferramentas básicas, com alguns exemplos retirados de [2], da Teoria dos Conjuntos Fuzzy, utilizadas no decorrer deste trabalho.

1.1 Conjuntos Fuzzy

Um subconjunto fuzzy F do universo U é definido em termos de uma função de pertinência μ que, a cada elemento $x \in U$, associa um número $\mu(x)$, entre zero e um, chamado de grau de pertinência de x a F.

Definição 1.1 *Seja U um conjunto e C um subconjunto de U. A função característica de C é dada por*

$$\chi_C(x) = \begin{cases} 1 & se \quad x \in C \\ 0 & se \quad x \notin C \end{cases} \tag{1.1}$$

Então χ_C é uma função cujo domínio é U e sua imagem está contida no conjunto $\{0,1\}$, com $\chi_C(x) = 1$, indicando que o elemento x está em C, enquanto $\chi_C(x) = 0$ aponta que x não é um elemento de C. Logo, a função característica descreve completamente o conjunto C, pois esta função mostra quais elementos do conjunto universo U são elementos também de C. Entretanto, existem casos em que a pertinência entre elementos e conjuntos não é dada de forma precisa. O que é possível é dizer qual elemento do conjunto enquadra-se "melhor"à regra que caracteriza o subconjunto.

Por exemplo, consideremos o subconjunto dos números naturais pequenos [2].

$$C = \{n \in \mathbb{N} : n \text{ é pequeno}\}. \tag{1.2}$$

A pergunta que devemos responder é: O número 0 pertence a C? E o número 999?

Esta resposta é incerta, mas no contexto da teoria dos conjuntos fuzzy poderíamos dizer que ambos pertencem a C. A única afirmação razoável, nesse caso, é que 0 é um número menor do que 999, portanto, pertence a C com grau de pertinência maior.

A função de pertinência associada a C deve ser "construída"de forma coerente com o termo "pequeno". Uma possibilidade para a função de pertinência de C é

$$\varphi_C(n) = \frac{1}{n+1} \tag{1.3}$$

onde $n \in \mathbb{N}$. Dessa forma poderíamos dizer que o número 0 pertence a C com grau de pertinência $\varphi_C(0) = 1$, enquanto que 999 pertence também a C, porém, com grau de pertinência $\varphi_C(999) = 0,001$. Veja na Figura 1.1 a representação da função de pertinência φ_C.

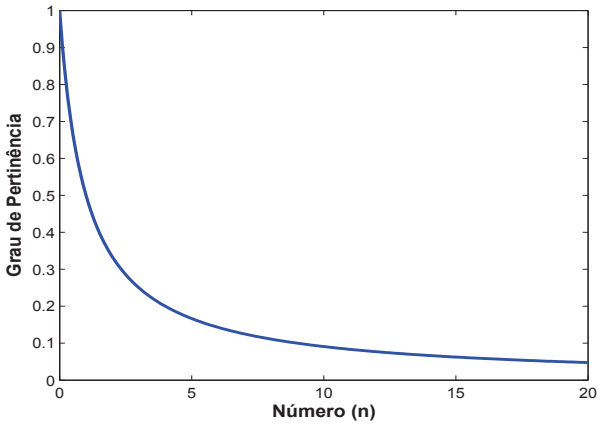

Figura 1.1: Representação dos números pequenos.

A seguir formalizaremos alguns conceitos da Teoria dos Conjuntos Fuzzy que serão utilizados neste trabalho, começando com o conceito de *subconjunto fuzzy*.

Definição 1.2 *Seja U um conjunto (que chamaremos de* conjunto fuzzy clássico*), um subconjunto fuzzy F de U é caracterizado por uma função*

$$\varphi_F : U \to [0, 1].$$

Esta função é chamada função de pertinência do subconjunto fuzzy F. O valor da função de pertinência é interpretado como o grau em que o elemento x de U tem a propriedade F. O índice F na função de pertinência é usado em analogia à função característica de subconjunto clássico, conforme a Definição 1.1.

O valor $\varphi_F \in [0,1]$ indica o grau de pertinência com que o elemento x de U está no conjunto F; $\varphi_F(x) = 0$ e $\varphi_F(x) = 1$ indicam, respectivamente, a não pertinência e a pertinência plena de x ao conjunto fuzzy F.

Um subconjunto fuzzy F é determinado pelo conjunto de pares ordenados:

$$F = \{(x, \varphi_F(x)) : x \in U\}.$$

Pode-se dizer, em caráter formal, que a definição de subconjunto *fuzzy* foi obtida simplesmente ampliando-se o contra-domínio da função característica que é o conjunto {0,1}, para o intervalo [0,1]. Portanto, um conjunto clássico é um caso particular de um dado conjunto *fuzzy*, cuja função de pertinência φ_F é uma função característica χ_F.

Exemplo 1.1 (Números próximos a 5). *Considere o subconjunto F dos números reais próximo a 5:*

$$F = \{x \in \mathbb{R} : x \text{ é próximo de } 5\}$$

Se definirmos a função $\varphi_F : \mathbb{R} \to [0,1]$, que associa a cada x real o valor de proximidade ao ponto 5, pela expressão

$$\varphi_F(x) = \begin{cases} (1 - |x-5|) & \text{se } 4 \leq x \leq 6 \\ 0 & \text{se } x \notin [4,6] \end{cases}, \tag{1.4}$$

então o subconjunto F dos pontos *próximos* de 5, caracterizado por φ_F, é tal que $\varphi_F(5,001) = 0,99$ e $\varphi_F(10) = 0$. Neste caso, dizemos que $x = 5,001$ é um ponto próximo de 5 com grau de proximidade 0,99 e $x = 10$ não é próximo de 5. Tal conjunto pode ser representado graficamente pela Figura 1.2.

A caracterização de proximidade é subjetiva e depende da função de pertinência que pode ser dada de uma infinidade de maneiras, dependendo de como se quer avaliar o termo "próximo".

Note que *próximo* de 5 significa estar numa vizinhança pré-determinada de 5. A subjetividade está exatamente na escolha do raio da vizinhança.

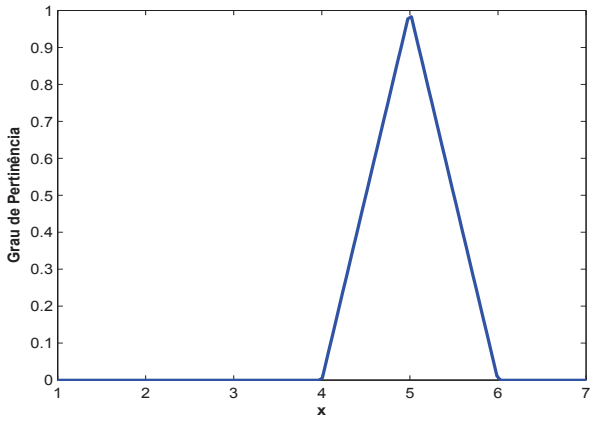

Figura 1.2: Representação dos números próximos a 5.

Exemplo 1.2 *Seja o conjunto das "pessoas de meia idade", os elementos são as idades, e os graus de pertinência dependeriam da idade. Uma representação para tal conjunto pode ser dada por:*

$$meia\ idade = \{(50, 1.0); (48, 0.95); (45, 0.9); (30, 0.2); (25, 0.004); (20, 0); (60, 0.8)\}$$

Uma representação gráfica para tal conjunto pode ser dada na Figura 1.3.

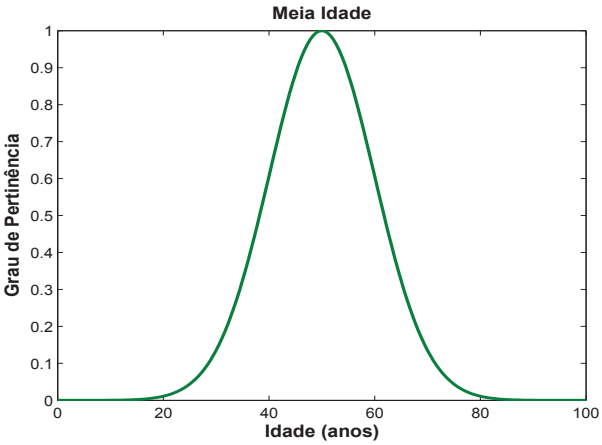

Figura 1.3: Representação das pessoas de meia idade.

Podemos ter também uma notação, proposta por Zadeh, mais conveniente para os conjuntos fuzzy, usando o sinal de "+"para denotar uma enuneração [27]. Utilizamos a barra "/"como uma notação alternativa para o par ordenado $(x, \varphi_F(x))$ o conjunto fuzzy pode ser escrito como segue:

$$F = \varphi_F(x_1)/x_1 + \varphi_F(x_2)/x_2 + \ldots + \varphi_F(x_n)/x_n. \tag{1.5}$$

Dessa forma, quando o conjunto universo U é finito, pode-se representar o conjunto fuzzy F por

$$F = \sum \varphi_F(x)/x. \tag{1.6}$$

Definição 1.3 *Podemos considerar somente elementos do universo cujo grau de pertinência seja diferente de zero. O subconjunto clássico de U definido por*

$$suppF = \{x \in U : \varphi_F(x) > 0\} \tag{1.7}$$

é denominado conjunto suporte *e tem papel fundamental na relação entre as teorias clássicas e a teoria fuzzy [27].*

1.2 Operações entre Conjuntos Fuzzy

Temos que dois conjuntos fuzzy são iguais se todos os elementos do universo têm o mesmo grau de pertinência em cada um deles.

Nesta seção estudaremos as operações típicas de conjuntos como união, intersecção e complementação.

Dizemos que um conjunto fuzzy A é subconjunto de um conjunto fuzzy B e escrevemos $A \subset B$, se $\varphi_A \leq \varphi_B$ para todo $x \in U$, ou seja, todo elemento do universo tem grau de pertinência no conjunto A menor que no conjunto B.

Temos ainda que a função de pertinência do conjunto vazio ($\emptyset$) é dada por $\varphi_\emptyset(x) = 0$, enquanto que o conjunto universo U tem função de pertinência $\varphi_U(x) = 1$, para todo $x \in U$.

Sejam A e B subconjuntos clássicos de U. Considere os conjuntos

$$A \cup B = \{x \in U; \quad x \in A \quad \texttt{ou} \quad x \in B\},$$

$$A \cap B = \{x \in U; \quad x \in A \quad \texttt{e} \quad x \in B\},$$

$$A' = \{x \in U; \quad x \notin A\}.$$

onde A e B são subconjuntos clássicos de U representados, respectivamente, por suas funções características χ_A e χ_B.

Os conjuntos $A \cup B$ (união), $A \cap B$ (intersecção) e A' (complementar) têm para qualquer $x \in U$, respectivamente, as seguintes funções características:

$$\chi_{A \cup B}(x) = max\{\chi_A(x), \chi_B(x)\},$$

$$\chi_{A \cap B}(x) = min\{\chi_A(x), \chi_B(x)\},$$

$$\chi_{A'}(x) = 1 - \chi_A(x).$$

Estendemos as operações de união, intersecção e complemento, generalizando as funções características que definem os conjuntos, para as respectivas funções de pertinência.

Definição 1.4 *Sejam A e B subconjuntos fuzzy de U representados, respectivamente, pelas funções de pertinência φ_A e φ_B. Temos que os conjuntos fuzzy $A \cup B$, $A \cap B$ e A' têm para qualquer $x \in U$ as respectivas funções de pertinência:*

$$\varphi_{A \cup B}(x) = max\{\varphi_A(x), \varphi_B(x)\},$$

$$\varphi_{A \cap B}(x) = min\{\varphi_A(x), \varphi_B(x)\},$$

$$\varphi_{A'}(x) = 1 - \varphi_A(x).$$

As Figuras 1.4, 1.5, 1.6 e 1.7 representam, respectivamente, os conjuntos fuzzy A e B, $A \cup B$, $A \cap B$ e A'. Se A e B forem conjuntos clássicos, então as funções características das operações são iguais as funções de pertinência definidas, o que mostra a coerência dessas definições.

Temos também que no caso de conjuntos fuzzy um elemento pode pertencer a um conjunto e ao mesmo tempo, ao seu complementar, e esta é a grande diferença entre a teoria clássica de conjuntos e a teoria dos conjuntos fuzzy, pois na teoria clássica, um elemento pertence ao conjunto ou ao seu complementar.

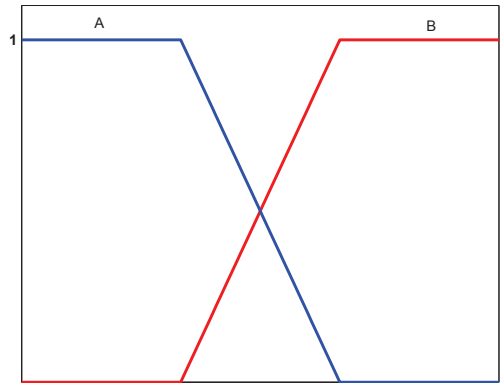

Figura 1.4: Conjuntos Fuzzy A e B.

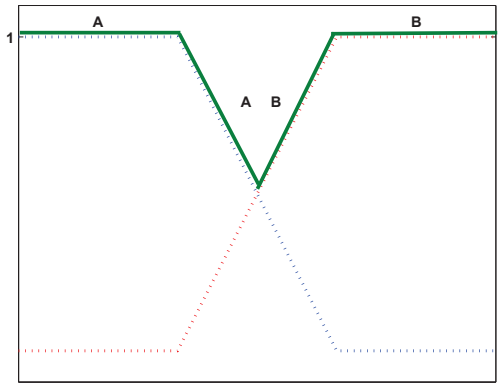

Figura 1.5: União entre A e B.

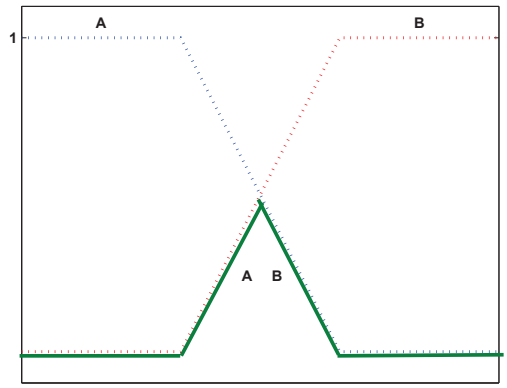

Figura 1.6: Intersecção entre A e B.

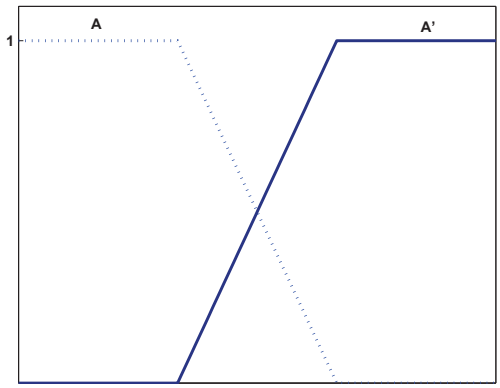

Figura 1.7: Complementar de A.

Exemplo 1.3 *Seja U um conjunto universo composto por pacientes de uma clínica, identificados pelos números $1, 2, 3, 4$ e 5. Sejam A e B os conjuntos fuzzy que representam os pacientes com febre e tosse seca, respectivamente. A Tabela 1.1 dada a seguir contém a união $A \cup B$, a intersecção $A \cap B$, o complemento A' e a intersecção $A \cap A'$. Observe que, diferentemente dos conjuntos clássicos, $A \cap A' \neq 0$, de acordo com a Definição 1.4.*

Paciente	Febre $U(A)$	Tosse Seca $U(B)$	$U(A \cup B)$	$U(A \cap B)$	$U(A')$	$U(A \cap A')$
1	0.7	1.0	1.0	0.7	0.3	0.3
2	0.5	0.5	0.5	0.5	0.5	0.5
3	0.9	0.0	0.9	0.0	0.1	0.1
4	0.4	0.5	0.5	0.4	0.6	0.4
5	1.0	0.2	1.0	0.2	0.0	0.0

Tabela 1.1: União, intersecção e complementar dos conjuntos A e B.

Mais exemplos com o tema diagnóstico médico e doenças pode ser encontrado em [25].

1.3 Níveis de um Conjunto Fuzzy

Caracterizamos um subconjunto fuzzy F de U por elementos do universo com seus respectivos graus de pertinência.

Dizemos que um elemento x de U pertence a uma classe se o seu grau de pertinência é maior do que determinado valor ou nível $\alpha \in [0, 1]$ que define a classe. O conjunto *crisp* desses elementos é chamado de α-nível de F e é denotado por $[F]^\alpha$ [27].

Definição 1.5 *Seja F um subconjunto fuzzy U e $\alpha \in [0, 1]$. O α-nível de F é o subconjunto U definido por:*

$$[F]^\alpha = \{x \in U : \varphi_F(x) \geq \alpha\}, \tag{1.8}$$

com $0 < \alpha \leq 1$ e além disso $[F]^0 = \overline{suppF}$.

Segue abaixo um exemplo sobre α-nível.

Exemplo 1.4 *Considere $U = \mathbb{R}$ e B um conjunto fuzzy de $\mathbb{R}$ dado pela função de pertinência a seguir:*

$$\varphi_B(x) = \begin{cases} x - 4 & se\ 4 \leq x < 5 \\ 6 - x & se\ 5 \leq x < 6 \\ 0 & se\ x \notin [4, 6] \end{cases} .$$

Temos então que $\overline{suppB} =]4, 6[$, portanto $[B]^0 = \overline{]4, 6[} = [4, 6]$.
Determinamos o conjunto α-nível, com $0 < \alpha \leq 1$, considerando $x - 4 \geq \alpha$ e $6 - x \geq \alpha$, então $[B]^\alpha = [4 + \alpha, 6 - \alpha]$. A Figura 1.8 está representando $[B]^\alpha$ no eixo x, para um determinado valor de α, em que $0 < \alpha \leq 1$.

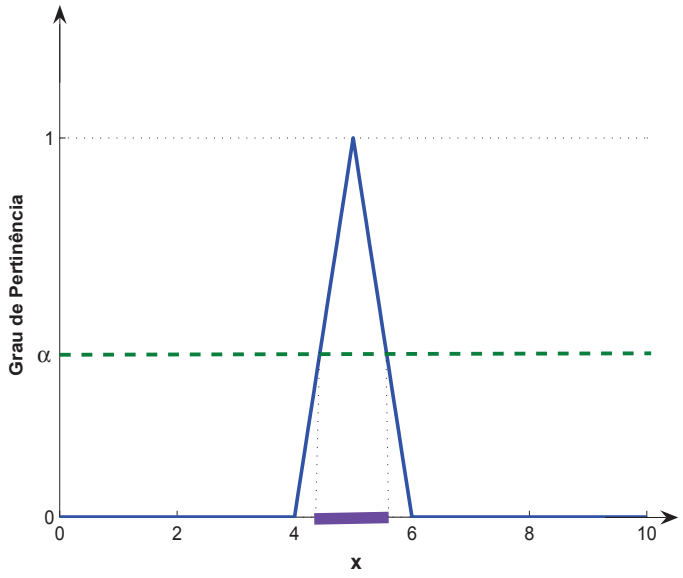

Figura 1.8: α-nível de B.

O teorema seguinte mostra que a família de conjuntos clássicos $[A]^\alpha$ determina completamente o conjunto fuzzy A.

Teorema 1.1 *Sejam A e B subconjuntos fuzzy de U. Uma condição necessária para que $A = B$ é que $[A]^\alpha = [B]^\alpha$, para todo $\alpha \in [0, 1]$ [2].*

Demonstração: *É claro que $A = B \Longrightarrow [A]^\alpha = [B]^\alpha$ para todo $\alpha \in [0, 1]$. Suponhamos agora que $[A]^\alpha = [B]^\alpha$ para todo $\alpha \in [0, 1]$. Se $A \neq B$ então $x \in U$ tal que $\varphi_A(x) \neq \varphi_B(x)$. Logo, temos que $\varphi_A(x) < \varphi_B(x)$ ou $\varphi_A(x) > \varphi_B(x)$. Supondo $\varphi_A(x) > \varphi_B(x)$, podemos*

concluir que $x \in [A]^{\varphi_A(x)}$ *e* $x \notin [B]^{\varphi_A(x)}$ *e, portanto* $[A]^{\varphi_A(x)} \neq [B]^{\varphi_A(x)}$*, o que contradiz a hipótese* $[A]^{\alpha} = [B]^{\alpha}$ *para todo* $\alpha \in [0,1]$*. De maneira análoga chegamos a uma contradição se admitirmos que* $\varphi_A(x) < \varphi_B(x)$*.*

1.4 Números Fuzzy

Temos na teoria dos conjuntos fuzzy o objetivo de calcular quantidades imprecisas. Por exemplo, se considerarmos o dobro de uma quantidade "em torno de 10", pensamos como resultado uma quantidade "em torno de 20". Para tal cálculo, serão criados objetos que generalizam os números reais, e serão chamados de *Números Fuzzy* [12].

Definição 1.6 *Chamamos de número fuzzy um subconjunto fuzzy* B *quando o domínio da função de pertinência* φ_B *é o conjunto dos números reais satisfazendo as três condições que seguem:*

1. Todos os α-níveis de B são não vazios, onde $0 < \alpha \leq 1$,
2. Todos os α-níveis de B são intervalos fechados de $\mathbb{R}$,
3. O suporte de B, $suppB = \{x \in U : \varphi_B(x) > 0\}$, é limitado.

Os números fuzzy, mais utilizados neste trabalho são os triangulares e trapezoidais. Serão apresentadas, a seguir, as definições formais para tais números fuzzy.

Definição 1.7 *Um número fuzzy* B *é do tipo triangular se sua função de pertinência tem o formato de um triângulo e é da forma*

$$\varphi_B(x) = \begin{cases} 0 & se\ x \leq a \\ \dfrac{x-a}{u-a} & se\ a < x \leq u \\ \dfrac{b-x}{b-u} & se\ u \leq x < b \\ 0 & se\ x \geq b. \end{cases} \tag{1.9}$$

Aplicando esta definição, no caso dito anteriormente dos números "em torno de 10", pode-se utilizar um número fuzzy triangular com a função de pertinência a seguir:

$$\varphi_B(x) = \begin{cases} 0 & \text{se } x \leq 9,5 \\ \dfrac{x-8}{2} & \text{se } 8 < x \leq 10 \\ \dfrac{12-x}{2} & \text{se } 10 \leq x < 12 \\ 0 & \text{se } x \geq 12. \end{cases} . \tag{1.10}$$

Pode-se observar que um número fuzzy do tipo triangular não é necessariamente simétrico, como na equação (1.10), pois o valor de $b-u$ pode não ser o mesmo de $u-a$ em (1.9). A tradução linguística para o número fuzzy triangular é "perto de"e se este for simétrico podemos utilizar o termo "em torno de".

Definição 1.8 *Um número fuzzy B é do tipo trapezoidal se sua função de pertinência tem o formato de um trapézio e é da forma*

$$\varphi_B(x) = \begin{cases} \dfrac{x-a}{b-a} & \textit{se } a < x \leq b \\ 1 & \textit{se } b \leq x < c \\ \dfrac{d-x}{d-c} & \textit{se } c < x \leq x < d \\ 0 & \textit{caso contrário.} \end{cases} . \tag{1.11}$$

Serão apresentadas as operações aritméticas para os números fuzzy, que são: *soma*, *multiplicação por uma constante*, *diferença*, *multiplicação entre números fuzzy* e *divisão*. Em seguida, um método prático com aplicação dos α-níveis.

Definição 1.9 *Considere A e B dois números fuzzy e $\eta \in \mathbb{R}$.*

a) *Soma:*

Temos que a soma dos números fuzzy A e B é o número fuzzy $A+B$, que possui função de pertinência dada por

$$\varphi_{A+B}(z) = \begin{cases} \sup\limits_{\phi(z)} min[\varphi_A(x), \varphi_B(y)] & se\ \phi(z) \neq 0 \\ 0 & se\ \phi(z) = 0 \end{cases}, \tag{1.12}$$

em que $\phi(z) = \{(x,y) : x+y = z\}$.

b) *Multiplicação por Escalar:*

Temos que a multiplicação de um escalar η pelo número fuzzy A é o número fuzzy ηA, que possui função de pertinência dada por

$$\varphi_{\eta A}(z) = sup_{\{(x,y):\eta x = z\}}[\varphi_A(x)] = \begin{cases} \sup\limits_{\{x:\eta x = z\}} [\varphi_A(x)] & se\ \eta \neq 0 \\ \chi_{\{0\}}(z) & se\ \eta = 0 \end{cases}$$

$$\varphi_{\eta A}(z) = \begin{cases} \varphi_A(\eta^{-1}z) & se\ \eta \neq 0 \\ \chi_{\{0\}}(z) & se\ \eta = 0 \end{cases} \tag{1.13}$$

em que $\chi_{\{0\}}$ é a função característica de $\{0\}$.

c) *Diferença:*

Temos que a diferença entre os números fuzzy A e B é o número fuzzy $A-B$, que possui função de pertinência dada por

$$\varphi_{A-B}(z) = \begin{cases} \sup\limits_{\phi(z)} min[\varphi_A(x), \varphi_B(y)] & se\ \phi(z) \neq 0 \\ 0 & se\ \phi(z) = 0 \end{cases}, \tag{1.14}$$

em que $\phi(z) = \{(x,y) : x-y = z\}$.

d) *Multiplicação entre Números Fuzzy:*

Temos que a multiplicação entre os números fuzzy A e B é o número fuzzy $A \cdot B$, que possui função de pertinência dada por

$$\varphi_{A.B}(z) = \begin{cases} \sup\limits_{\phi(z)} min[\varphi_A(x), \varphi_B(y)] & se\ \phi(z) \neq 0 \\ 0 & se\ \phi(z) = 0 \end{cases}, \tag{1.15}$$

em que $\phi(z) = \{(x, y) : x.y = z\}$.

e) *Divisão:*

Temos que a divisão do número fuzzy A pelo número fuzzy B, se $0 \notin suppB$ é o número fuzzy descrito por uma função de pertinência dada por

$$\varphi_{A/B}(z) = \begin{cases} \sup\limits_{\phi(z)} min[\varphi_A(x), \varphi_B(y)] & se\ \phi(z) \neq 0 \\ 0 & se\ \phi(z) = 0 \end{cases}, \tag{1.16}$$

em que $\phi(z) = \{(x, y) : x/y = z\}$.

Esta é a definição de operações elementares entre números fuzzy [2].

Teorema 1.2 *Sejam A e B números fuzzy com α-níveis dados, respectivamente, por $[A]^\alpha = [a_1^\alpha, a_2^\alpha]$ e $[B]^\alpha = [b_1^\alpha, b_2^\alpha]$. Então valem as seguintes propriedades:*

(a) A soma entre A e B é o número fuzzy $A + B$ cujos α-níveis são

$$[A + B]^\alpha = [A]^\alpha + [B]^\alpha = [a_1^\alpha + b_1^\alpha, a_2^\alpha + b_2^\alpha].$$

(b) A diferença entre A e B é o número fuzzy $A - B$ cujos α-níveis são

$$[A - B]^\alpha = [A]^\alpha - [B]^\alpha = [a_1^\alpha - b_1^\alpha, a_2^\alpha - b_2^\alpha].$$

(c) A multiplicação de um escalar λ por A é o número fuzzy λA cujos α-níveis são

$$[\lambda A]^\alpha = \lambda [A]^\alpha$$

(d)A multiplicação de A por B é o número fuzzy $A \cdot B$ cujos α-níveis são

$$[A \cdot B]^{\alpha} = [A]^{\alpha}[B]^{\alpha} = [minP, maxP],$$

onde $P = \{a_1^{\alpha}b_1^{\alpha}, a_1^{\alpha}b_2^{\alpha}, a_2^{\alpha}b_1^{\alpha}, a_2^{\alpha}b_2^{\alpha}\}$.

(e) A divisão de A por B, se $0 \notin suppB$*, é o número fuzzy cujos α-níveis são*

$$\left[\frac{A}{B}\right]^{\alpha} = \frac{[A]^{\alpha}}{[B]^{\alpha}} = [a_1^{\alpha}, a_2^{\alpha}]\left[\frac{1}{b_2^{\alpha}}, \frac{1}{b_1^{\alpha}}\right].$$

Será feita, a seguir, a generalização, por meio de α-níveis, das operações aritméticas para intervalos [27]. Seja $\otimes$ qualquer uma das operações mencionadas de a até e, os α-níveis do conjunto $A \otimes B$ são dados por

$$[A \otimes B]^{\alpha} = [A]^{\alpha} \otimes [B]^{\alpha}$$

para todo $\alpha \in [0, 1]$.

1.5 Lógica Fuzzy

Na literatura, o termo "lógica fuzzy"é utilizado de duas formas: a primeira para representar e manipular informações inexatas com o propósito de tomar decisões sem considerar a teoria dos conjuntos fuzzy, suas funções de pertinência e suas álgebras em geral. A segunda refere-se à extensão da lógica clássica.

A lógica fuzzy não lida com questões ambíguas. As incertezas das quais a lógica fuzzy trata são do tipo monotônicas no sentido de que quanto menos incertas forem as premissas, menos incertas serão as conclusões e, neste sentido, pode-se dizer que a lógica clássica é uma espécie de limite da lógica fuzzy quando as incertezas tendem a zero.

O sucesso da lógica fuzzy vem do seu caráter prático, pois possibilita conclusões a partir de proposições incertas. A área que trata de formalizações dessas proposições é conhecida como *raciocínio aproximado* e sua arquitetura tem a forma do método de investigação proposto por Sócrates:

Gripe forte provoca febre alta

Febre alta provoca dores de cabeça frequentemente

Conclusão: Gripe forte provoca dores de cabeça frequentemente.

A última sentença (conclusão) é uma dedução obtida a partir das premissas, porém, alguns dos predicados não são termos precisos e, por este motivo, a lógica clássica não trata dessas sentenças. Vejamos a seguir alguns conceitos da lógica tradicional que nos servirão de base para descrever e compreender a lógica fuzzy [2].

Os primeiros passos da lógica matemática são realizados com o estudo dos conectivos "e", "ou", dentre outros, e estes são tipicamente usados na modelagem matemática de expressões do tipo:

"Se a está em A e b está em B,

então c está em C".

$A \subset B$ e $a, b \in A$.

Portanto, $c \in C$.

Para elaborar métodos dedutivos como este, para o caso fuzzy, é necessário o conceito de variáveis linguísticas. A noção de tal conceito é o mesmo de uma variável definida na matemática clássica, ou seja, ambas substituem valores.

Para o caso da variável linguística os valores assumidos são termos linguísticos associados a subconjuntos fuzzy, em particular, números fuzzy. Então, os valores de uma variável linguística serão termos linguísticos, como: altíssimo, alto, porte médio, baixo, muito baixo, dentre outros, que terão como significado fornecidos quantitativamente por conjuntos fuzzy.

Exemplo 1.5 *Vamos construir um aparelho de controle de velocidade de veículos, que tenha por objetivo o controle de velocidade em um determinado ponto de uma estrada. "Velocidade"é uma variável linguística que pode assumir os valores "baixa", "média"e "alta". Considere "baixa"uma velocidade inferior a cerca de* $40km/h$*, "média"uma velocidade em torno de* $80km/h$ *e "alta"uma velocidade superior a cerca de "120km/h". Cada valor ou termo assumido pela variável "Velocidade"é expresso qualitativamente por um dos termos linguísticos e quantitativamente por um subconjunto fuzzy.*

As proposições que associam uma variável linguística a um termo são denominadas proposições fuzzy. Do exemplo 1.5 temos três proposições fuzzy:

Velocidade é baixa;

Velocidade é média;

Velocidade é alta.

Na lógica clássica as proposições são unicamente "Verdadeiras"ou "Falsas", e podem ser compostas com o auxílio dos conectivos mencionados anteriormente. O valor lógico de uma proposição composta depende dos valores lógicos de cada uma dessas proposições envolvidas na composição e da regra de cada conectivo definida por tabelas verdade. As sentenças verdadeiras têm valor lógico 1, enquanto as sentenças falsas têm valor lógico 0.

Porém, para avaliar logicamente, em se tratando da teoria dos conjuntos fuzzy, uma proposição composta do tipo "Se a está em A e b está em B, então c está em C"devemos inicialmente atribuir um valor pertencente ao intervalo $[0, 1]$ que indique o quanto a proposição "a é A"é verdadeira e realizar a avaliação lógica, no sentido fuzzy, de cada um dos conectivos encontrados na proposição. Para tal, é necessário estender os conectivos por meio das normas triangulares.

Definição 1.10 *Uma co-norma triangular (s-norma) é uma operação binária* $s : [0, 1] \times [0, 1] \to [0, 1]$ *satisfazendo as seguintes condições: a. Comutatividade:* $xsy = ysx$

b. Associatividade: $xs(ysz) = (xsy)sz$

c. Monotonicidade: Se $x \leq y$ *e* $w \leq z$ *então* $xsw \leq ysz$

d. Condições de fronteira: $xs0 = x$, $xs1 = 1$.

A s-norma estende o operador $\vee$ que modela o conectivo "ou"[27].

Definição 1.11 *Uma norma triangular (t-norma) é uma operação binária* $t : [0, 1] \times [0, 1] \to [0, 1]$ *satisfazendo as seguintes condições: a. Comutatividade:* $xty = ytx$

b. Associatividade: $xt(ytz) = (xty)tz$

c. Monotonicidade: Se $x \leq y$ *e* $w \leq z$ *então* $xtw \leq ytz$

d. Condições de fronteira: $xt0 = 0$, $xt1 = x$.

A t-norma estende o operador $\wedge$ que modela o conectivo "e"[27].

Definição 1.12 *Uma aplicação $\mu : [0,1] \to [0,1]$ é uma negação se satisfazer as seguintes condições: a. Monotonicidade: μ é decrescente*

b. Involução: $\mu(\mu(x)) = x$

c. Condições de fronteira: $\mu(0) = 1$, $\mu(1) = 0$.

Definição 1.13 *Qualquer operação $\Rightarrow: [0,1] \times [0,1] \to [0,1]$ que reproduza a tabela verdade da implicação clássica é denominada implicação fuzzy.*

1.6 Relação Fuzzy

Na teoria clássica de conjuntos, qualquer subconjunto do produto cartesiano usual $V_1 \times V_2 \times \ldots \times V_n$ é uma relação R sobre $V_1 \times V_2 \times \ldots \times V_n$ e pode ser representada por uma função chamada de *função característica*, dada a seguir:

$$\chi_R : V_1 \times V_2 \times \ldots \times V_n \to \{0,1\}.$$

com

$$\chi_R(x_1, x_2, \ldots, x_n) = \begin{cases} 1 & \text{se } (x_1, x_2, \ldots, x_n) \in R \\ 0 & \text{se } (x_1, x_2, \ldots, x_n) \notin R. \end{cases}.$$

Quando se relaciona conjuntos fuzzy, ao invés de conjuntos clássicos, a relação é considerada uma *relação fuzzy*. Para definir relação fuzzy estendemos o conceito de função característica de uma relação clássica para uma função de pertinência.

Definição 1.14 *Uma relação fuzzy R sobre $V_1 \times V_2 \times \ldots \times V_n$ é qualquer subconjunto fuzzy de $V_1 \times V_2 \times \ldots \times V_n$ tal que sua função de pertinência é dada por*

$$\varphi_R : V_1 \times V_2 \times \ldots \times V_n \to [0,1].$$

O número $\varphi_R(x_1, x_2, \ldots, x_n) \in [0,1]$ indica o grau com que os elementos x_i estão relacionados segundo a relação dada R.

A operação produto cartesiano é, na teoria dos conjuntos fuzzy, similar à intersecção. A

principal diferença é que na teoria dos conjuntos fuzzy os subconjuntos fuzzy são de um mesmo universo e no produto cartesiano clássico, esses subconjuntos podem ser diferentes.

Definição 1.15 *Um produto cartesiano fuzzy* $A_1 \times A_2 \times \ldots \times A_n$ *dos subconjuntos fuzzy* A_1, A_2, ...,A_n *de* $V_1 \times V_2 \times \ldots \times V_n$ *é a relação fuzzy* R *cuja função de pertinência é dada por*

$$\varphi_R(x_1, x_2, \ldots, x_n) = \varphi_{A_1}(x_1) \wedge \varphi_{A_2}(x_2) \wedge \ldots \wedge \varphi_{A_n}(x_n)$$

onde $\wedge$ *é a t-norma min.*

Um caso particular da relação binária R definida no produto cartesiano é a composição $R \circ S$, entre duas relações fuzzy binárias em $V_1 \times V_2$ e$V_2 \times V_3$, respectivamente. Tal relação será definida a seguir.

Definição 1.16 *A composição* $R \circ S$ *é uma relação fuzzy binária em* $V_1 \times V_3$ *com função de pertinência expressa por*

$$\varphi_{R \circ S}(x_1, x_3) = \sup_{x_2 \in V_2} [(\varphi_R(x_1, x_2)) t (\varphi_S(x_2, x_3))]$$

Matematicamente, o conceito de relação é formalizado a partir da teoria de conjuntos. A escolha entre o modelo clássico e o modelo aplicado à teoria dos conjuntos fuzzy depende do fenômeno a ser estudado, mas a escolha pela teoria dos conjuntos fuzzy sempre tem maior peso no sentido de que esta teoria inclui a teoria clássica de conjuntos. A seguir descreveremos um dos principais objetos de estudo para este trabalho, os chamados Sistemas Baseados em Regras Fuzzy.

1.7 Sistema Baseado em Regras Fuzzy

Em nosso cotidiano, as ações humanas controlam os mais diversos sistemas do mundo real por meio de informações imprecisas. Cada indivíduo funciona da seguinte forma: recebe informações que são interpretadas segundo seus parâmetros e então decide qual atitude tomar. O controle e a execução de tarefas devem seguir uma sequência de "ordens"linguísticas, traduzida por um conjunto de regras, capazes de serem decodificadas pelo controlador.

Uma tentativa de reproduzir a estratégia de um controlador humano, na execução de suas tarefas, é dada pelos *controladores fuzzy*, considerados como um caso típico de um *Sistema Baseado em Regras Fuzzy (SBRF)*, ou seja, dada uma entrada fuzzy, um sistema que utiliza da teoria dos conjuntos fuzzy para produzir saídas.

Um SBRF contém quatro componentes, que são: um processador de entrada, que realiza a fuzzificação dos dados de entrada, uma coleção de regras fuzzy chamada de base de regras, uma máquina de inferência fuzzy e um processador de saída, que fornece como saída um vetor. Cada um desses componentes estão conectados conforme indicados na Figura 1.9, supondo $x \in \mathbb{R}^n$ e $y \in \mathbb{R}^m$.

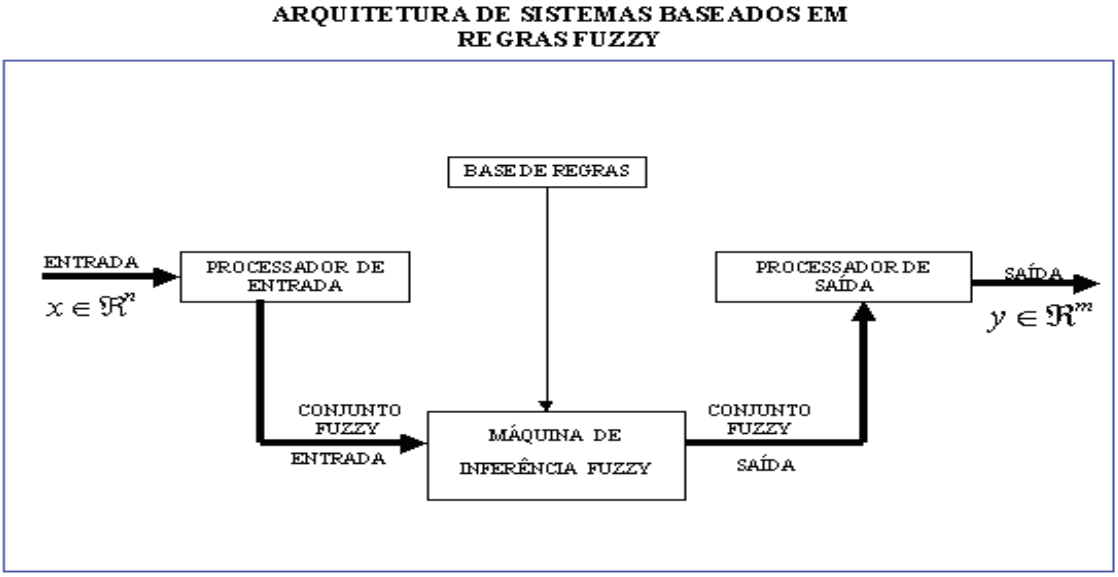

Figura 1.9: Sistema Baseado em Regras Fuzzy [10].

1.7.1 Processador de Entrada (Fuzzificação)

A fuzzificação é o estágio onde as entradas do sistema baseado em regras fuzzy são modeladas por conjuntos fuzzy com seus respectivos domínios. Justifica-se, assim, a grande importância de especialistas no fenômeno a ser modelado, pois as funções de pertinência são formuladas para cada conjunto fuzzy envolvido no processo. A entrada de dados é um vetor, mas a manipulação desses dados no sistema baseado em regras fuzzy é baseada na teoria dos conjuntos fuzzy. Após a fuzzificação as variáveis serão descritas por conjuntos fuzzy e suas funções de pertinência, que como dito anteriormente, são elaboradas pelo especialista.

1.7.2 Base de Regras

No sistema baseado em regras fuzzy, cada proposição fuzzy tem a forma "se (estado) - então (resposta)", em que cada "estado"e cada "resposta"são valores assumidos por variáveis linguísticas, e esses, por sua vez, são modelados por conjuntos fuzzy. Os conjuntos fuzzy que compõem o "estado"são chamados de *antecedentes*, enquanto os conjuntos fuzzy que compõem a "resposta"são chamados de *consequentes*. A coleção de regras fuzzy forma a base de regras. Um exemplo típico de base de regras é o caso abaixo:

"Se a banana está *amarela*, então a banana está *madura*"

Neste caso, temos que o conjunto fuzzy antecedente seria aquele que classificaria uma banana pelo grau de pertinência ao grupo de frutas amarelas, ou seja, quanto mais amarela, maior o grau de pertinência a este conjunto. O consequente seria representado pelo conjunto fuzzy representado pela função de pertinência ao conjunto das frutas maduras, isto é, quanto mais madura, maior o grau de pertinência a este conjunto.

A base de regras pode ser elaborada a partir do conhecimento e experiência do especialista no processo a ser estudado, ou baseado no aprendizado utilizando dados, ou seja, o sistema se encarrega de criar a sua própria base de regras a partir dos dados.

1.7.3 Máquina de Inferência Fuzzy

É neste componente que cada proposição fuzzy é traduzida matematicamente por meio das técnicas da teoria dos conjuntos fuzzy. Definimos as s-normas, as t-normas e as regras de inferência que serão utilizadas para obtermos a relação fuzzy que modela a base de regras. Este componente é tão importante quanto a base de regras, pois é dele que depende o sucesso do sistema baseado em regras fuzzy, já que sairá daqui a saída fuzzy a ser adotada pelo controlador, a partir de cada entrada fuzzy. Neste trabalho está sendo utilizado o Método de Inferência de Mamdani, que será descrito a seguir.

Método de Inferência de Mamdani

No método de inferência proposto por Mamdani, as regras fuzzy são interpretadas pelo produto cartesiano e agregadas pelo conectivo "ou"modelado pela s-norma máximo. Ou seja, o

método de inferência de Mamdani se resume na união da composição das relações de entradas com as definidas por cada regra. Para ilustrar o método de inferência considere as regras

$$R_1: \quad \text{Se } x \text{ é } A_1 \text{ e } y \text{ é } B_1 \text{ então } z \text{ é } C_1;$$
$$R_2: \quad \text{Se } x \text{ é } A_2 \text{ e } y \text{ é } B_2 \text{ então } z \text{ é } C_2.$$

A Figura 2.9 ilustra como uma saída z de um sistema de inferência do tipo Mamdani é gerada a partir das entradas x e y, vistos como conjuntos unitários. Observe que a saída obtida pelo método de Mamdani é um conjunto fuzzy, sendo portanto, necessário um método para obter um número real que a represente. Estes processos são chamados métodos de Defuzzificação e serão apresentados na próxima seção.

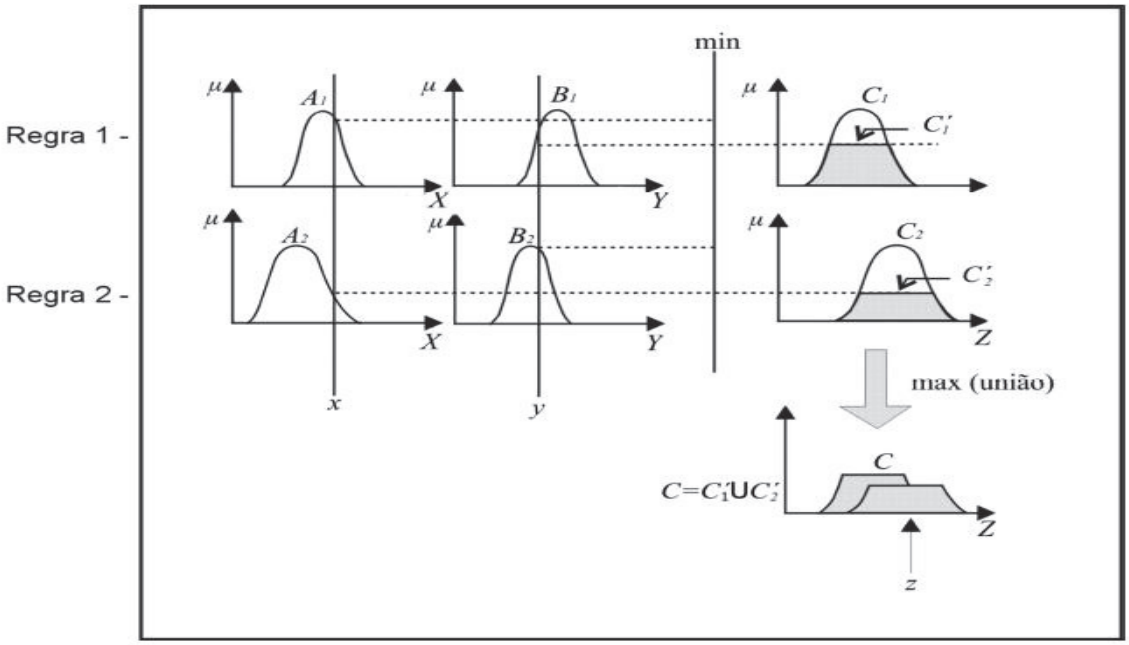

Figura 1.10: Saída final do método de Mamdani.

1.7.4 Processador de Saída (Defuzzificação)

No sistema baseado em regras fuzzy, a cada entrada fuzzy o método de inferência produz uma saída fuzzy que indica o controle que deve ser adotado. Entretanto, se a entrada for um número real, é esperado que a saída correspondente seja também um número real, mas isso em geral não ocorre em sistemas baseados em regras fuzzy pois, mesmo para uma entrada real, a saída é fuzzy. Deve, então, existir um método para defuzzificar a saída e assim obter um número real que indicará o controle a ser adotado.

São muitos os métodos de defuzzificação que podem ser adotados. Qualquer número real que, de alguma forma, pode representar razoavelmente o conjunto fuzzy B pode ser chamado de um defuzzificador de B.

O mais comum dentre esses métodos é o Método do Centro de Gravidade [27].

Este método assemelha-se à média ponderada para a distribuição de dados, com diferença que os pesos são os valores que indicam o grau de compatibilidade da saída z_i com o conceito modelado pelo conjunto fuzzy B em z. As equações (1.17) e (1.18) apresentam o centro de gravidade $G(B)$, respectivamente, para os domínios discreto e contínuo.

$$G(B) = \frac{\sum_{i=0}^{n} z_i \varphi_B(z_i)}{\sum_{i=0}^{n} \varphi_B(z_i)}. \tag{1.17}$$

$$G(B) = \frac{\int_R z \varphi_B(z)}{\int_R \varphi_B(z)}. \tag{1.18}$$

Também são métodos de defuzzificação o Centro dos Máximos e Média dos Máximos. Omitiremos o aprofundamento de tais métodos, mas tais esclarecimentos e definições podem ser encontrados em [2].

Conhecer os sistemas baseados em regras fuzzy, bem como todos os seus componentes e funções é de fundamental importância para o bom entendimento deste trabalho, visto que o objetivo principal é aplicar a teoria dos modificadores linguísticos, obtendo, assim, os sistemas fuzzy modificados, porém, este será um assunto tratado posteriormente.

No próximo capítulo serão definidos Equações Diferenciais Ordinárias Numéricas e Métodos Numéricos de Integração.

Capítulo 2

Métodos Numéricos para Equações Diferenciais Ordinárias

Serão estudados neste capítulo alguns esquemas numéricos para a resolução de problemas que podem ser representados por um modelo matemático que emprega equações diferenciais ordinárias. Um esquema numérico é considerado eficiente quando este apresenta soluções dentro de uma precisão desejada, com custo computacional baixo. Os erros cometidos nesta aproximação são decorrentes da discretização do problema, ou seja, passar do modelo matemático para o esquema numérico e da forma como as máquinas apresentam os dados numéricos.

Os esquemas numéricos são classificados como esquemas diretos e esquemas iterativos. Os esquemas diretos fornecem a solução após um número finito de passos enquanto que os esquemas iterativos repetem um número de passos, até que um critério de parada seja satisfeito [20].

2.1 Métodos Numéricos para Solução de Equações Diferenciais Ordinárias

Muitos dos modelos matemáticos, de diversas áreas de pesquisa, são representados por equações diferenciais. Em muitos casos, a teoria garante a existência e unicidade a solução, porém, nem sempre podemos obter a forma analítica desta solução.

Vamos analisar esquemas numéricos para solução de Problemas de Valor Inicial (PVI) para equações diferenciais ordinárias de primeira ordem. Ou seja, encontrar $y(x)$ tal que

$$\begin{cases} y' & = f(x,y) \\ y(x_0) & = y_0. \end{cases} . \tag{2.1}$$

Os esquemas numéricos calculam a aproximação de $y(x)$ em $x_1, x_2, x_3, \ldots$, em que $x_k = x_0 + kh$ para um passo $h > 0$. O valor da função em x_k é y_k, que é obtido em função dos anteriores $y_{k-1}, y_{k-2}, \ldots, y_1, y_0$. Desta forma, aproximamos a função para $x > x_0$, o que justifica o PVI.

São duas classes de métodos:

- Métodos de Passo Simples: O cálculo de y_k, depende apenas de y_{k-1}.
- Métodos de Passo Múltiplo: O cálculo e y_k depende de m valores anteriores $y_{k-1}, y_{k-2}, \ldots, y_{k-m}$. Neste caso, o método é dito e m-passos.

2.1.1 Método de Euler

Considere o problema de valor inicial (PVI) dado pela equação (2.1). Uma forma de aproximar a derivada de uma função no ponto x_1, é dado por

$$\lim_{h\to 0} \frac{y(x_0+h) - y(x_0)}{h} \approx \frac{y(x_0+h) - y(x_0)}{h}. \tag{2.2}$$

Como $x_1 = x_0 + h$ pelo PVI (2.1), temos

$$\frac{y(x_1) - y(x_0)}{h} = \frac{y_1 - y_0}{h} = f(x_0, y_0) \Longrightarrow y_1 = y_0 + hf(x_0, y_0). \tag{2.3}$$

Relacionamos y_1 com y_0, um valor dado ao problema de valor inicial. Então, obtemos uma aproximação $y(x_1) \approx y_1$.

Analogamente, obtemos y_2 em função de y_1, e de uma forma geral teremos

$$y_{k+1} = y_k + hf(x_k, y_k). \tag{2.4}$$

Este método é conhecido como Método de Euler. Neste método temos que a cada valor calculado o erro está aumentando. Isto acontece por cometermos um erro local na aproximação da derivada, e este erro vai se acumulando a cada valor calculado.

O erro local é estimado por

$$|E_{loc}(x_{k+1})| \leq \frac{h^2}{2} \max_{\xi \in [x_k, x_{k+1}]} |y''(\xi)|. \tag{2.5}$$

Em seguida serão apresentados os métodos de Runge-Kutta.

2.1.2 Métodos de Runge-Kutta

O método de Runge-Kutta de 1ª ordem é o Método de Euler, que coincide com o método da série de Taylor de 1ª ordem. Vamos determinar um método de 2ª ordem, conhecido como Método de Euler Melhorado ou Método de Heun. O objetivo é melhorar o Método de Euler, afim de aprimorar sua precisão.

Considere a aproximação $ye_{n+1} = y_n + hf(x_n, y_n)$, que é obtida pela aproximação do método de Euler.

Seja a reta L_1, que tem coeficiente angular dado por $y'(x_n) = f(x_n, y_n)$.

$$\begin{aligned} L_1(x) &= y_n + (x - x_n)y'_n \\ &= y_n + (x - x_n)f(x_n, y_n) \end{aligned} \tag{2.6}$$

$$\begin{aligned} L_1(x_{n+1}) &= y_n + (x_{n+1} - x_n)y'_n \\ &= y_n + hf(x_n, y_n) \\ &= ye_{n+1}. \end{aligned} \tag{2.7}$$

Considere a reta L_2, com coeficiente angular dado por $f(x_{n+1}, ye_{n+1}) = f(x_n, y_n + hy'_n)$ que passa pelo ponto P. Então,

$$L_2(x) = ye_{n+1} + (x - x_{n+1})f(x_{n+1}, ye_{n+1}) \tag{2.8}$$

Considere agora a reta L_0, que passa pelo ponto P e tem coeficiente angular como sendo a média dos coeficientes angulares de L_1 e L_2. Finalmente, a reta que passa pelo ponto (x_n, y_n) e é paralela à reta L_0, tem a forma

$$L(x) = y_n + (x - x_n)\frac{1}{2}(f(x_n, y_n) + f(x_{n+1}, y_n + hy'_n)). \tag{2.9}$$

Em $x_n + 1$, temos

$$y_{n+1} = y_n + h\frac{1}{2}(f(x_n, y_n) + f(x_{n+1})y_n + hy'_n)) \tag{2.10}$$

O valor de y_{n+1} está mais próximo do valor exato do que ye_{n+1}. A construção está explícita na Figura 2.1.

Este esquema é chamado de Método de Euler Melhorado, no qual o erro local é estimado por

$$|E_{loc}(x_{n+1})| \leq \frac{h^3}{6} \max_{\xi \in [x_n, x_{n+1}]} |y'''(\xi)| \tag{2.11}$$

O Método de Euler melhorado foi determinado por uma construção geométrica. Pode-se obter uma demonstração analítica desenvolvendo a série de Taylor da $f(x, y)$ calculado no ponto $(x_{n+1}, y_n + hy'_n)$.

A expressão encontrada deve coincidir com a série de Taylor até 2ª ordem. De modo geral, um método de Runge-Kutta de 2ª ordem é dado por

$$y_{n+1} = y_n + a_1 hf(x_n, y_n) + a_2 hf(x_n + b_1 h, y_n + b_2 hy'_n) \tag{2.12}$$

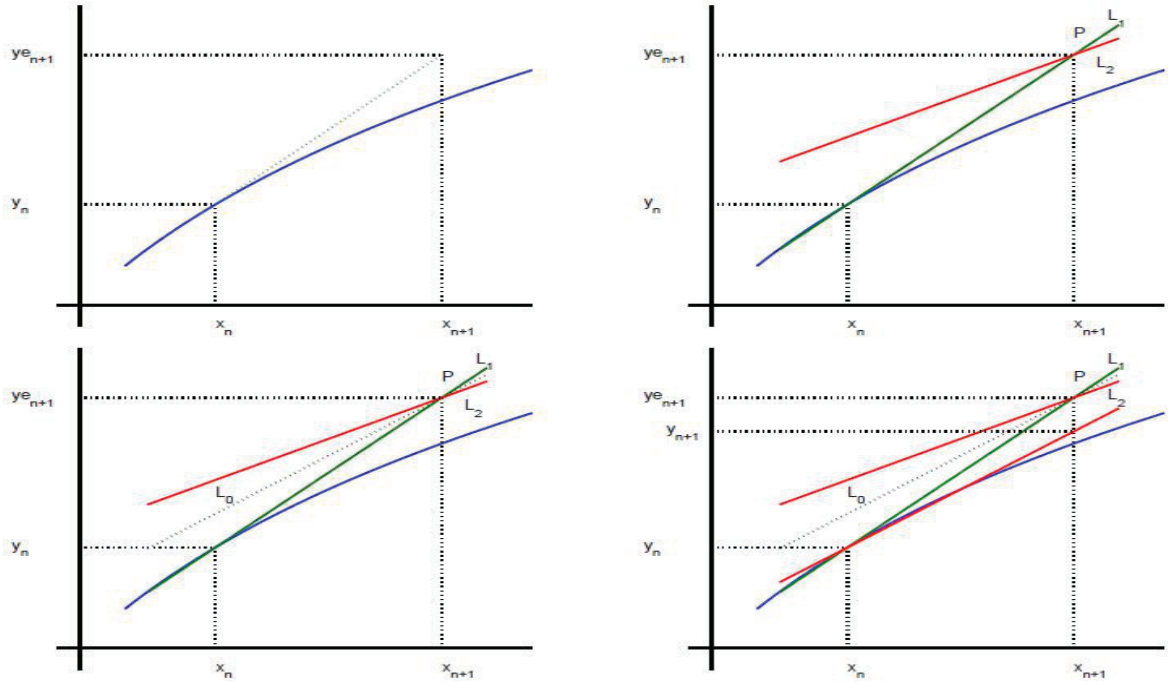

Figura 2.1: Método de Euler melhorado [20].

em que

$$\begin{cases} a_1 + a_2 &= 1 \\ a_2 b_1 &= 1/2 \\ a_2 b_2 &= 1/2. \end{cases} . \tag{2.13}$$

O Método de Euler melhorado é obtido com $a_1 = a_2 = \frac{1}{2}$ e $b_1 = b_2 = 1$. Métodos de ordem superior são obtidos seguindo o mesmo procedimento. Serão representados a seguir um método de 3^a e 4^a ordem.

Runge-Kutta de 3^a ordem:

$$\begin{aligned} y_{n+1} &= y_n + \frac{2}{9}K_1 + \frac{1}{3}K_2 + \frac{4}{9}K_3 \\ K_1 &= hf(x_n, y_n) \\ K_2 &= hf(x_n + h/2, y_n + K_1/2) \\ K_3 &= hf(x_n + 3h/4, y_n + 3K_2/4). \end{aligned} \tag{2.14}$$

O erro de truncamento local é da ordem de h^4 e o erro de truncamento global é de h^3, que representa a ordem de aproximação em um intervalo.

Runge-Kutta de 4^a ordem:

$$\begin{aligned} y_{n+1} &= y_n + \frac{1}{6}(K_1 + 2K_2 + 2K_3 + K_4) \\ K_1 &= hf(x_n, y_n) \\ K_2 &= hf(x_n + h/2, y_n + K_1/2) \\ K_3 &= hf(x_n + h/2, y_n + K_2/2) \\ K_4 &= hf(x_n + h, y_n + K_3). \end{aligned} \tag{2.15}$$

Neste caso, o erro de truncamento local é da ordem de h^5 e o erro de truncamento global é de h^4, que representa a ordem de aproximação em um intervalo.

2.1.3 Métodos de Passo Múltiplo

Vamos integrar o PVI (2.1) no intervalo $[x_n, x_{n+1}]$, ou seja

$$\int_{x_n}^{x_{n+1}} y'(x)dx = \int_{x_n}^{x_{n+1}} f(x, y(x))dx, \tag{2.16}$$

e pelo Teorema Fundamental do Cálculo

$$y(x_{n+1}) = y(x_n) + \int_{x_n}^{x_{n+1}} f(x, y(x))dx. \tag{2.17}$$

A integral sobre $f(x, y)$ é aproximada pela integral do polinômio interpolador que pode utilizar pontos que não pertencem a $[x_n, x_{n+1}]$. Dependendo dessa escolha dos pontos onde aproximaremos a $f(x, y)$, podemos ter duas classificações:

- Explícito: Quando utilizamos os pontos $x_n, x_{n-1}, x_{n-2}, \ldots, x_{n-m}$ para interpolar $f(x, y)$.

- Implícito: São obtidos quando no conjunto de pontos, sobre os quais interpolamos $f(x, y)$, temos o ponto x_{n+1}.

Serão dados a seguir alguns métodos numéricos de integração.

2.2 Integração Numérica

Nosso objetivo é estudar esquemas numéricos que aproximem a integral definida de uma função $f(x)$ num intervalo $[a, b]$. A integração numérica é aplicada quando a primitiva da função não é conhecida ou quando só conhecemos $f(x)$ num conjunto discreto de pontos. As fórmulas de Newton-Côtes aproximam a função $f(x)$ por um polinômio interpolador e a integral é aproximada por

$$\int_a^b f(x)dx \approx A_0 f(x_0) + A_1 f(x_1) + \cdots + A_n f(x_n) = \sum_{i=0}^{n} A_i f(x_i). \tag{2.18}$$

Nas fórmulas de Newton-Côtes os pontos são igualmente espaçados, isto é, $x_k = x_0 + kh$, com $h = (b-a)/n$, e os coeficientes A_i são determinados pelo polinômio escolhido para aproximar a função $f(x)$.

2.2.1 Regra do Trapézio

A forma de Newton para o polinômio $p_n(x)$ que interpola $f(x)$ em $x_0, x_1, \ldots, x_n$, $(n+1)$ pontos distintos é dada da forma

$$\begin{aligned} p_n(x) = f[x_0] + f[x_0, x_1](x - x_0) + f[x_0, x_1, x_2](x - x_0)(x - x_1) + \cdots \\ \cdots + f[x_0, x_1, \ldots, x_n](x - x_0)(x - x_1)\cdots(x - x_n), \end{aligned} \tag{2.19}$$

em que $f[x_0, \ldots, x_n]$ é o operador diferenças divididas [20].

A regra do trapézio é a fórmula de Newton que aproxima a função $f(x)$ pelo polinômio interpolador de grau 1 sobre os pontos $x_0 = a$ e $x_1 = b$. Este polinômio, na forma de Newton, é dado por

$$p_1(x) = f(x_0) + f[x_0, x_1](x - x_0). \tag{2.20}$$

Então, aproximamos a integral de $f(x)$ pela integral de $p_1(x)$, ou seja,

$$\int_a^b f(x)dx \cong \int_{x_0}^{x_1} p_1(x)dx. \tag{2.21}$$

Portanto, temos que:

$$\int_a^b f(x)dx = \frac{h}{2}(f(x_0) + f(x_1)), \tag{2.22}$$

em que $h = x_1 - x_0$. A fórmula (2.22) representa a área do trapézio que tem $f(x_1)$ e $f(x_0)$ como valores das bases e h como altura. Tal trapézio pode ser representado como na Figura 2.2.

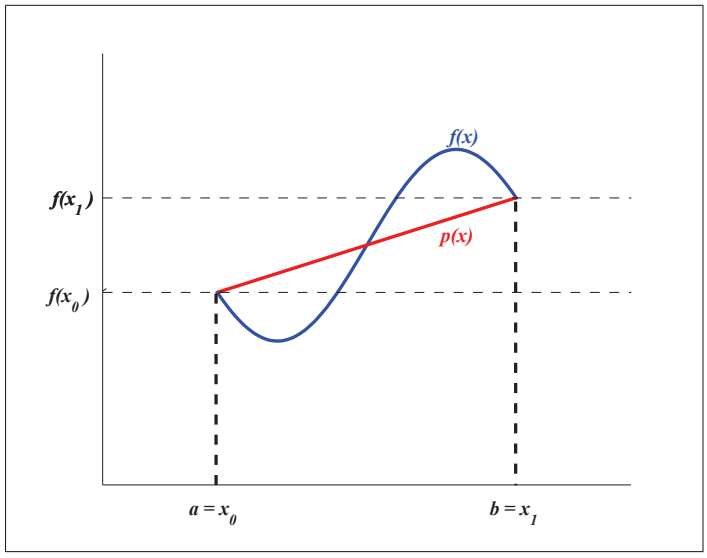

Figura 2.2: Aproximação pela Regra do Trapézio

O erro de truncamento é dado por

$$E_T = -\frac{h^3}{12} f''(\xi), \tag{2.23}$$

em que $\xi \in (a, b)$.

Na prática usaremos a estimativa

$$|E_T| \leq \frac{h^3}{12} \max_{\xi \in [a,b]} |f''(\xi)|. \tag{2.24}$$

Veremos, a seguir um método com maior precisão.

2.2.2 Regra do Trapézio Repetida

A regra do trapézio aproxima bem funções suaves e/ou em intervalos e integração de amplitude pequena [20].

Para intervalos de amplitude grande podemos fazer uma subdivisão do intervalo $[a, b]$ em n subintervalos de mesma amplitude e aplicamos a regra o trapézio em cada um eles. Temos que: $h = \dfrac{b-a}{n}$ e $x_p = x_0 + kh$ com $k = 0, 1, \ldots, n$. Aplicando a regra do trapézio em cada subintervalo $[x_p, x_{p+1}]$, temos que:

$$\int_a^b f(x)dx = \frac{h}{2}(f_0 + 2(f_1 + f_2 + f_3 + \cdots + f_{n-1}) + f_n) \tag{2.25}$$

O erro cometido é igual à soma dos erros de cada subintervalo, e portanto, o erro é da forma:

$$|E_{TR}| \leq n\frac{h^3}{12} \max_{\xi \in [a,b]} |f''(\xi)|. \tag{2.26}$$

2.2.3 Regra de $1/3$ Simpson

Para este método iremos utilizar o polinômio de grau 2 que interpola a função $f(x)$. Serão necessários três pontos: x_0, x_1, x_2.

Como estes pontos devem ser igualmente espaçados, tomamos $h = \dfrac{(b-a)}{2}$. Consideremos o polinômio interpolador na forma de Newton, como segue

$$p_2(x) = f(x_0) + (x - x_0)f[x_0, x_1] + (x - x_0)(x - x_1)f[x_0, x_1, x_2]. \tag{2.27}$$

Dessa forma temos que:

$$\begin{aligned} \int_a^b f(x)dx \quad &\approx \quad \int_{x_0}^{x_2} p_2(x)dx \\ &= \quad \frac{h}{3}(f(x_0) + 4f(x_1) + f(x_2)). \end{aligned} \tag{2.28}$$

O erro de truncamento é dado por

$$E_S = -\frac{h^5}{90} f^{(4)}(\xi), \xi \in [a, b]. \tag{2.29}$$

Na prática usamos a estimativa

$$|E_S| \leq \frac{h^5}{90} \max_{\xi \in [a,b]} |f^{(4)}(\xi)|. \tag{2.30}$$

2.2.4 Regra de 1/3 Simpson Repetida

A aproximação pode ser melhorada, da mesma forma que foi feito para o Método do Trapézio. Dividimos o intervalo de integração em n subintervalos de mesma amplitude.

Como para o Método de 1/3 Simpson é preciso ter três pontos, a regra se aplica a cada dois subintervalos da forma $[x_p, x_{p+2}]$, ou seja, n deve ser um número par. Então $h = \dfrac{b-a}{n}$ e $x_p = x_0 + ph$, para $p = 1, 2, \ldots, n$.

Aplicando 1/3 Simpson para cada subintervalo $[x_p, x_{p+2}]$, temos

$$\begin{aligned} \int_a^b f(x)dx &\approx \frac{h}{3}(f_0 + 4f_1 + f_2) + \frac{h}{3}(f_2 + 4f_3 + f_4) + \cdots + \frac{h}{3}(f_{n-2} + 4f_{n-1} + f_n) \\ &= \frac{h}{3}[f_0 + 4(f_1 + f_2 + \cdots + f_{n-1}) + 2(f_2 + f_4 + \cdots + f_{n-2}) + f_n]. \end{aligned}$$

O erro cometido é a soma dos erros em cada intervalo $[x_p, x_{p+2}]$ e como temos $\dfrac{n}{2}$ subintervalos, temos

$$|E_{SR}| \leq n\frac{h^5}{180} \max_{\xi \in [a,b]} |f^{(4)}(\xi)|.$$

Temos que as fórmulas de Newton-Côtes são obtidas quando aproximamos a função a ser integrada por um polinômio interpolados, cujo grau determina a proporção dos erros que são encontrados. Do ponto de vista prático, não é usual utilizar polinômios de grau muito alto para aproximar as funções.

A melhor estratégia é a utilização de fórmulas repetidas, que permitem melhor aproximação e precisão, usando fórmulas que são obtidas por polinômios de menor grau.

No próximo capítulo serão apresentados os Sistemas Baseados em Regras Fuzzy (SBRF) e os sistemas dinâmicos p-fuzzy utilizados para aplicar a teoria dos números fuzzy na solução de equações diferenciais ordinárias.

Capítulo 3

Sistemas p-Fuzzy para Modelos Populacionais

Na modelagem de fenômenos existem casos cujas variáveis de estado estão relacionadas às suas variações temporais. Esta relação é estabelecida a partir de parâmetros que precisam ser estimados, mas os casos que predominam são aqueles em que há necessidade de coleta de um número elevado de dados para descrever o fenômeno para que o mesmo possa ser modelado de forma adequada. Como ferramenta para este tipo de modelagem são utilizadas as equações diferenciais [2].

Trataremos neste momento da escolha de sistemas baseados em regras fuzzy que incorporam informações imprecisas em suas variáveis, nas suas variações e na relação entre variáveis e variações, podendo obter o tratamento matemático para tais incertezas. Esses sistemas são chamados de p-fuzzy, que significa parcialmente fuzzy. Serão propostos a aplicação de sistemas p-fuzzy para a solução de problemas modelados por meio de Equações de Diferenças e Equações Diferenciais Ordinárias (EDOs).

As bases de regras e os parâmetros das funções de pertinência do SBRF podem ser construídos a partir de dados obtidos experimentalmente ou com o auxílio de um especialista.

Os sistemas recebem o nome de p-fuzzy, pois são parcialmente fuzzy, no sentido que o campo de direções do Problema de Valor Inicial (PVI) em questão é conhecido parcialmente. No entanto, sua solução é real e em cada instante t, um valor é obtido após um processo de defuzzificação.

Definição 3.1 *Um PVI p-fuzzy pode ser dado por*

$$\begin{cases} \dfrac{du}{dt} & = F(t, u(t)) \\ u(a) & = u_0 \end{cases} \tag{3.1}$$

em que F é parcialmente conhecida e descrita por uma base de regras fuzzy.

Até agora nenhum processo de defuzzificação está sendo exigido para modelar F, o que desobriga que a solução $u(t)$ de (3.1) em algum sentido seja um número real. Porém, se algum método de defuzzificação for adotado, o que obtém-se é $u(t) \in \mathbb{R}$. Neste caso, estamos diante da metodologia que veremos a seguir para F como função dada por um controlador fuzzy [2].

3.1 Sistemas Dinâmicos Fuzzy

É proposta uma metodologia para solucionar sistemas dinâmicos tradicionais por meio dos Sistemas Baseados em Regras Fuzzy (SBRF). Os sistemas dinâmicos tradicionais tem a forma

$$x_{t+1} = F(x_t) \Leftrightarrow x_{t+1} - x_t = f(x_t) \tag{3.2}$$

para o caso discreto e

$$\frac{dx}{dt} = f(x) \tag{3.3}$$

para o caso contínuo.

O fato é que em ambos os casos o campo f representa a variação e o sistema é estudado a partir deste. Mas se f não for dada explicitamente ou se, por algum motivo, ela é conhecida apenas parcialmente, com base nas características do fenômeno que se deseja modelar para estudar a evolução do sistema, adotamos uma expressão matemática para f que contemple tais características.

A sugestão é adotar um modelo linguístico fuzzy cuja base de regras é "construída"a partir das características do fenômeno. Mais ainda, as variáveis de estado devem ser as entradas do sistema, enquanto que as saídas do sistema devem representar as variações dos estados. Esta é a particularidade dos SBRF aplicados a sistemas dinâmicos.

Como as regras são formuladas baseando-se no que se conhece do fenômeno, o esperado é que a função f_r dada pelo SBRF capte tal conhecimento. Então pode-se substituir o campo teórico f por f_r e estudamos as soluções da equação das diferenças

$$x_{t+1} - x_t = f_r(x_t) \tag{3.4}$$

para o caso discreto e da equação diferencial

$$\frac{dx}{dt} = f_r(x) \tag{3.5}$$

para o caso contínuo.

Dependendo das propriedades matemáticas de f e da metodologia escolhida para os SBRF, o esperado é que as soluções de (3.2) e (3.3) sejam convergentes para (3.4) e (3.5), respectivamente, uma vez que f_r converge para f a medida que o número de regras r aumenta [2]. Há situações em que a função f_r é explicitamente conhecida e, portanto, pode-se obter as soluções analíticas.

Considere que um fenômeno seja modelado matematicamente pelo PVI dado por

$$\begin{cases} \dfrac{dx}{dt} & = f(t,x) \\ x(t_0) & = x_0 \end{cases} \tag{3.6}$$

cuja solução é

$$x(t) = x_0 + \int_{t_0}^{t} f(s, x(s))ds. \tag{3.7}$$

Considere um sistema dinâmico fuzzy com r regras que levam em consideração características do campo f. Esta base de regras, por meio de um SBRF, define uma f_r.

Suponha que f_r seja convergente para f quando $r \to \infty$ e suponha ainda que a função f_r seja tal que o PVI dado por

$$\begin{cases} \dfrac{dx}{dt} & = f_r(t,x) \\ x(t_0) & = x_0 \end{cases} \tag{3.8}$$

tenha como solução

$$x_r(t) = x_0 + \int_{t_0}^{t} f_r(s, x(s))ds. \tag{3.9}$$

Admitimos que $x_r \to x$ quando $r \to \infty$ e assumimos que cada função f_r tenha propriedades que garantem que (3.8) está bem definido e sua solução seja dada por (3.9).

Serão usados métodos numéricos clássicos para obter a solução de (3.8), ou seja, o método fornecerá uma sequência x_{r_n} que estima o valor de x_r. Como $x_r \to x$ quando $r \to \infty$ e $x_{r_n} \to x_r$ quando $n \to \infty$, temos que $x_{r_n} \to x$ quando $n, r \to \infty$.

Esta é a metodologia que será utilizada para obter estimativas de soluções de equações diferenciais ordinárias a partir de SBRF, quando temos apenas informações parciais sobre o campo de direções, ou seja, quando estamos trabalhando com sistemas baseados em regras p-fuzzy.

3.2 Critérios de Comparação

No Capítulo a seguir serão apresentados modelos matemáticos de controle e crescimento populacional. Para a maioria desses modelos são apresentadas as soluções determinísticas e a solução dada pelo sistema p-fuzzy, e posteriormente serão feitas modificações no sistema p-fuzzy e será obtida a solução do sistema p-fuzzy modificado e p-fuzzy modificado no tempo.

Neste trabalho, temos como objetivo verificar se a solução p-fuzzy se aproxima a solução determinística. A mesma comparação será feita posteriormente com o sistema p-fuzzy modificado e p-fuzzy modificado no tempo. Para tal processo é necessário determinarmos critérios de comparação, e assim, verificar se um sistema p-fuzzy é ou não uma boa aproximação para a solução determinística.

Apresentaremos a metodologia utilizada para obter uma melhor aproximação entre a solução do sistema p-fuzzy e a solução determinística.

Seja x o vetor com os pontos da solução determinística e x' o vetor com os pontos gerados pelo sistema p-fuzzy, utilizando as funções de pertinência dos modelos a serem apresentados.

Consideramos dois tipos de erro, E_1 e E_2, descritos a seguir:

$$E_1 = max|x - x'|, \tag{3.10}$$

$$E_2 = \frac{max|x - x'|}{min|x'|}. \tag{3.11}$$

Dessa forma, para cada sistema p-fuzzy apresentado em cada um dos modelos, podemos determinar o quanto esta solução p-fuzzy se aproxima da solução determinística. Para o modelo do controle de pragas, último dos modelos apresentados neste capítulo, a saída final é o controle da praga, e para este modelo utilizamos um outro critério, que é o erro E_3 dado por:

$$E_3 = \frac{max|x - x'|}{max|x'|}. \tag{3.12}$$

É necessário, para este modelo, utilizar o erro E_3, pois o vetor solução proveniente do sistema p-fuzzy tem valores nulos, portanto não podemos efetuar o quociente, visto que ocorreria a divisão por zero. Para tanto, substituímos o $min|x'|$ por $max|x'|$ em (3.11), obtendo E_3.
Os programas utilizados para o desenvolvimento dos sistemas p-fuzzy, p-fuzzy modificados e modificados no tempo foram feitos no Matlab, e as rotinas para cada um dos modelos citados a seguir são apresentadas no Apêndice B.

3.3 Sistemas p-Fuzzy para Alguns Modelos Matemáticos

A seguir apresentamos a aplicação da teoria dos sistemas p-fuzzy e SBRF para alguns modelos matemáticos [2] e [12]. Para todos os modelos que serão citados a seguir, os sistemas baseados em regras fuzzy têm Mamdani como método de inferência, e Centro de Gravidade como método de defuzzificação.

3.3.1 Modelo do Crescimento Populacional de Malthus

Entre os modelos de crescimento populacional mais importantes está o "modelo de crescimento malthusiano". Tal modelo foi proposto por Thomas Robert Malthus (1766-1834), economista britânico considerado o pai da demografia por sua teoria de crescimento populacional [23]. Nomeado professor de história e de economia política em um colégio da Companhia das Índias (o East India Company College), em Haileybury, expôs suas ideias em dois livros conhecidos como Primeiro Ensaio e Segundo Ensaio: "Um ensaio sobre o princípio da população na medida em que afeta o melhoramento futuro da sociedade, com notas sobre as especulações de Mr. Godwin, M. Condorcet e outros escritores"(1798) e "Um ensaio sobre o princípio da população ou uma visão de seus efeitos passados e presentes na felicidade humana, com uma investigação das nossas expectativas quanto à remoção ou mitigação futura dos males que ocasiona"(1803). Malthus teve seu reconhecimento devido a uma obra sociológica e demográfica que publicou anonimamente, em 1798: "Um Ensaio sobre o Princípio de População".

É imprescindível salientar que Malthus não prescrevia originalmente nenhuma equação matemática de crescimento populacional, simplesmente enunciava que a população cresceria geometricamente enquanto que a produção de alimentos cresceria numa taxa aritmética, desde que não houvesse mecanismo de controle.

A primeira interpretação matemática da conjectura de Malthus foi possível a partir de um modelo determinístico, pressupondo que "o crescimento de uma população é proporcional à própria população"[2].

Dos princípios vistos anteriormente sobre sistemas p-fuzzy e mantendo esta linha de investigação, vamos estudar o PVI dado pela equação (3.6), em que f é substituída por uma base de regras fuzzy coerente com o modelo malthusiano, cuja principal suposição é que em cada instante t, a taxa de crescimento de uma população, é diretamente proporcional à população.

O modelo contínuo de Malthus é dado por

$$\begin{cases} \dfrac{dx}{dt} &= \alpha x(t) \\ x(0) &= x_0 \end{cases}, \tag{3.13}$$

e sua solução analítica é dada por $x(t) = x_0 e^{\alpha t}$, em que α é uma taxa de crescimento constante [12].

A base de regras constituída é para a taxa de variação por unidade de tempo, denotada por $\frac{dx}{dt}$ em função da população x. Dessa forma, x é a variável de entrada e $\frac{dx}{dt}$ é a variável de saída. A arquitetura do sistema p-fuzzy para o modelo de Malthus pode ser vista na Figura 3.1.

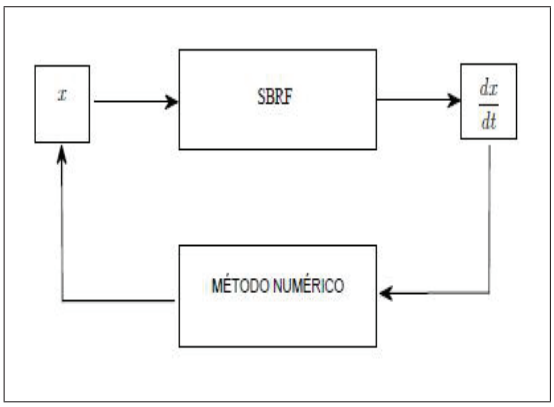

Figura 3.1: Arquitetura do sistema p-fuzzy para o modelo de Malthus [7].

Consideremos quatro qualificações para cada uma das variáveis linguísticas, da seguinte forma:

r_1**:** Se a população x é *muito baixa* (MB) então a variação $\frac{dx}{dt}$ é *muito baixa* (MB);

r_2**:** Se a população x é *baixa* (B) então a variação $\frac{dx}{dt}$ é *baixa* (B);

r_3**:** Se a população x é *média* (M) então a variação $\frac{dx}{dt}$ é *média* (M);

r_4**:** Se a população x é *alta* (A) então a variação $\frac{dx}{dt}$ é *alta* (A).

Nas Figuras 3.2 e 3.3 temos as funções de pertinência que correspondem, respectivamente, às variáveis de entrada e saída para o modelo malthusiano. Para os termos linguísticos MB e A, as funções de pertinência são trapezoidais e para as regras B e M temos as funções de pertinência triangulares.

Utilizando o SBRF, determinamos o valor de $\frac{dx}{dt}$ no próximo instante. Assim, calculamos o próximo valor de x, com integração numérica, aplicando a regra dos trapézios [12]. O raciocínio é repetido em 100 iterações obtendo a trajetória da população no tempo como mostra a Figura 3.4.

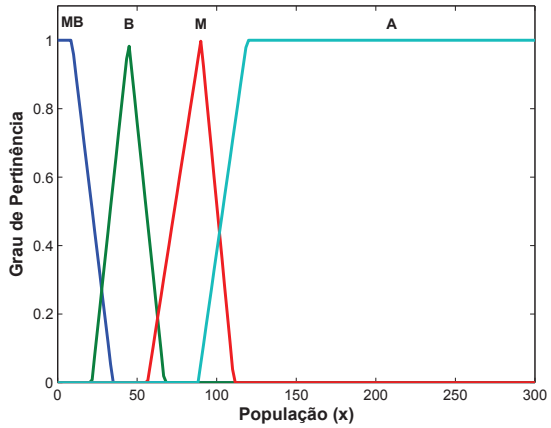

Figura 3.2: Funções de pertinência de entrada do modelo malthusiano.

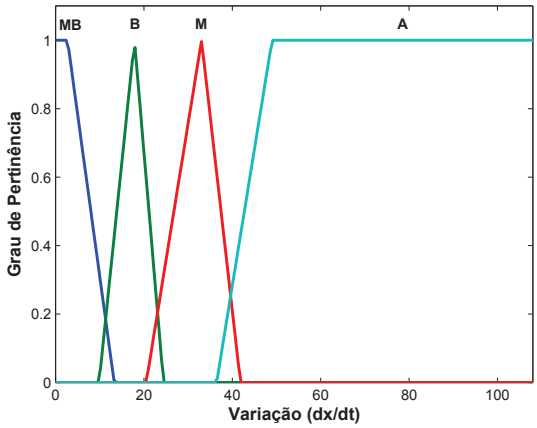

Figura 3.3: Funções de pertinência de saída do modelo malthusiano.

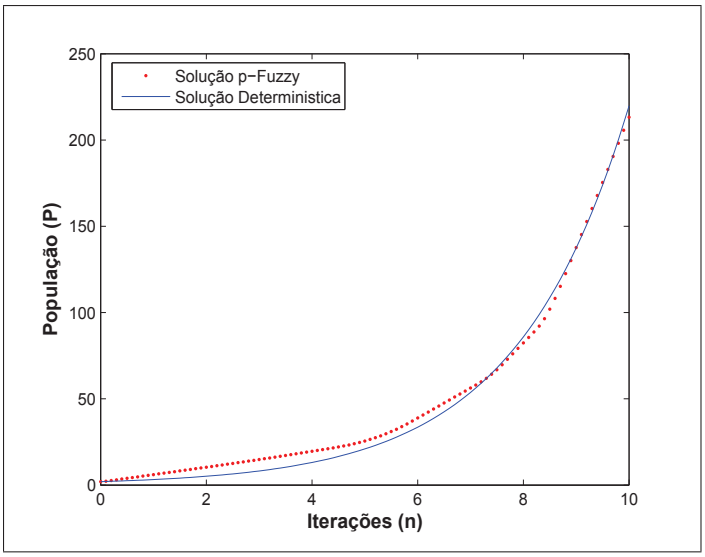

Figura 3.4: Solução do sistema p-fuzzy com população inicial igual a 2 e iterações $n = 100$.

A diferença entre as duas soluções foi calculada no Matlab utilizando os critérios dados pelas expressões (3.10) e (3.11) apresentados anteriormente, e os valores obtidos foram $E_1 = 7.2524$ e $E_2 = 3.6262$. Dado os valores encontrados, podemos dizer que a diferença entre as duas soluções é pequena, ou seja, o sistema p-fuzzy é uma boa aproximação para a solução obtida analiticamente a partir da equação diferencial ordinária.

3.3.2 Modelo Logístico de Verhulst

Pierre François Verhulst (1804-1849), natural de Bruxelas (Bélgica), foi matemático e doutor em teoria dos números da Universidade de Gante em 1825.

Verhulst iniciou seus estudos de filologia clássica em Bruxelas, mas logo interessou-se pela matemática. Ainda como estudante conquistou dois prêmios por seus trabalhos no cálculo das variações. Mais tarde publicou artigos no campo da teoria dos números e da física.

Seu modelo de crescimento populacional, proposto em 1838, é baseado na avaliação de estatísticas disponíveis e complementa a teoria do crescimento exponencial com termos representando os fatores de inibição do crescimento. Desde os anos 1970 do século XX a equação logística tem recebido grande atenção como exemplo importante da teoria do caos [24].

O modelo de Verhulst supõe que uma população, vivendo em um determinado meio, deverá crescer até um limite sustentável, ou seja, ela tende a se estabilizar.

O modelo tradicional de Verhulst para crescimento populacional é regido pelo PVI

$$\begin{cases} \dfrac{dP}{dt} &= rP(1-\frac{P}{P_\infty}) \\ P(0) &= P_0, r>0. \end{cases} \tag{3.14}$$

em que P_0 é a população inicial, r a razão de crescimento e P_∞ é o valor limite da população.

As soluções clássicas de (3.14), que representam as populações $P(t)$ em cada instante t, são dadas por

$$P(t) = \frac{P_\infty P_0}{(P_\infty - P_0)e^{-rt} + P_0}. \tag{3.15}$$

Para este modelo, denotaremos P para a população, que é a variável de entrada, e $\dfrac{1}{P}\dfrac{dP}{dt}$ para a taxa de crescimento relativa por unidade de tempo (ou taxa de crescimento específico), que é a variável de saída. A arquitetura do sistema p-fuzzy para o modelo de Verhulst pode ser vista na Figura 3.1.

As regras do modelo logístico de Verhulst estão de acordo com as principais características de um modelo geral de população com crescimento inibido que seja regulado por uma capacidade máxima. Este modelo é um caso particular.

Consideremos as qualificações para cada uma das variáveis linguísticas, da seguinte forma:

R_1: Se a população P é *baixa* (A_1) então a taxa de crescimento específico $\frac{1}{P}\frac{dP}{dt}$ é *baixa positiva* (B_2);

R_2: Se a população P é *média baixa* (A_2) então a taxa de crescimento específico $\frac{1}{P}\frac{dP}{dt}$ é *média positiva* (B_3);

R_3: Se a população P é *média* (A_3) então a taxa de crescimento específico $\frac{1}{P}\frac{dP}{dt}$ é *alta positiva* (B_4);

R_4: Se a população P é *média alta* (A_4) então a taxa de crescimento específico $\frac{1}{P}\frac{dP}{dt}$ é *média positiva* (B_3);

R_5: Se a população P é *alta* (A_5) então a taxa de crescimento específico $\frac{1}{P}\frac{dP}{dt}$ é *baixa positiva* (B_2);

R_6: Se a população P é *altíssima* (A_6) então a taxa de crescimento específico $\frac{1}{P}\frac{dP}{dt}$ é *baixa negativa* (B_1).

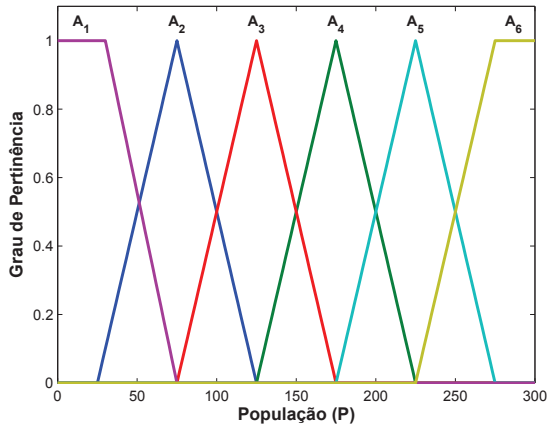

Figura 3.5: Funções de pertinência de entrada do modelo de Verhulst.

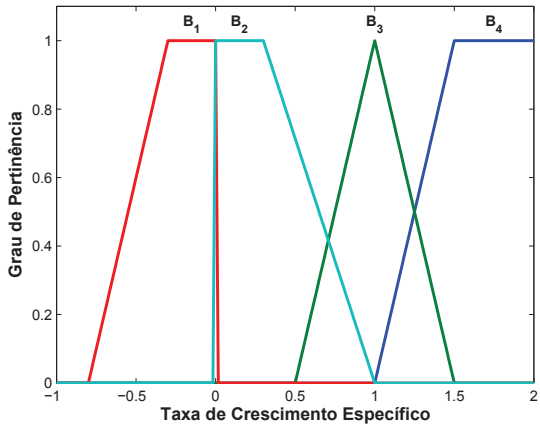

Figura 3.6: Funções de pertinência de saída do modelo de Verhulst.

Os parâmetros adotados para o modelo de Verhulst neste trabalho são a razão de crescimento $r = 0.01785$ e o valor limite da população $P_\infty = 256$. Além disso, especificamos que o caso tratado, considera-se $P_0 < P_\infty$, ou seja, a população P é crescente.

As Figuras 3.5 e 3.6 representam, respectivamente, as funções de pertinência de entrada e saída para o modelo logístico de Verhulst. Para A_1, A_6, B_5 e B_6 temos que as funções de

pertinência são do tipo trapezoidais e para A_2, A_3, A_4, A_5, B_1, B_2, B_3 e B_4 as funções de pertinência são triangulares.

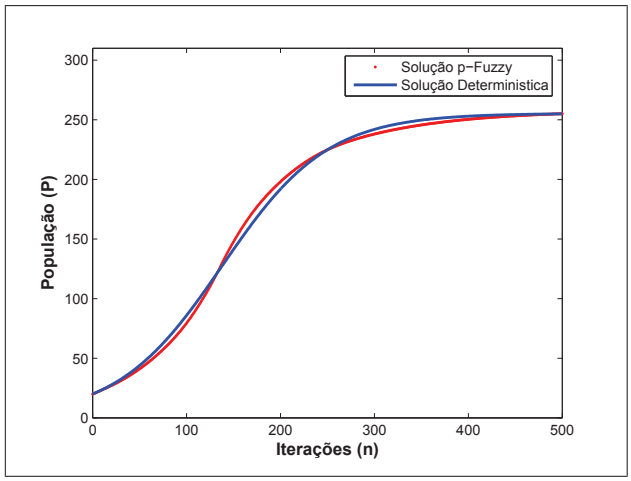

Figura 3.7: Solução do sistema p-fuzzy com população inicial $P_0 = 20$, $P_\infty = 256$ e iterações $n = 500$.

O método numérico escolhido para obter a aproximação do sistema p-fuzzy para a equação diferencial ordinária (3.14) é o método de Euler e, assim como no modelo malthusiano, calculamos o erro da solução p-fuzzy em relação a solução analítica com os dois critérios dados pelas expressões (3.10) e (3.11). Os valores encontrados foram $E_1 = 6.5285$ e $E_2 = 0.0256075$. Analisando a Figura 3.7 que apresenta as soluções analítica e fuzzy, e considerando também os valores dos erros E_1 e E_2, podemos dizer que o sistema p-fuzzy é uma boa aproximação para a solução obtida analiticamente a partir da equação diferencial ordinária.

3.3.3 Modelo de Montroll

A maioria dos modelos de crescimento populacional, formalizados a partir de equações diferenciais ordinárias, tem a forma $\frac{1}{P}\frac{dP}{dt} = f(P)$. A função f é denominada crescimento específico da população [2].

Para o modelo de Montroll, temos que

$$f(P) = a\left(1 - \left(\frac{P}{P_\infty}\right)^q\right), \tag{3.16}$$

em que $q > 0$. O valor do parâmetro q é o indicador da posição de ponto de inflexão da curva.

Quando $q = 1$, a equação (3.16) é simplesmente o modelo de Verhulst. Podemos ver os diferentes comportamentos para o modelo, em função dos valores de q, na Figura 3.8.

Devemos encontrar a solução analítica para a equação 3.16. Então temos que:

$$\frac{dP}{dt} = aP\left[1 - \left(\frac{P}{P_\infty}\right)^q\right] \tag{3.17}$$

em que $a > 0$, $q > 0$ e $P(0) = P_0$.

Dessa forma, $\frac{dP}{dt} = aP - \frac{a}{P_\infty^q}P^{q+1}$, ou seja, $\frac{dP}{dt} - aP = -\frac{a}{(P_\infty)^q}P^{q+1}$. Chamando $(q+1) = n$ e $\frac{a}{(P_\infty)^q} = m$, temos que $\frac{dP}{dt} - aP = -mP^n$, que é a equação de Bernoulli, considerando $w = P^{1-n} = P^{-q}$, então:

$$\frac{dw}{dt} + (1-n)(-a)w = (1-n)(-m) \Rightarrow \frac{dw}{dt} + qaw = qm. \tag{3.18}$$

Resolvendo a equação (3.18) utilizando o fator de integração de $u(t) = e^{qat}$.

Então, $w(t) = \frac{m}{a} + \frac{qmc}{e^{qat}}$, em que c é uma constante. Utilizando a condição inicial e com $w = p^{-q}$, temos a solução analítica para a equação (3.17), que é dada por

$$P(t) = P_\infty \left[\frac{e^{qat}}{e^{qat} + \left(\frac{P_\infty}{P_0}\right)^q - 1}\right]^{1/q}. \tag{3.19}$$

Assim, como no sistema p-fuzzy do modelo de Verhulst, o método numérico utilizado foi o método de Euler e a arquitetura do sistema pode ser vista na Figura 3.1.

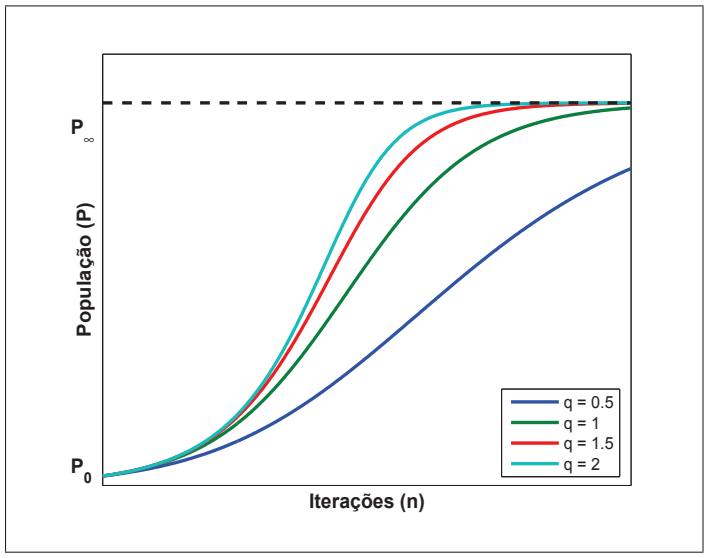

Figura 3.8: Soluções para o modelo de Montroll para diversos valores de q com pupulação inicial $P_0 = 20$ e número de iterações $n = 500$.

A base de regras para o modelo de Montroll é dada por:

R_1**:** Se a população P é *muito baixa* (A_1) então a taxa de crescimento específico $\frac{1}{P}\frac{dP}{dt}$ é *baixa positiva* (B_2);

R_2**:** Se a população P é *baixa* (A_2) então a taxa de crescimento específico $\frac{1}{P}\frac{dP}{dt}$ é *média positiva* (B_3);

R_3**:** Se a população P é *média* (A_3) então a taxa de crescimento específico $\frac{1}{P}\frac{dP}{dt}$ é *alta positiva* (B_4);

R_4**:** Se a população P é *média alta* (A_4) então a taxa de crescimento específico $\frac{1}{P}\frac{dP}{dt}$ é *média positiva* (B_3);

R_5**:** Se a população P é *alta* (A_5) então a taxa de crescimento específico $\frac{1}{P}\frac{dP}{dt}$ é *baixa positiva* (B_2);

R_6**:** Se a população P é *altíssima* (A_6) então a taxa de crescimento específico $\frac{1}{P}\frac{dP}{dt}$ é *baixa negativa* (B_1).

onde as funções de pertinência são trapezoidais para A_1, A_6 e B_4, e triangulares para A_2, A_3, A_4, A_5, B_1, B_2 e B_3.

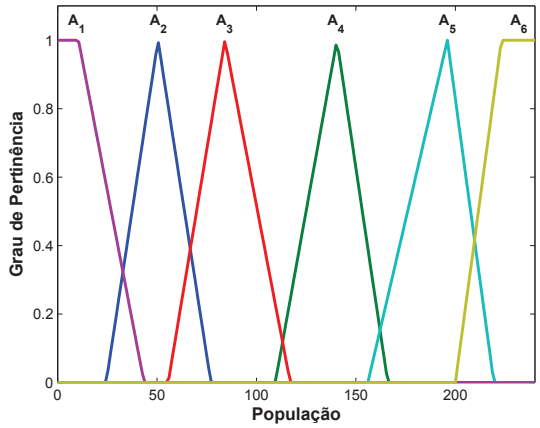

Figura 3.9: Funções de pertinência de entrada do modelo de Montroll.

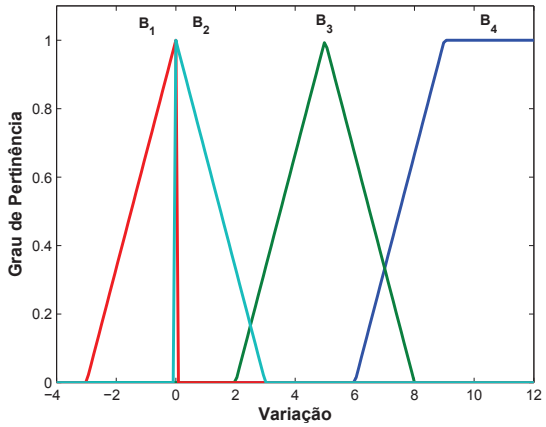

Figura 3.10: Funções de pertinência de saída do modelo de Montroll.

Temos que as Figuras 3.9 e 3.10 representam, respectivamente, as funções de pertinência de entrada e saída para o modelo de Montroll.

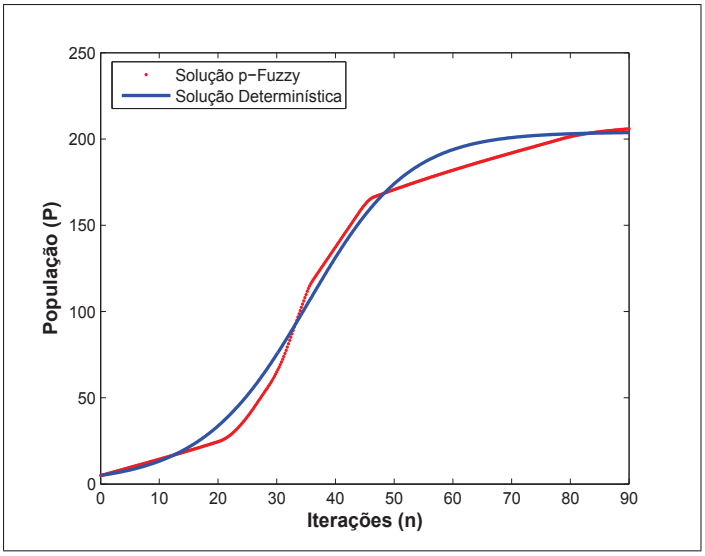

Figura 3.11: Solução do sistema p-fuzzy com população inicial $P_0 = 20$ e número de iterações $n = 500$.

A Figura 3.11 apresenta as soluções analítica e fuzzy para o sistema de equações diferenciais ordinárias que determinam o modelo de Montroll. Foi considerado neste caso $q = 1.2$, os valores para a taxa de crescimento relativa $a = 0.1$ e para o valor limite da população $P_\infty = 204$.

O erro de aproximação da solução p-fuzzy para a solução analítica foi calculado utilizando os critérios das expressões (3.10) e (3.11) e os erros obtidos foram $E_1 = 12.6393$ e $E_2 = 2.5278$. Portanto, como pode-se observar no gráfico e pelos cálculos dos erros que a diferença entre as soluções analítica e fuzzy é pequena, fazendo com que o sistema p-fuzzy seja, para esta solução analítica, uma boa aproximação.

3.3.4 Modelo de Transferência da População HIV Assintomática para Sintomática

A Síndrome da Imunodeficiência Adquirida (AIDS) tornou-se um problema mundial de saúde. É uma síndrome proveniente de um processo que compromete o sistema imunológico decorrente de infecção pelo HIV (Vírus de Imunodeficiência Humana). O objetivo é trabalhar utilizando um sistema baseado em regras fuzzy para elaborar um modelo de conversão de uma população HIV assintomática para uma sintomática, sem tratamento com antiretrovirais.

O modelo estudado é baseado no modelo clássico proposto por Anderson (1986) [1], o qual estabelece que a taxa de conversão (γ) da infecção para a AIDS está em função do tempo. Na história natural do HIV, seria a transferência entre a fase assintomática e sintomática $x+y = 1$.

Este modelo é descrito por

$$\begin{aligned} \frac{dx}{dt} &= -\gamma(t)x & x(0) = 1 \\ \frac{dy}{dt} &= \gamma(t)x = \gamma(t)(1-y) & y(0) = 0. \end{aligned} \tag{3.20}$$

em que x (assintomática) representa a fração de indivíduos infectados, mas que ainda não desenvolveram a doença, enquanto que y (sintomática) representa a fração de indivíduos infectados que já desenvolveram a doença. O modelo p-fuzzy é construído a partir de um sistema baseado em regras fuzzy, em que a entrada é a fração da população sintomática (y) e a saída é a variação da população sintomática $\left(\frac{dy}{dt}\right)$ [10].

As funções de pertinência de entrada e saída podem ser vistas, respectivamente, nas Figuras 3.12 e 3.13.

A base de regras utilizada é

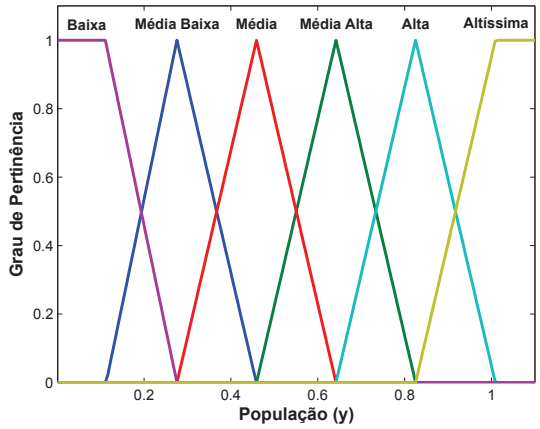

Figura 3.12: Funções de pertinência de entrada do modelo da AIDS.

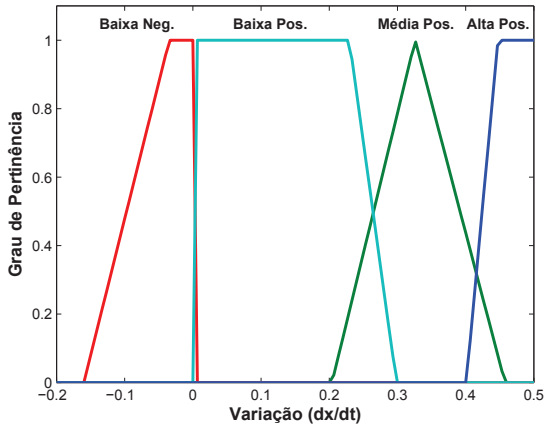

Figura 3.13: Funções de pertinência de saída do modelo da AIDS.

r_1**:** Se a população y é *baixa*, então a variação $\dfrac{dy}{dt}$ é *média positiva*;

r_2**:** Se a população y é *média baixa*, então a variação $\dfrac{dy}{dt}$ é *média positiva*;

r_3**:** Se a população y é *média*, então a variação $\dfrac{dy}{dt}$ é *alta positiva*;

r_4**:** Se a população y é *alta*, então a variação $\dfrac{dy}{dt}$ é *baixa*;

r_5**:** Se a população y é *média alta*, então a variação $\dfrac{dy}{dt}$ é *baixa positiva*;

r_6**:** Se a população y é *altíssima*, então a variação $\dfrac{dy}{dt}$ é *baixa negativa*.

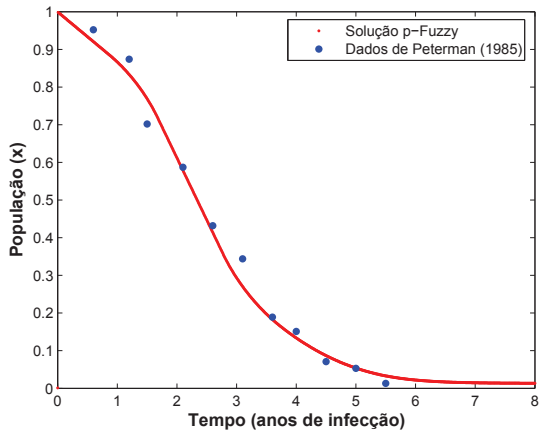

Figura 3.14: Evolução no tempo da população HIV assintomática.

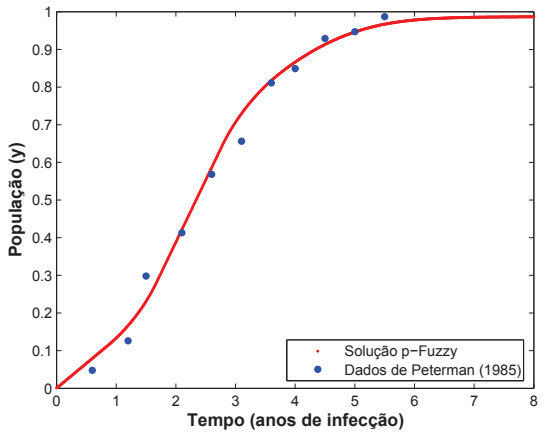

Figura 3.15: Evolução no tempo da população HIV sintomática.

As simulações numéricas apresentadas nas Figuras 3.14 e 3.15 são realizadas a partir de uma fração inicial da população sintomática $y(t_0)$, por integração numérica, utilizando a Regra

do Trapézio, e repetindo o processo para 800 iterações obtemos o comportamento de uma população HIV sintomática. A população assintomática ($x(t)$) é calculada por meio da equação $x(t) = 1 - y(t)$.

As Figuras 3.14 e 3.15 mostram que as curvas obtidas pelo modelo p-fuzzy estão próximas dos dados de Peterman (1985) [19], que foram apresentados no modelo clássico proposto por Anderson (1986) [1] para a população sintomática.

Neste modelo, calculamos o erro de aproximação entre o sistema p-fuzzy e os dados de Peterman por meio dos critérios dados nas expressões (3.10) e (3.11), obtendo os valores $E_1 = 0.07339$ e $E_2 = 0.9238$, ou seja, o sistema p-fuzzy é uma boa aproximação para a solução analítica, dada pelos dados de Peterman (1986).

Assim, concluímos que seguindo um sistema baseado em regras fuzzy podemos ter o modelo do comportamento da população HIV assintomática e sintomática, sem o uso de equações diferenciais.

3.3.5 Modelo Presa-Predador de Lotka-Volterra

Por volta de 1925, Alfred J. Lotka e Vito Volterra desenvolveram um dos modelos matemáticos de mais largo uso e de destacada importância para representar interações entre presas e seus predadores. Tal modelo é conhecido como Modelo Presa-Predador de Lotka-Volterra. Criado por Volterra (1925), foi bem aceito por explicar as alterações observadas nas populações de pescadas e tubarões no Mar Adriático, por ocasião da paralisação de atividades pesqueiras devido à I Guerra Mundial e posteriormente retomada, quando do término da guerra [15] e [26]. O modelo presa-predador clássco de Lotka-Volterra pressupõe que:

1. Tanto as presas como os predadores estão distribuídos uniformemente num mesmo habitat, ou seja, todos os predadores têm a mesma chance de encontrar cada presa;

2. O encontro entre os elementos das duas espécies seja ao acaso, a uma taxa proporcional ao tamanho das duas populações, já que quanto maior o número de presas, mais fácil será encontrá-las e quanto mais predadores, maior o número de ataques;

3. A população de presas cresce exponencialmente na ausência de predadores (crescimento ilimitado por escassez de predadores);

4. A população de predadores decresce exponencialmente na ausência de presas (decréscimo por escassez de alimento);

5. A população de predadores é favorecida pela abundância das presas;

6. A população de presas é desfavorecida pelo aumento de predadores.

Estas seis hipóteses (ver [3]) em relação ao modelo são resumidas nas equações abaixo, denominadas Modelo de Lotka-Volterra:

$$\begin{cases} \dfrac{dx}{dt} = ax - \alpha xy \\ \\ \dfrac{dy}{dt} = -by + \beta xy \end{cases} . \tag{3.21}$$

As variáveis de estado x e y são, respectivamente, quantidade de presas e quantidade de predadores em cada instante t. As hipóteses do modelo podem ser encontradas em [2]. Os parâmetros representam:

- a : taxa de crescimento da população de presas;
- α : proporção de sucesso dos ataques dos predadores as presas;
- β : taxa de conversão de biomassa das presas em predadores;
- b : taxa de mortalidade de predadores na ausência de presas.

As hipóteses de Lotka-Volterra caracterizam um modelo presa-predador cujos contingentes populacionais oscilam com o tempo. O plano de fase referente ao modelo de Lotka-Volterra, é apresentado na 3.16.

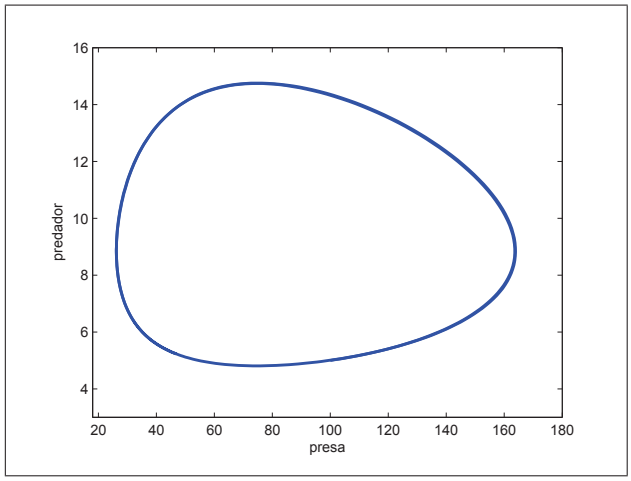

Figura 3.16: Plano de fases do modelo presa-predador de Lotka-Volterra com condições iniciais $x_0 = 100$ e $y_0 = 5$.

Temos como objetivo elaborar uma base de regras que substitua as equações (3.21), para modelar a dinâmica entre as presas e os predadores, por meio de um modelo p-fuzzy de Lotka-Volterra. A arquitetura do sistema p-fuzzy para o modelo de Lotka-Volterra pode ser visto na Figura 3.17.

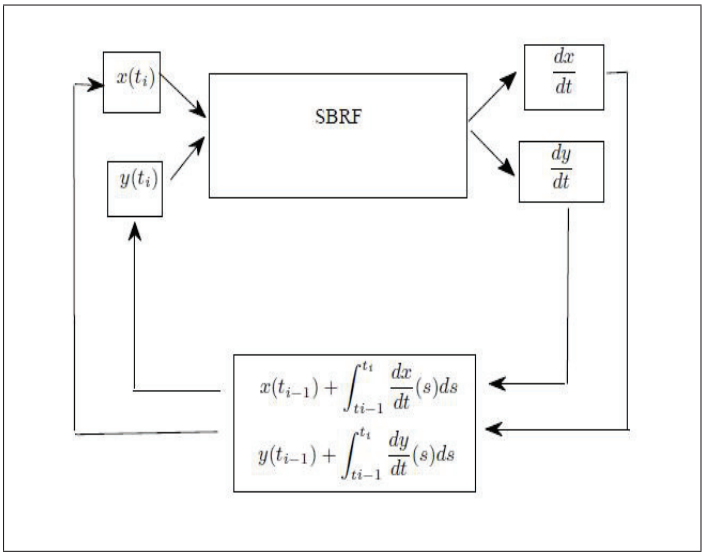

Figura 3.17: Arquitetura do sistema p-fuzzy para o modelo presa-predador de Lotka-Volterra.

As variáveis linguísticas de entrada são a quantidade de presas (x) e quantidade de predadores (y); as variáveis de saída são a variação relativa da quantidade de presas por unidade de tempo, denotada por $\frac{1}{x}\frac{dx}{dt}$; e a variação relativa da quantidade de predadores por unidade de tempo, denotada por $\frac{1}{y}\frac{dy}{dt}$ [2].

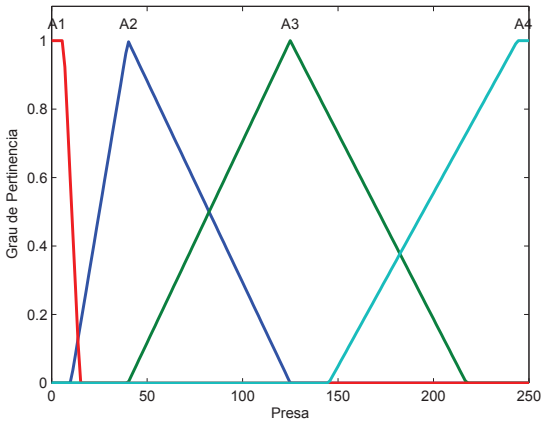

Figura 3.18: Funções de Pertinência para as Presas (x).

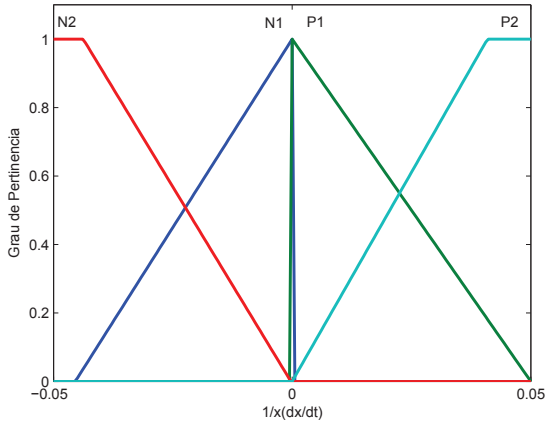

Figura 3.19: Funções de Pertinência para $\frac{1}{x}\frac{dx}{dt}$.

As Figuras 3.18 e 3.19 representam, respectivamente, as funções de pertinência para as variáveis de entrada e de saída para as presas (população e variação da população), enquanto

que as Figuras 3.20 e 3.21 representam, respectivamente, as funções de pertinência das variáveis de entrada e saída para os predadores (população e variação da população).

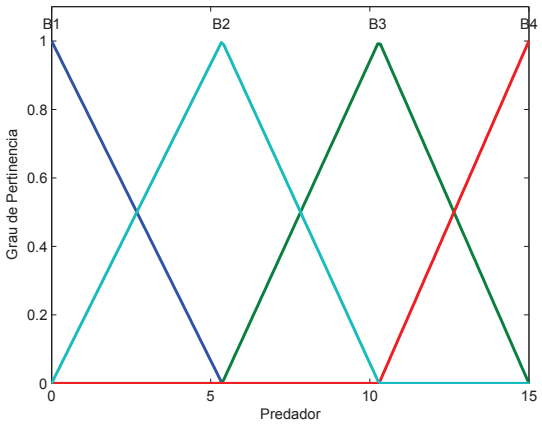

Figura 3.20: Funções de Pertinência para os Predadores (y).

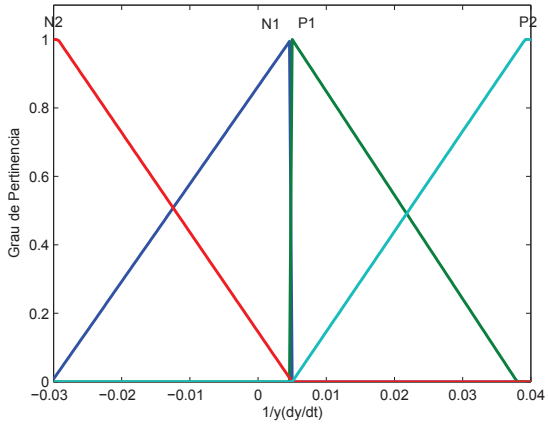

Figura 3.21: Funções de Pertinência para $\frac{1}{y}\frac{dy}{dt}$.

Os termos linguísticos da variável x são: *baixa* (A_1), *média baixa* (A_2), *média alta* (A_3) e *alta* (A_4) e da variável y são *baixa* (B_1), *média baixa* (B_2), *média alta* (B_3) e *alta* (B_4). Para os crescimentos específicos, $\frac{1}{x}\frac{dx}{dt}$ e $\frac{1}{y}\frac{dy}{dt}$ os termos linguísticos são *alto positivo* (P_2), *baixo positivo* (P_1), *baixo negativo* (N_1) e *alto negativo* (N_2) [2].

A base de regras é dada por:

- Se x é A_1 e y é B_1, então $\frac{1}{x}\frac{dx}{dt}$ é P_2 e $\frac{1}{y}\frac{dy}{dt}$ é N_2
- Se x é A_2 e y é B_1, então $\frac{1}{x}\frac{dx}{dt}$ é P_2 e $\frac{1}{y}\frac{dy}{dt}$ é N_1
- Se x é A_3 e y é B_1, então $\frac{1}{x}\frac{dx}{dt}$ é P_2 e $\frac{1}{y}\frac{dy}{dt}$ é P_1
- Se x é A_4 e y é B_1, então $\frac{1}{x}\frac{dx}{dt}$ é P_2 e $\frac{1}{y}\frac{dy}{dt}$ é P_2
- Se x é A_1 e y é B_2, então $\frac{1}{x}\frac{dx}{dt}$ é P_1 e $\frac{1}{y}\frac{dy}{dt}$ é N_2
- Se x é A_2 e y é B_2, então $\frac{1}{x}\frac{dx}{dt}$ é P_1 e $\frac{1}{y}\frac{dy}{dt}$ é N_1
- Se x é A_3 e y é B_2, então $\frac{1}{x}\frac{dx}{dt}$ é P_1 e $\frac{1}{y}\frac{dy}{dt}$ é P_1
- Se x é A_4 e y é B_2, então $\frac{1}{x}\frac{dx}{dt}$ é P_1 e $\frac{1}{y}\frac{dy}{dt}$ é P_2

- Se x é A_1 e y é B_3, então $\frac{1}{x}\frac{dx}{dt}$ é N_1 e $\frac{1}{y}\frac{dy}{dt}$ é N_2
- Se x é A_2 e y é B_3, então $\frac{1}{x}\frac{dx}{dt}$ é N_1 e $\frac{1}{y}\frac{dy}{dt}$ é N_1
- Se x é A_3 e y é B_3, então $\frac{1}{x}\frac{dx}{dt}$ é N_1 e $\frac{1}{y}\frac{dy}{dt}$ é P_1
- Se x é A_4 e y é B_3, então $\frac{1}{x}\frac{dx}{dt}$ é N_1 e $\frac{1}{y}\frac{dy}{dt}$ é P_2
- Se x é A_1 e y é B_4, então $\frac{1}{x}\frac{dx}{dt}$ é N_2 e $\frac{1}{y}\frac{dy}{dt}$ é N_2
- Se x é A_2 e y é B_4, então $\frac{1}{x}\frac{dx}{dt}$ é N_2 e $\frac{1}{y}\frac{dy}{dt}$ é N_1
- Se x é A_3 e y é B_4, então $\frac{1}{x}\frac{dx}{dt}$ é N_2 e $\frac{1}{y}\frac{dy}{dt}$ é P_1
- Se x é A_4 e y é B_4, então $\frac{1}{x}\frac{dx}{dt}$ é N_2 e $\frac{1}{y}\frac{dy}{dt}$ é P_2

A partir do sistema baseado em regras fuzzy, determinamos o valor de $\frac{1}{x}\frac{dx}{dt}$ no próximo instante. Calculamos o próximo valor de x com integração numérica. Especificamente, utilizamos o método de Euler para obter o comportamento das populações ao longo do tempo, Figura 3.22.

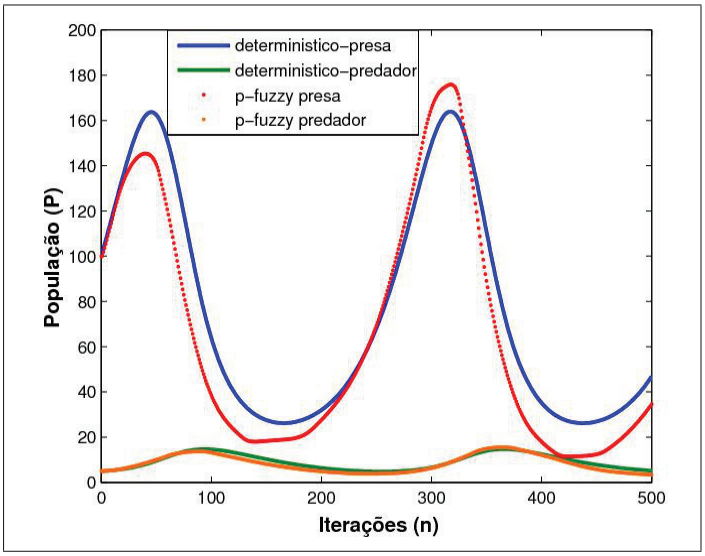

Figura 3.22: Evolução das populações no tempo por meio do sistema determinístico e p-fuzzy com condições iniciais $x_0 = 100$, $y_0 = 5$ e iterações $n = 460$.

Para o modelo presa-predador p-fuzzy, com parâmetros fixos $a = 0.039$, $\alpha = 0.0048$, $\beta = 0.0002$ e $b = 0.015$, obtivemos os erros em relação ao modelo clássico nos valores de $E_1 = 38.1874$ e $E_2 = 5.1225$, sendo estes erros calculados segundo os critérios das equações (3.10) e (3.11). Esses valores foram obtidos da seguinte forma: foram calculados os erros para a população de presas e para a população de predadores. Em seguida esses erros foram somados, dando origem ao erro para o sistema p-fuzzy do modelo presa-predador. Com essas informações analíticas e visualizando a Figura 3.22, podemos concluir que este sistema p-fuzzy é uma boa aproximação para a solução determinística, obtida numericamente por meio de equações diferenciais ordinárias.

Na próxima seção apresentaremos um modelo de equações de diferenças para o controle de pragas.

3.3.6 Modelo do Controle de Pragas

Um país como o Brasil, com tantas peculiaridades e pluralidades climáticas e geográficas, abriga uma diversidade enorme de insetos e plantas. Além das espécies nativas e cultivadas para fins comerciais, tanto para consumo interno como para exportação, historicamente muitas espécies vegetais foram introduzidas por colonizadores e imigrantes, sendo responsáveis pela introdução de espécies exóticas de predadores fitófagos.

Quando surge uma praga em alguma lavoura, os agricultores têm se utilizado de inseticidas eficientes ao maior número de espécies e isso tem sido a tônica dos últimos 30 anos no Brasil. Atualmente, o controle de pragas é feito com uso de inseticidas, cujos efeitos colaterais, tais como a contaminação de alimentos, de mananciais, do homem, do ar e do solo, são conhecidos. A ressurgência de pragas num curto período após a aplicação do defensivo é um problema comum e ocorre devido ao vácuo biótico ocasionado pelo uso do inseticida. A praga retorna livre de seus inimigos naturais, podendo desenvolver grandes populações [4].

Inseticidas são "substâncias químicas utilizadas para matar, atrair e repelir insetos, sendo sua descoberta, isolamento, síntese, avaliação toxicológica e de impacto ambiental um vasto tópico de pesquisas no mundo inteiro e que tem se desenvolvido bastante nas últimas décadas" [4].

Especificamente, por exemplo, verificamos que alguns tipos de pulgões são os principais vetores na disseminação das doenças das plantas perenes. Recentemente, uma doença denomi-

nada “morte-súbita”, causada por vírus transportados por pulgões, tem sido a grande ameaça dos laranjais paulistas.

O controle dessas doenças é baseado, sobretudo, no controle químico com aplicações intermitentes de biocidas quase sempre sem levar em conta a infestação da praga. Proporemos um modelo para controlar uma determinada praga sem especificá-la, a seguir.

O modelo de controle será construído com base nos modelos de densidade populacional p-fuzzy, descrito na equação:

$$\begin{cases} x_{k+1} = (x_k + \Delta_x(x_k)) \times (1 - C(x_k, \Delta_x(x_k))) \\ x_0 \in \mathbb{R}, \end{cases} \tag{3.22}$$

em que x_k é a população no instante k, $\Delta_x(x_k)$ é a variação populacional e $C(x_k, \Delta_x(x_k))$ é a porcentagem da população de pragas que morre após aplicação do biocida.

O valor $(1 - C(x_k, \Delta_x(x_k))) \in [0, 1]$ representa a porcentagem da população que sobreviverá à aplicação [21].

Este modelo apresenta um sistema discreto. Para tal sistema, temos que, a Figura 3.23 representa esquematicamente o seu funcionamento.

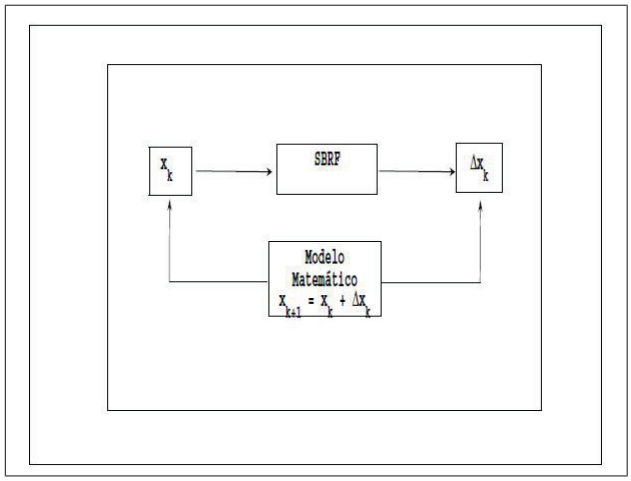

Figura 3.23: Arquitetura de um modelo p-fuzzy discreto.

Construímos dois SBRF, o primeiro é um sistema cuja variável de entrada é x_k e a de saída é $\Delta_x(x_k)$. O segundo SBRF, para o controle de pragas, utiliza como entrada o par $(x_k, \Delta_x(x_k))$, em que $\Delta_x(x_k)$ é dado pelo sistema anterior e a saída $C(x_k, \Delta_x(x_k))$. Após 15 iterações, com $x_0 = 20$, o SBRF aciona o controle e mantém a densidade populacional das pragas a um nível

adequado. A arquitetura dos sistemas pode ser vista na Figura 3.24.

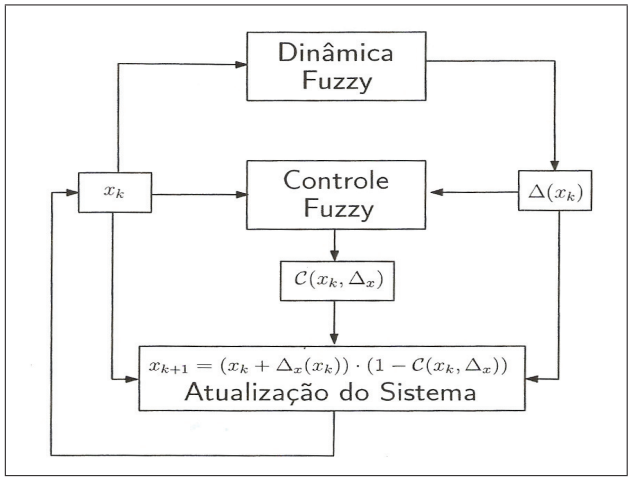

Figura 3.24: Arquitetura do sistema p-fuzzy para o modelo do controle de pragas [21].

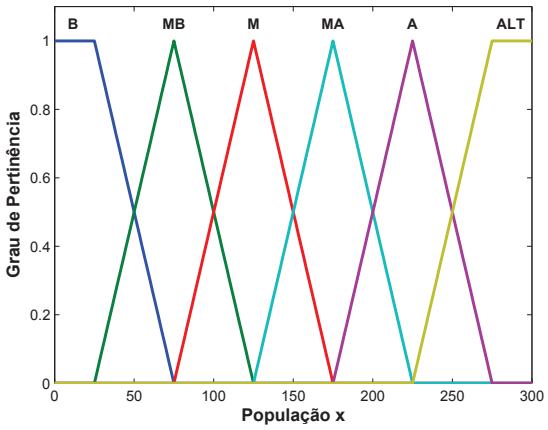

Figura 3.25: Funções de pertinência de entrada para o SBRF que produz a variação da população.

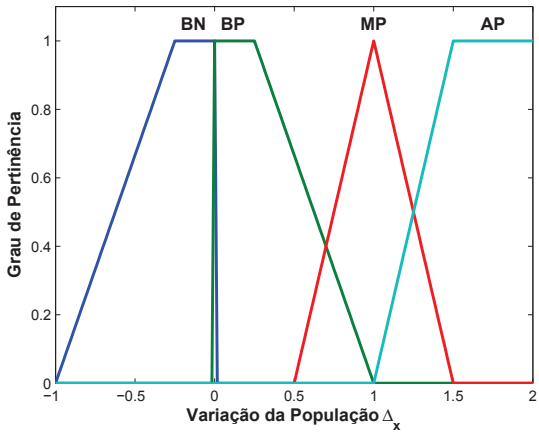

Figura 3.26: Funções de pertinência de saída para o SBRF que produz a variação da população.

O primeiro SBRF tem como variável de entrada a população no instante k, x_k e variável de saída a variação da população $\Delta_x(x_k)$. As funções de pertinência para as variáveis de entrada e saída desse SBRF estão representadas, respectivamente, nas Figuras 3.25 e 3.26.

Para este SBRF, temos a seguinte base de regras:

r_1**:** Se a população x_k é *baixa* (B), então a variação $\Delta_x(x_k)$ é *baixa positiva* (BP);

r_2**:** Se a população x_k é *média baixa* (MB), então a variação $\Delta_x(x_k)$ é *média positiva* (MP);

r_3**:** Se a população x_k é *média* (M), então a variação $\Delta_x(x_k)$ é *alta positiva* (AP);

r_4**:** Se a população x_k é *média alta* (MA), então a variação $\Delta_x(x_k)$ é *média positiva* (MP);

r_5**:** Se a população x_k é *alta* (A), então a variação $\Delta_x(x_k)$ é *baixa positiva* (BP);

r_6**:** Se a população x_k é *altíssima* (ALT), então a variação $\Delta_x(x_k)$ é *baixa negativa* (BN).

O segundo SBRF tem como variáveis de entrada a população no instante k, x_k e a variação da população Δ_x e a variável de saída é o controle $C(x_k, \Delta_x(x_k))$.

A Figura 3.27 representa a função de pertinência de saída para o segundo SBRF.

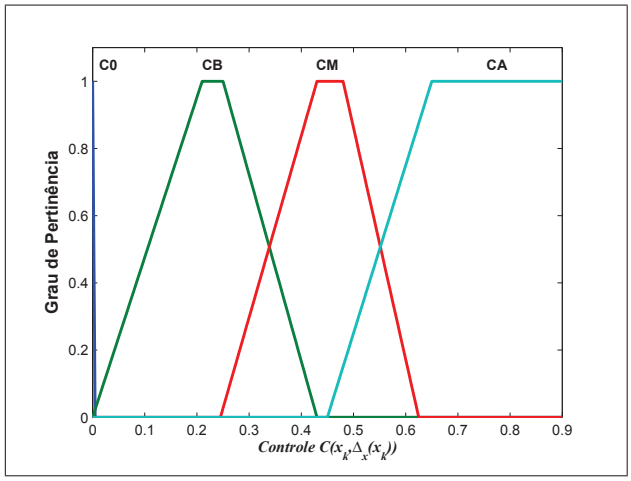

Figura 3.27: Funções de pertinência de saída para o SBRF que produz o controle das pragas.

Para o segundo SBRF temos a seguinte base de regras:

x/Δ_x	baixa negativa (BN)	baixa positiva (BP)	média positiva (MP)	alta positiva (AP)
baixa (B)	C_0	C_0	C_0	C_B
média baixa (MB)	C_0	C_0	C_B	C_M
média (M)	C_0	C_B	C_M	C_M
média alta (MA)	C_B	C_M	C_A	C_A
alta (A)	C_M	C_M	C_A	C_A
altíssima (ALT)	C_A	C_A	C_A	C_A

A Figura 3.28 representa a evolução da praga com e sem controle. Pode-se verificar que aplicando o biocida, a população de pragas não ultrapassa o valor de 40, considerando a população no intervalo $[0, 300]$. Dessa forma, pode-se dizer que o modelo de controle atinge o resultado esperado, visto que há um controle visível das pragas com controle fuzzy.

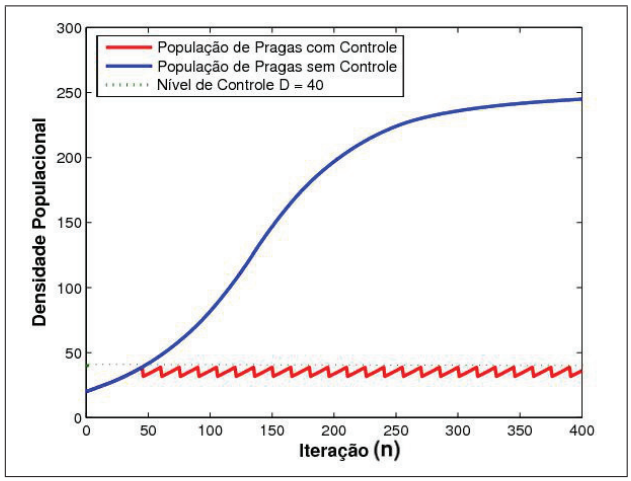

Figura 3.28: Dinâmica populacional do SBRF que produz o controle de pragas, com condição inicial $x_0 = 20$ e número de iterações $n = 400$.

Para todos os modelos citados neste capítulo foram desenvolvidos sistemas p-fuzzy para melhor aproximação para a solução determinística ou, como no caso do controle de pragas, a proposta de se ter um modelo com controle fuzzy, que proporciona um controle maior das pragas com a aplicação de biocida.

No próximo capítulo estudaremos os sistemas p-fuzzy modificados segundo a teoria dos modificadores linguísticos, ou seja, as funções de pertinência de cada um dos sistemas serão modificados de modo a aproximar ainda mais a solução fuzzy da solução analítica, ou para o caso do controle de pragas, poder ter um controle ainda maior da praga.

Capítulo 4

Sistemas p-Fuzzy Modificados e p-Fuzzy Modificados no Tempo

Estudamos nos capítulos anteriores os SBRF, os sistemas p-fuzzy e observamos que um dos componentes mais importantes de ambos os sistemas são as funções de pertinência dos conjuntos fuzzy que representam as variáveis de entrada e saída de cada sistema. As funções de pertinência mais utilizadas até agora foram as triangulares e trapezoidais e ambas são compostas por retas crescentes e decrescentes.

A partir deste capítulo estudaremos os sistemas modificados, ou seja, serão feitas modificações nas funções de pertinência de cada modelo tratado no Capítulo 4. Tais modificações alteram as funções de pertinência. As funções triangulares, por exemplo, tomam outra forma após serem modificadas. Portanto, toda a base de regras é modificada e isso faz com que o programa para calcular a "nova"solução seja alterado. No trabalho, desenvolvemos rotinas que "incluem"essas alterações no toolbox fuzzy do software Matlab.

No Apêndice A encontram-se os códigos e as rotinas dessa metodologia e no Apêndice B estão as rotinas de cada um dos modelos aqui estudados.

4.1 Modificadores Linguísticos

Como o próprio nome sugere, modificadores linguísticos são frequentemente utilizados para alterar atributos, ou seja, modelar advérbios. A teoria dos conjuntos fuzzy dá-nos a representação de subconjuntos fuzzy que representam atributos de variáveis linguísticas. Neste caso, os modificadores linguísticos são denominados modificadores fuzzy.

Definição 4.1 *Um* modificador fuzzy *m sobre U é uma aplicação definida em $\mathcal{F}(U)$ com valores em $\mathcal{F}(U)$:*

$$m : \mathcal{F}(U) \to \mathcal{F}(U) \tag{4.1}$$

onde $\mathcal{F}(U)$ é a classe dos subconjuntos fuzzy de U [2].

Os principais modificadores fuzzy são:

1. *Expansivo* se, para todo $A \in \mathcal{F}(U)$, $A \subseteq m(A)$, ou seja, $\varphi_A(x) \leq \varphi_{m(A)}(x)$;

2. *Restritivo* se, para todo $A \in \mathcal{F}(U)$, $A \supseteq m(A)$, ou seja, $\varphi_A(x) \geq \varphi_{m(A)}(x)$.

Os modificadores fuzzy mais usados são do tipo potência.

Definição 4.2 *Um modificador é do tipo potência se para cada $A \in \mathcal{F}(U)$ tem-se $u_{m(A)}(x) := (u_A(x))^s$, para algum $s \in [0, \infty)$ [2].*

Podemos observar que $0 < s < 1$ então m_S é *expansivo* e se $s > 1$ então m_S é *restritivo*, já que $\varphi_A(x) \in [0, 1]$.

Exemplo 4.1 *Consideremos o conjunto fuzzy dos indivíduos "jovens"definido pela função de pertinência*

$$\varphi_J(x) = \begin{cases} 1 & \text{se } x \leq 25 \\ \left(1 + \dfrac{x - 25}{5}\right)^{-2} & \text{se } x > 25 \end{cases} \tag{4.2}$$

A função de pertinência dada para os indivíduos "jovens"pode ser vista na Figura 4.1.

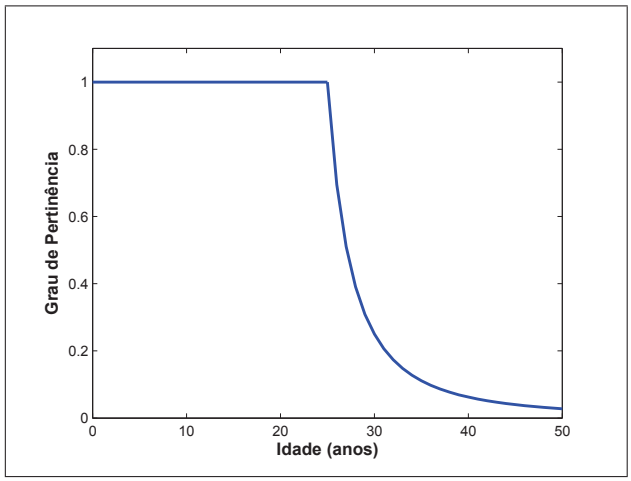

Figura 4.1: Função de pertinência para os indivíduos "jovens".

Quando aplicamos modificadores fuzzy em termos primários como o adjetivo "jovem", definimos novos termos fuzzy como "muito jovem", por exemplo. Assim, se tomarmos para "muito jovem"o subconjunto fuzzy MJ, cuja função de pertinência é dada por

$$\varphi_{MJ}(x) = \varphi_{m(J)}(x) = (\varphi_J(x))^2, \tag{4.3}$$

teremos o modificador $m(A) = (A)^2$ e, para um indivíduo cuja idade é $x = 30$, seu grau de pertinência ao conjunto dos "jovens"é $\varphi_J(30) = 0,25$ enquanto que, para o conjunto modificado dos "muito jovens"temos $\varphi_{MJ}(30) = 0,25^2 = 0,0625 < \varphi_J(30)$ [2]. Na Figura 4.2 podemos observar o comportamento de um modificador do tipo restritivo, em comparação a função de pertinência não-modificada.

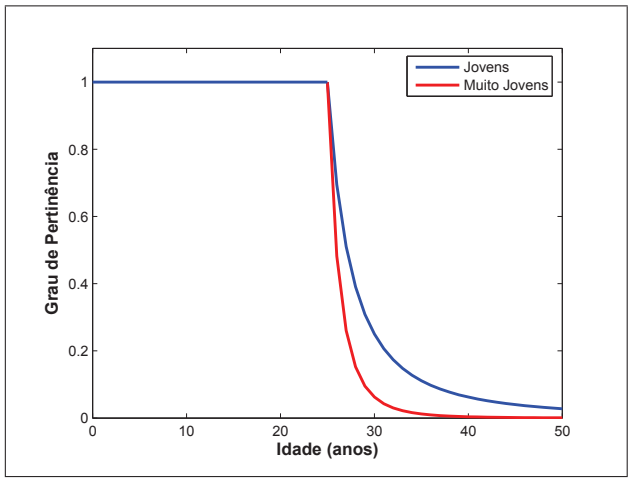

Figura 4.2: Função de pertinência para os indivíduos "muito jovens".

Também poderíamos modelar o conjunto dos "pouco jovens", utilizando a potência $\frac{1}{2}$, ao invés da potência 2. Neste caso, teríamos um modificador expansivo e, para um indivíduo cuja idade é $x = 30$ o grau de pertinência ao conjunto dos "pouco jovens"é $\varphi_{PJ}(30) = 0,25^{\frac{1}{2}} = 0,5 > \varphi_J(30)$. A representação deste conjunto pode ser vista na Figura 4.3. Nesta figura, podemos ver a diferença entre os modificadores do tipo restritivo e expansivo.

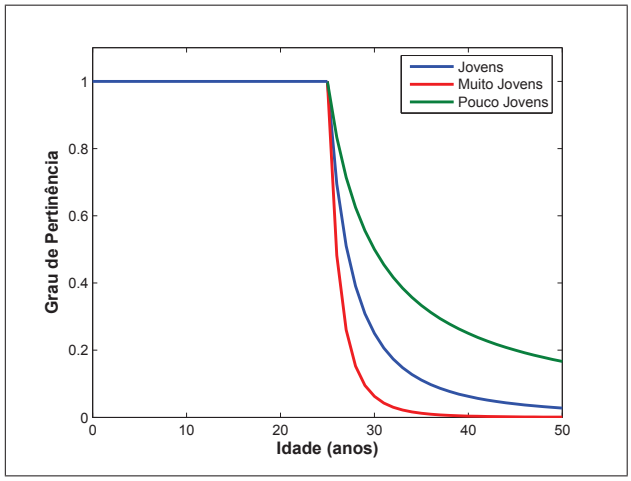

Figura 4.3: Função de pertinência para os indivíduos "jovens", "muito jovens"e "pouco jovens".

Para estudos mais aprofundados, o leitor pode consultar [6] e [13].

4.2 Modificação das Funções de Pertinência no Matlab

Para todos os modelos estudados, as simulações foram feitas no programa Matlab, utilizando o toolbox *fuzzy*. No toolbox, existe a programação das funções de pertinência com potência um, nesses programas foram feitas modificações, necessárias para trabalhar com as funções de pertinência elevadas a potências diferentes de um.

Descreveremos a seguir as alterações feitas, utilizando como exemplo a função de pertinência triangular, do toolbox fuzzy que é denominada *trimf*.

4.2.1 Funções de Pertinência Triangulares Modificadas

Para que essa função seja programada, são considerados três parâmetros $[a\ b\ c]$, como entrada da função de pertinência triangular. Esses pontos determinam os vértices para o gráfico da função de pertinência triangular. A função de pertinência citada pode ser vista na Figura 4.4.

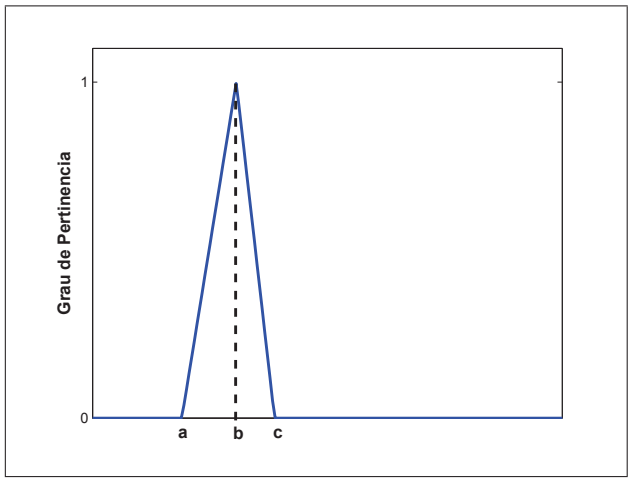

Figura 4.4: Função de pertinência triangular com parâmetros $[a\ \ b\ \ c]$.

Para modificarmos essa função, foi inserido mais um parâmetro para a função de pertinência triangular na base do toolbox fuzzy, ou seja, agora a função deve ter quatro parâmetros de entrada $[a\ \ b\ \ c\ \ d]$, onde esse novo parâmetro d representa a potência que será elevada a função de pertinência. Considerando $d = 0.5$, podemos ver a função de pertinência, que denominamos *trimfquadrado* na Figura 4.5. Escolhemos o valor $d = 0.5$ apenas para ilustrar a modificação no programa.

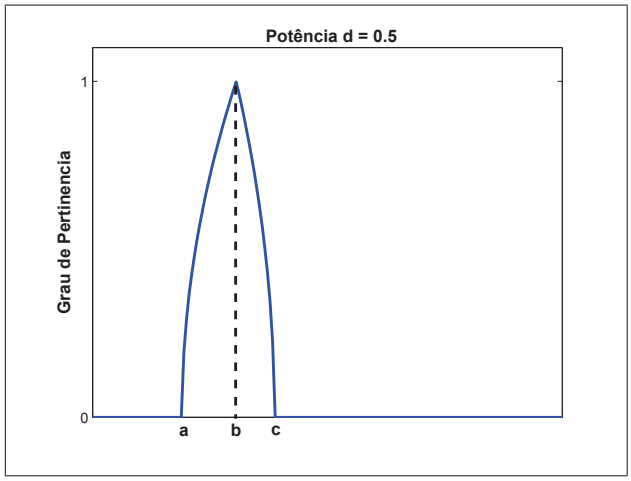

Figura 4.5: Função de pertinência triangular modificada pela potência $d = 0.5$.

Em seguida apresentamos as modificações das funções de pertinência trapezoidais.

4.2.2 Funções de Pertinência Trapezoidais Modificadas

Agora, para a função de pertinência do tipo trapezoidal, denominada *trapmf*, consideramos os parâmetros $[a\ \ b\ \ c\ \ d]$ como entrada. Esses pontos determinam os vértices para o gráfico da

função de pertinência trapezoidal. Essa função de pertinência pode ser vista na Figura 4.6.

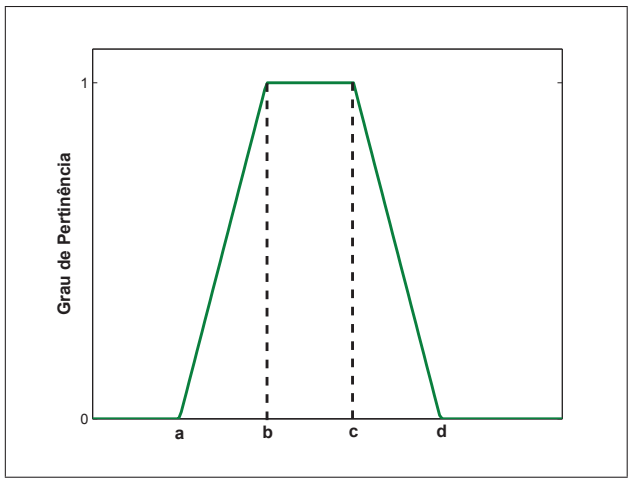

Figura 4.6: Função de pertinência trapezoidal com parâmetros $[a\ \ b\ \ c\ \ d]$.

Para modificarmos essa função, foi inserido mais um parâmetro para a função de pertinência trapezoidal na base do toolbox fuzzy, ou seja, agora a função deve ter cinco parâmetros de entrada $[a\ \ b\ \ c\ \ d\ \ e]$, onde esse novo parâmetro e representa a potência que será elevada a função de pertinência. Considerando $e = 2.0$, podemos ver a função de pertinência, que denominamos *trapmfquad* na Figura 4.7. Assim como na função de pertinência triangular modificada, o valor $e = 2.0$ foi escolhido para ilustrar didaticamente a modificação no programa.

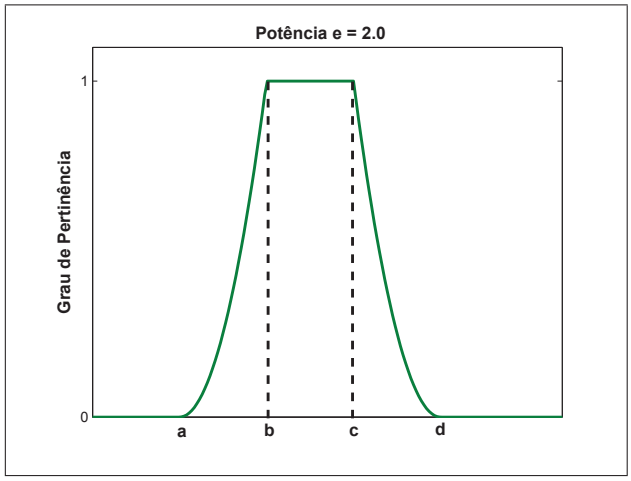

Figura 4.7: Função de pertinência trapezoidal modificada pela potência $e = 2.0$.

As funções de pertinência tanto triangulares como trapezoidais, utilizadas neste trabalho, podem ser vistas integralmente no Apêndice A.

A seguir descreveremos a metodologia para escolha do melhor valor da potência s dentre os valores considerados no intervalo $[0.1, 1.9]$.

4.3 Metodologia para Escolha da Potência e Critérios de Comparação

4.3.1 Sistemas p-Fuzzy Modificados

Será apresentado o método utilizado para a escolha da melhor potência s dentro do intervalo considerado a seguir, para os sistemas p-fuzzy modificados dos modelos populacionais descritos anteriormente. A metodologia utilizada foi a mesma para todos os modelos apresentados e o procedimento será descrito a seguir. A princípio devemos ressaltar que os sistemas p-fuzzy modificados são assim chamados por suas funções de pertinência serem modificadas por uma potência diferente de 1. Se tal potência s for igual a 1, não ocorrerão modificações, e então teremos os sistemas p-fuzzy, apresentados no Capítulo 3.

Consideramos o intervalo $I_1 = [0.1, 1.9]$ que está em torno do número 1, que significa não alteração das funções de pertinência. Dividimos I_1 com espaçamento 0.1. Dessa forma, utilizamos a rotina no Matlab para o modelo populacional considerado, obtendo então um sistema p-fuzzy modificado para cada uma das potências s $F = \{0.1, 0.2, 0.3, \ldots, 1.9\}$.

Seja x o vetor com os pontos da solução determinística e x^{s_i} o vetor com os pontos gerado pelo sistema p-fuzzy modificado através das funções de pertinência dos modelos estudados elevadas a potência s_i, onde $s_i \in F$. Determinamos os erros $E_1(s_i)$ e $E_2(s_i)$, que são considerados os erros para cada potência s_i, calculados entre o modelo determinístico e o modelo p-fuzzy modificado. Os critérios estabelecidos são os mesmos que os das equações (3.10) e (3.11), mas agora aplicado a cada uma das potências testadas. Então temos que esses erros são:

$$E_1(s_i) = max|x - x^{s_i}|, \tag{4.4}$$

$$E_2(s_i) = \frac{max|x - x^{s_i}|}{min|x^{s_i}|}. \tag{4.5}$$

Em seguida, calculamos o $\overline{E_1} = min\{E_1(s_i)\}$ e $\overline{E_2} = min\{E_2(s_i)\}$ e determinamos $\overline{s_1}$ e $\overline{s_2}$ que são as potências que geram $\overline{E_1}$ e $\overline{E_2}$, respectivamente.

Repetimos o procedimento para a potência $\overline{s_i}$ no intervalo $[\overline{s_1}-0.5, \overline{s_1}+0.5]$ e $\overline{\overline{s_i}}$ no intervalo $[\overline{s_2}-0.5, \overline{s_2}+0.5]$, com espaçamento 0.01, isto é, elevamos as funções de pertinência dos modelos estudados as potências $\overline{s_i}$ e $\overline{\overline{s_i}}$. Em seguida, determinamos a melhor potência dentre as testadas no programa.

Comparando as potências encontradas para os critérios E_1 e E_2, obtidos pelas equações (4.4) e (4.5), podemos observar em todas as simulações que utilizamos quaisquer um dos dois critérios adotados, a potência resultante como a melhor entre as calculadas, ou seja, aquela que produzia menor erro entre o modelo determinístico e o sistema p-fuzzy modificado, e em ambos os critérios a potência resultante é a mesma.

4.3.2 Sistemas p-Fuzzy Modificados no Tempo

Os sistemas p-fuzzy modificados no tempo também podem ser denominados sistemas p-fuzzy não autônomos. Neste trabalho foram feitos sistemas p-fuzzy modificados no tempo para dois dos modelos estudados: Malthus e controle de pragas.

Para o modelo p-fuzzy modificado no tempo é necessário encontrar uma função que alterasse as funções de pertinência a cada iteração. O ponto de partida para encontrar essa função foram as funções $f(i) = i$ e $f(i) = \dfrac{1}{i}$, onde $i = 1, 2, 3, \ldots$ é o número de iterações do sistema. Em seguida, foi somado a essas funções um valor a, onde $a \in [0.5, 1.5]$, sendo então as funções testadas $f(i) = a + i$ e $f(i) = a + \dfrac{1}{i}$. Devemos ressaltar que a única condição para a potência que modifica as funções de pertinência é $s \in [0, \infty)$ [2].

Os critérios para os sistemas p-fuzzy modificados no tempo serão descritos para cada um dos dois modelos estudados neste trabalho. Para o modelo de Malthus, seguimos os erros descritos pelas equações (4.4) e (4.5), mas onde a dependência era da potência s_i passa a ser agora da função f_i. Para este modelo, o procedimento é análogo.

Para o modelo do controle de pragas, ocorreram algumas alterações na metodologia. Seja x o vetor com os pontos da solução determinística e $x^{f(i)}$ o vetor com os pontos gerado pelo sistema p-fuzzy modificado no tempo através das funções de pertinência dos modelos estudados elevadas a potências que se modificavam a cada iteração conforme a função $f(i)$, onde $i = 1, 2, 3, \ldots$ é o número de iterações do sistema. Determinamos os erros $E_1(f(i))$ e $E_2(f(i))$, considerados os erros para cada função que altera as potências $f(i)$, do modelo determinístico para o modelo

p-fuzzy modificado no tempo. Os critérios estabelecidos são dados por:

$$E_1(f(i)) = max|x - x^{f(i)}|, \tag{4.6}$$

$$E_2(f(i)) = \frac{max|x - x^{f(i)}|}{max|x^{f(i)}|}. \tag{4.7}$$

É necessário esclarecer que a utilização do $max|x^{f(i)}|$ no denominador da equação (4.7) se deve ao fato de que o vetor $x^{f(i)}$ possui coordenadas nulas, e portanto não podemos dividir pelo $min|x^{f(i)}|$. Em seguida, calculamos o $\overline{E_1} = max\{E_1(f(i))\}$ e $\overline{E_2} = max\{E_2(f(i))\}$, pois para este modelo queremos saber qual a função que produz um erro maior, ou seja, qual função altera as potências de modo a se obter um maior controle da praga. A função de alteração de potências $f(i)$ que determinar os maiores valores para $\overline{E_1}$ e $\overline{E_2}$ são as escolhidas como melhor função para o sistema p-fuzzy modificado no tempo, dentre as funções testadas no trabalho.

4.4 Sistema p-Fuzzy Modificado

Um sistema é chamado de modificado quando as funções de pertinência são elevadas a potências diferentes de um. Dado um sistema baseado em regras fuzzy, ou um sistema p-fuzzy, para que esse sistema seja modificado, basta elevar suas funções de pertinência a uma potência s, onde $s \in [0, \infty)$.

A seguir, serão descritos os modelos vistos anteriormente, no Capítulo 3, mas as suas funções de pertinência serão modificadas por potência. Nosso objetivo é verificar se, após aplicada a teoria dos modificadores fuzzy (obtendo os sistemas modificados), teremos ou não uma vantagem em relação ao sistema p-fuzzy, isto é, para a maioria dos casos, iremos verificar se existe a vantagem de utilizar os modificadores do tipo potência que torne um sistema p-fuzzy modificado mais próximo a um modelo clássico, em relação a um modelo p-fuzzy sem modificações.

4.4.1 Modelo de Malthus

O sistema p-fuzzy para o modelo de Malthus produz, em relação ao modelo determinístico, os erros no valor de $E_1 = 7.2524$ e $E_2 = 3.6262$, cujos critérios são descritos pelas expressões (4.4) e (4.5).

A seguir, apresentamos a Tabela 4.1 contendo algumas das simulações efetuadas e os erros obtidos através da metodologia descrita na seção 4.3.

Potência	E_1	E_2
s=1	7.2524	3.6262
s=0.5	6.6215	3.3107
s=0.3	6.5645	3.2822
s=0.28	6.5589	3.2794

Tabela 4.1: Resultado das simulações para busca da potência s para o Modelo de Malthus Modificado.

Assim, concluímos que a melhor potência para o modelo é $s = 0.28$, ou seja, o modificador é do tipo expansivo. Elevando as funções de pertinência de entrada e saída a esta potência, obtemos as funções de pertinência modificadas das Figuras 4.8 e 4.9.

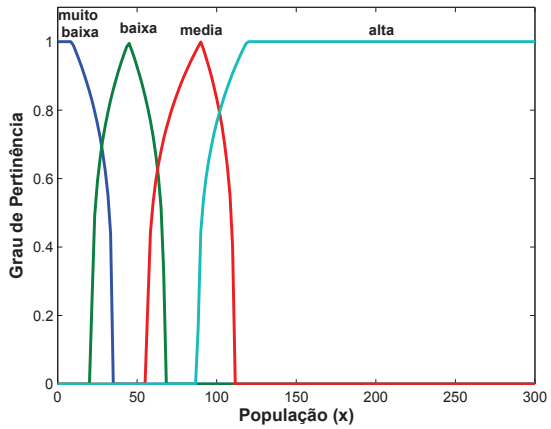

Figura 4.8: Funções de pert. de entrada modif. para o modelo de Malthus.

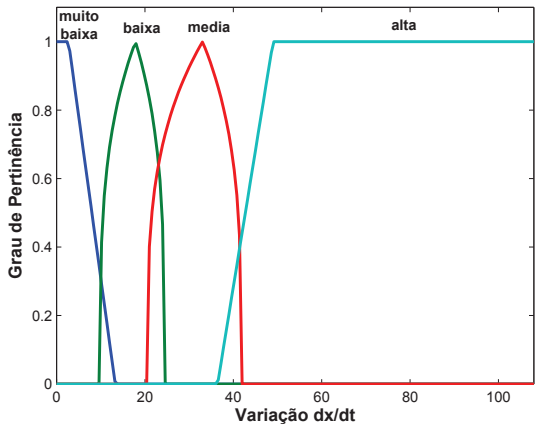

Figura 4.9: Funções de pert. de saída modif. para o modelo de Malthus.

Com as funções de pertinência de entrada e saída modificadas pela potência $s = 0.28$, temos que o sistema p-fuzzy modificado produz, em relação a solução determinística, erros dados por $E_1' = 6.5589$ e $E_2' = 3.2794$, ou seja, temos as vantagens $V_1' = 0.6935$ e $V_2' = 0.3468$ em relação ao sistema p-fuzzy sem modificações. Dessa forma, podemos afirmar que o modelo é melhor aproximado pelo sistema p-fuzzy modificado do que pelo sistema p-fuzzy. A Figura 4.10 mostra a solução do sistema p-fuzzy modificado em relação a solução determinística considerando a população inicial igual a 2.

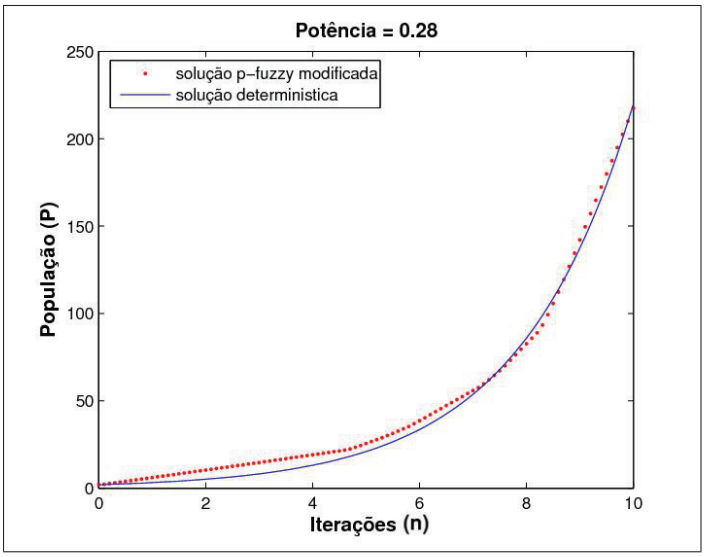

Figura 4.10: Solução do sistema p-fuzzy modificado e solução determinística com número de iterações $n = 100$.

A rotina computacional do sistema p-fuzzy modificado encontra-se no Apêndice B.1.

4.4.2 Modelo de Verhulst

O sistema p-fuzzy para o modelo de Verhulst produz, em relação ao modelo determinístico, os erros $E_1 = 8.7666$ e $E_1 = 0.4383$, cujas formas são descritas pelas expressões (4.4) e (4.5).

Na Tabela 4.2 são descritas alguns erros cometidos para algumas potências, aplicando a teoria dos modificadores linguísticos, e utilizando a metodologia descrita na seção 4.3.

Potência	E_1	E_2
s=1	6.5285	0.0256075
s=1.1	5.9891	0.023492
s=1.15	5.9493	0.0233356
s=1.13	5.8283	0.0228612

Tabela 4.2: Resultado das simulações para busca da potência s para o Modelo de Verhulst Modificado.

Assim, concluímos que a melhor potência para o modelo é $s = 1.13$, ou seja, temos modificadores do tipo restritivos. Elevando as funções de pertinência de entrada a esta potência, obtemos as funções de pertinência modificadas da Figura 4.11.

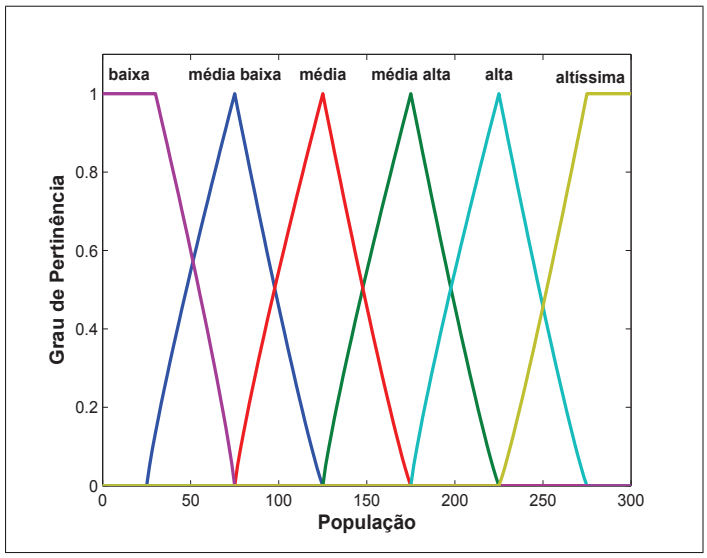

Figura 4.11: Funções de pertinência de entrada modificadas pela potência $s = 1.13$

Com as funções de pertinência de entrada modificadas pela potência $s = 1.13$, temos que o sistema p-fuzzy modificado produz, em relação a solução determinística, os erros de $E_1' = 5.8283$ e $E_1' = 5.8283$, ou seja, temos as vantagens de $V_1' = 0.7$ e $V_2' = 0.0027463$ em relação ao sistema p-fuzzy sem modificações.

Dessa forma, podemos afirmar que o modelo é melhor aproximado pelo sistema p-fuzzy modificado do que pelo sistema p-fuzzy. A Figura 4.12 mostra a solução do sistema p-fuzzy modificado em relação a solução determinística com população inicial igual a 20.

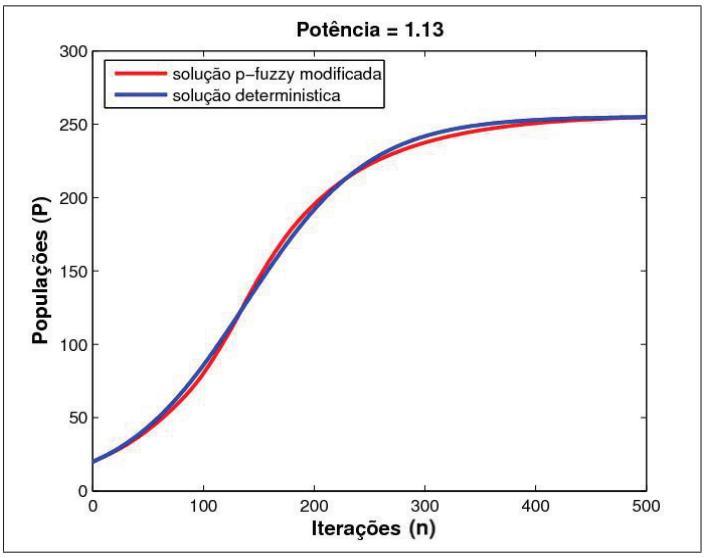

Figura 4.12: Solução do sistema p-fuzzy modificado e solução determinística com número de iterações $n = 500$.

Para este sistema p-fuzzy modificado, temos a rotina computacional descrita no Apêndice B.2.

4.4.3 Modelo de Montroll

O sistema p-fuzzy para o modelo de Montroll produz, em relação ao modelo determinístico, os erros $E_1 = 12.6393$ e $E_2 = 2.5278$, segundo os critérios das equações (3.10) e (3.11).

Apresentamos a Tabela 4.3 contendo alguns valores de s simulados e os erros E_1 e E_2, utilizando a metodologia descrita na seção 4.3.

Potência	E_1	E_2
s=1	12.6393	2.5278
s=0.9	11.3975	2.2795
s=0.8	11.3882	2.2777
s=0.83	10.9971	2.1994

Tabela 4.3: Resultado das simulações para busca da potência s para o Modelo de Montroll Modificado.

Assim, concluímos que a melhor potência para o modelo é $s = 0.83$ (modificador expansivo), e elevando as funções de pertinência de entrada e saída a esta potência, obtemos as funções de pertinência modificadas das Figuras 4.13 e 4.14.

Com as funções de pertinência de entrada e saída modificadas pela potência $s = 0.83$, temos que o sistema p-fuzzy modificado produz, em relação a solução determinística, os erros

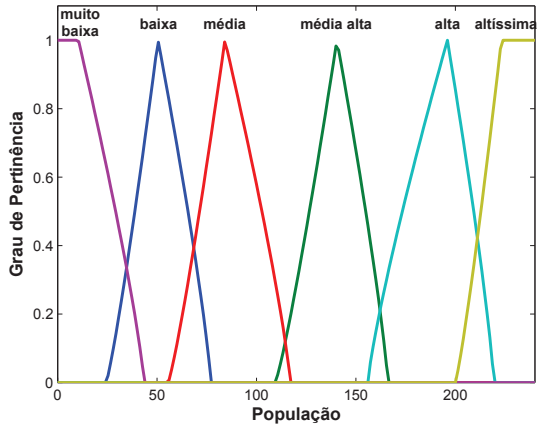

Figura 4.13: Funções de pertinência de entrada modificadas para o modelo de Montroll.

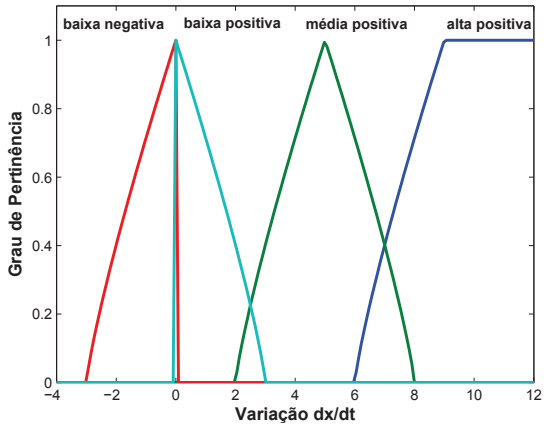

Figura 4.14: Funções de pertinência de saída modificadas para o modelo de Montroll.

$E_1' = 10.9971$ e $E_1' = 2.1994$, ou seja, temos as vantages $V_1' = 1.6421$ e $V_2' = 0.3284$ em relação ao sistema p-fuzzy sem modificações.

Portanto, temos que o modelo é melhor aproximado pelo sistema p-fuzzy modificado do que pelo sistema p-fuzzy. A Figura 4.15 mostra a solução do sistema p-fuzzy modificado em relação a solução determinística com população inicial igual a 5.

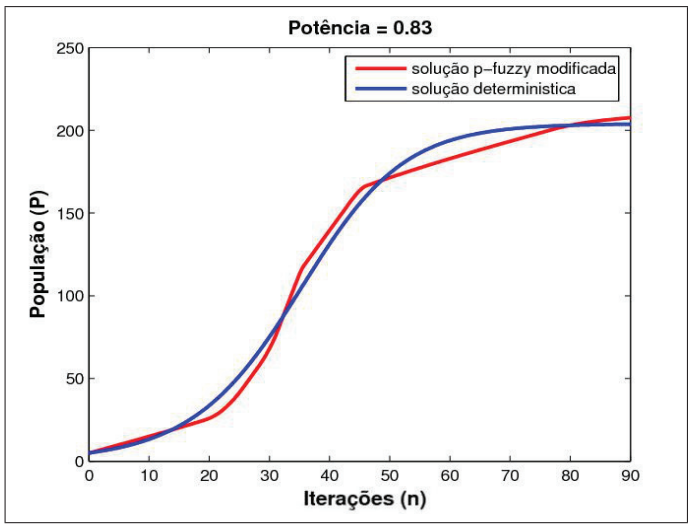

Figura 4.15: Solução do sistema p-fuzzy modificado e solução determinística com número de iterações $n = 450$.

No Apêndice B.3 podem ser vistas as rotinas computacionais para o sistema p-fuzzy modificado do modelo de Montroll.

4.4.4 Modelo de Transferência da População HIV Assintomática para Sintomática

O sistema p-fuzzy para o modelo da AIDS produz, em relação aos dados de Peterman (1985), os erros $E_1 = 0.07339$ e $E_2 = 0.9238$, segundo critério descrito nas expressões (4.4) e (4.5).

Apresentamos, a seguir, a Tabela 4.4 contendo algumas das simulações efetuadas e os respectivos erros para algumas potências, aplicando a teoria dos modificadores linguísticos, e utilizando a metodologia descrita na seção 4.3.

Potência	E_1	E_2
s=1	0.0733916	0.9238
s=1.2	0.07365	0.9387
s=1.09	0.07018	0.8886
s=1.08	0.06996	0.8852

Tabela 4.4: Resultado das simulações para busca da potência s para o Modelo da AIDS Modificado.

Dessa forma, temos que a melhor potência para o modelo da AIDS é $s = 1.08$ e elevando as funções de pertinência de entrada e saída a esta potência, obtemos as funções de pertinência modificadas das Figuras 4.16 e 4.17.

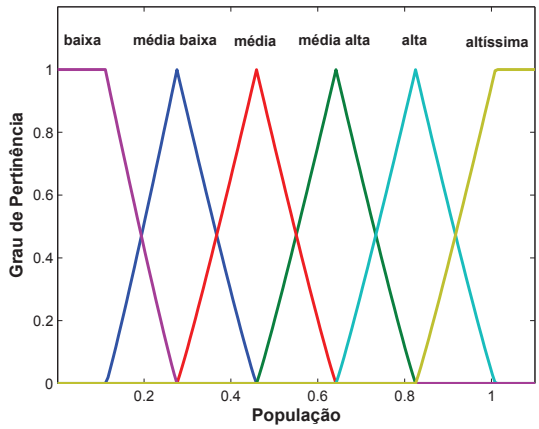

Figura 4.16: Funções de pertinência de entrada modificadas para o modelo da AIDS.

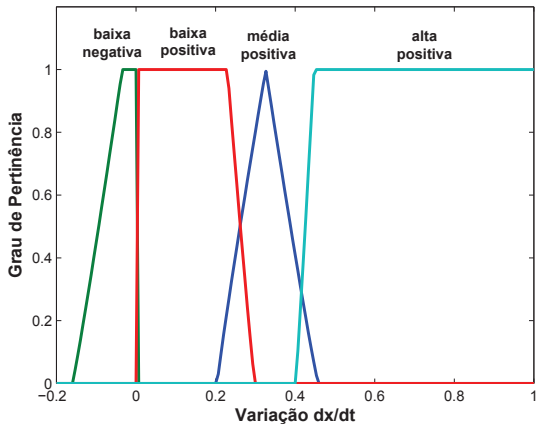

Figura 4.17: Funções de pertinência de saída modificadas para o modelo da AIDS.

Com as funções de pertinência de entrada e saída modificadas pela potência $s = 1.08$, temos que o sistema p-fuzzy modificado produz, em relação aos dados de Peterman (1985), os erros

$E_1' = 0.06996$ e $E_2' = 0.8852$, ou seja, temos as vantagens de $V_1' = 0.0034316$ e $V_2' = 0.0386$ em relação ao sistema p-fuzzy sem modificações. Para este modelo, observamos modificadores tanto expansivos como restritivos.

Assim, podemos afirmar que o modelo é melhor aproximado pelo sistema p-fuzzy modificado do que pelo sistema p-fuzzy. As Figuras 4.18 e 4.19 mostram a solução do sistema p-fuzzy modificado em relação aos dados de Peterman, respectivamente, para a população HIV assintomática e para a população HIV sintomática.

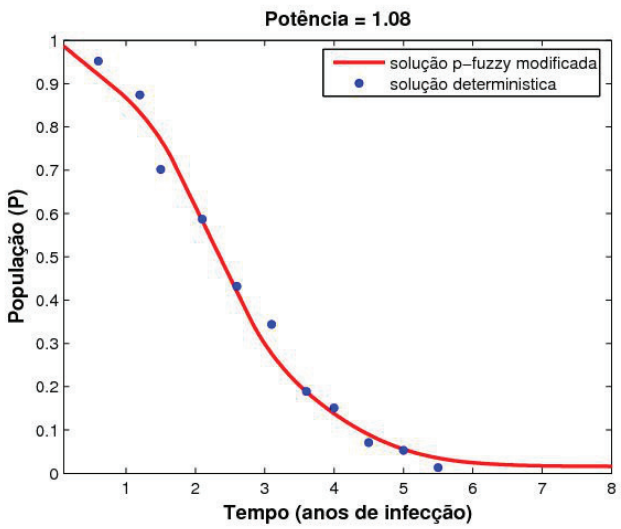

Figura 4.18: Evolução da população HIV Assintomática com sistema p-fuzzy modificado.

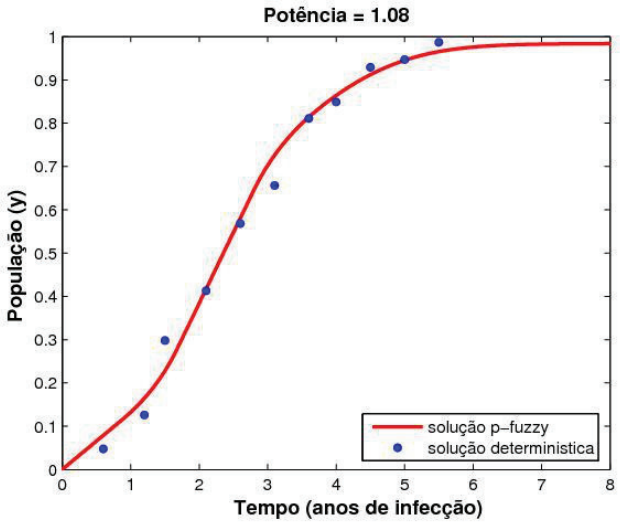

Figura 4.19: Evolução da população HIV Sintomática com sistema p-fuzzy modificado.

Para o sistema p-fuzzy modificado do modelo da AIDS, temos as rotinas computacionais localizadas no Apêndice B.4. A seguir será apresentado o sistema p-fuzzy modificado para o modelo presa-predador (veja a referência [8]).

4.4.5 Modelo Presa-Predador

O sistema p-fuzzy para o modelo presa-predador produz, em relação ao modelo clássico, os erros $E_1 = 38.1874$ e $E_2 = 5.1225$, segundo critérios apresentados nas expressões (4.4) e (4.5).

Na Tabela 4.5 são descritas algumas das simulações efetuadas, e os erros obtidos, utilizando a metodologia descrita na seção 4.3.

Potência	E_1	E_2
s=1	38.1874	5.1225
s=0.7	34.1832	4.2125
s=0.75	33.7511	4.0123
s=0.759	32.2027	3.8117

Tabela 4.5: Resultado das simulações para busca da potência s para o Modelo Presa-Predador Modificado.

Dessa forma, podemos dizer que a melhor potência encontrada foi $s = 0.759$ (modificador expansivo), e elevando as funções de pertinência de entrada a esta potência, obtemos as funções de pertinência modificadas das Figuras 4.20 e 4.21, respectivamente, para presas e predadores.

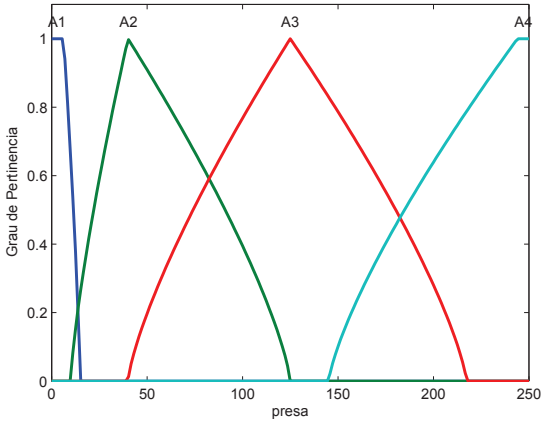

Figura 4.20: Funções de pertinência de entrada modificadas para as presas.

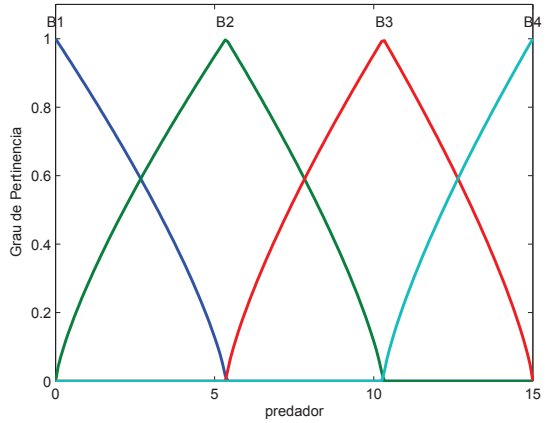

Figura 4.21: Funções de pertinência de entrada modificadas para os predadores.

Com as funções de pertinência de entrada modificadas pela potência $s = 0.759$, temos que o sistema p-fuzzy modificado produz, em relação ao modelo clássico, os erros $E_1' = 32.2027$ e $E_2' = 3,8117$, ou seja, temos as vantagens $V_1' = 5.9847$ e $V_2' = 1.3108$ em relação ao sistema p-fuzzy sem modificações [8].

Logo, o modelo é melhor aproximado pelo sistema p-fuzzy modificado do que pelo sistema p-fuzzy. A Figura 4.22 mostra a solução do sistema p-fuzzy modificado em relação ao modelo clássico.

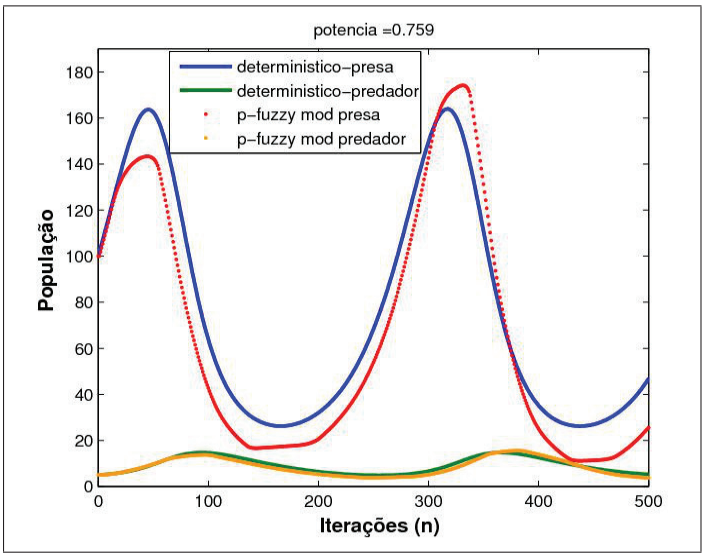

Figura 4.22: Solução do sistema p-fuzzy modificado e solução determinística com número de iterações $n = 460$.

Podem ser vistas no Apêndice B.5 a rotina computacional para o sistema p-fuzzy modificado do modelo presa-predador.

4.4.6 Modelo do Controle de Pragas

O sistema p-fuzzy para o modelo do controle de pragas mantém o nível de população das pragas abaixo de 40, devido ao controle.

A seguir, ilustraremos na Tabela 4.6 algumas das simulações efetuadas, utilizando a teoria dos modificadores fuzzy, e utilizando a metodologia descrita na seção 4.3.

Potência	E_1	E_2
s=1	0.00	0.00
s=0.96	0.005041	0.02616
s=0.92	0.008571	0.04447
s=0.9	0.009850	0.05111

Tabela 4.6: Resultado das simulações para busca da potência s para o Modelo Presa-Predador Modificado.

Podemos concluir que a melhor potência encontrada é $s = 0.9$, ou seja, o modificador será expansivo, e elevando as funções de pertinência de saída a esta potência, obtemos as funções de pertinência modificadas da Figura 4.23.

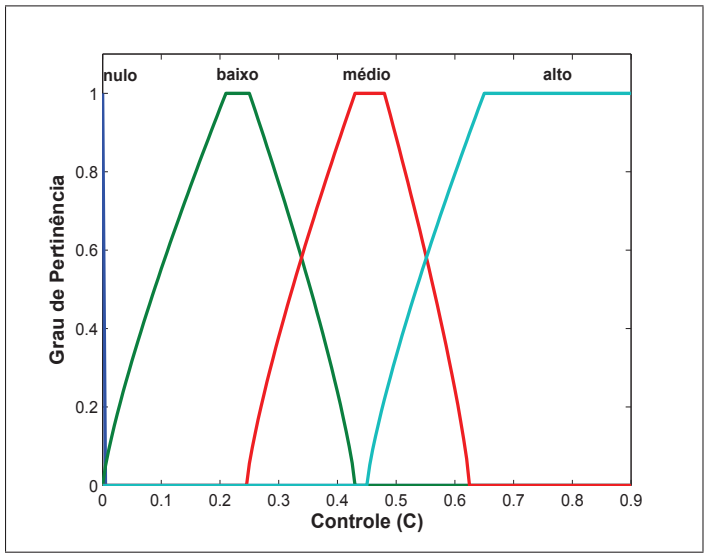

Figura 4.23: Funções de pertinência de entrada modificadas pela potência $s = 0.9$.

Com as funções de pertinência modificadas pela potência $s = 0.9$, temos que o sistema p-fuzzy modificado produz, em relação ao sistema p-fuzzy, os erros $E_1 = 0.009850$ e $E_2 = 0.05111$, ou seja, temos um controle maior da praga com a utilização do sistema modificado em relação ao sistema sem modificações.

Dessa forma, podemos afirmar que o modelo é melhor aproximado pelo sistema modificado do que pelo sistema sem modificações. A Figura 4.24 mostra o controle do sistema modificado em relação ao sistema sem controle.

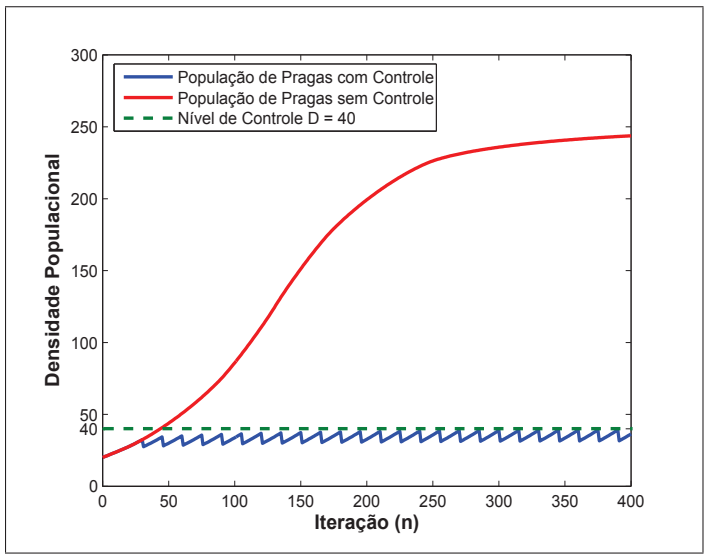

Figura 4.24: Solução do sistema p-fuzzy modificado e solução determinística com número de iterações $n = 400$.

A descrição da rotina computacional utilizada neste modelo pode ser vista no Apêndice B.6.

4.5 Sistemas p-Fuzzy Modificados no Tempo

O desafio deste trabalho é aplicar a teoria dos modificadores fuzzy no tempo, isto é, as funções de pertinência devem ser modificadas por uma potência diferente a cada iteração.

De forma geral, é necessário encontrar uma função para a alteração das potências que modificam as funções de pertinência. Se a dificuldade dos modelos modificados é a busca da potência adequada, para os modelos modificados no tempo, a grande dificuldade é encontrar essa função responsável por alterar as potências das funções de pertinência. A metodologia utilizada para encontrar tal função foi descrita anteriormente, na seção 4.3.

A função procurada depende da variável i, onde $i = 1, \ldots, n$, sendo n o número de iterações do sistema. Aplicamos a teoria dos modificadores fuzzy a cada iteração para dois dos modelos citados acima: Malthus e Controle de Pragas. Descreveremos, a seguir, qual foi o comportamento de cada modelo, com soluções p-fuzzy modificadas no tempo.

Vale ressaltar que a aplicação da teoria dos modificadores fuzzy no tempo não é apresentada nas referências citadas, assim, este trabalho apresenta uma pequena contribuição para os sistemas p-fuzzy. Os programas e rotinas computacionais que foram utilizados para obter os resultados das simulações nesta seção encontram-se no Apêndice B.

4.5.1 Sistema p-Fuzzy Modificado no Tempo para o Modelo de Malthus

O sistema p-fuzzy para o modelo de Malthus produz, em relação ao modelo determinístico, os erros $E_1 = 7.2524$ e $E_2 = 3.6262$ e quando aplicando a teoria dos modificadores fuzzy, foi construído um sistema p-fuzzy modificado pela potência $s = 0.28$.

Com funções de pertinência das variáveis de entrada e saída elevadas a potência $s = 0.28$ temos que o sistema p-fuzzy modificado produz, em relação a solução determinística, os erros $E_1' = 6.5589$ e $E_2' = 3.2794$ e, portanto, temos as vantagens $V_1' = 0.6935$ e $V_2' = 0.3468$ em relação ao sistema p-fuzzy sem modificações.

A Tabela 4.7 apresenta algumas das simulações e os erros obtidos utilizando as potências para as funções de pertinência de entrada e saída, segundo os critérios descritos nas expressões (4.6) e (4.7), e a metodologia apresentada na seção 4.3.

Função	E_1	E_2
$f(i) = 1/i$	8.2922	4.1461
$f(i) = 0.5 + 1/i$	6.6215	3.6563
$f(i) = 0.28 + 0.5/i$	6.5645	3.2819
$f(i) = 0.2 + 0.5/i$	6.5589	3.2709
$f(i) = 0.2 + 0.2/i$	6.5389	3.2694

Tabela 4.7: Resultado das simulações para busca da função $f(i)$ para o Modelo de Malthus Modificado no Tempo.

Foram realizados vários testes na busca pela função de alteração da potência em cada iteração e a melhor função encontrada foi $f(i) = 0.2 + \dfrac{0.2}{i}$, que determina a sequência de potências que alteram as funções de pertinência com $i = 1, \ldots, n$, sendo $n = 100$ o número de iterações. Em todas as iterações, temos modificadores do tipo expansivos.

A vantagem do sistema modificado no tempo em relação ao sistema sem modificações e em relação ao sistema modificado são, respectivamente, $V_1 = 0.02$ e $V_2 = 0.01$. Como foi encontrada a função de alteração de potências adequada, o sistema p-fuzzy modificado no tempo é aquele que melhor se aproxima do modelo clássico, em relação ao sistema p-fuzzy e sistema p-fuzzy modificado.

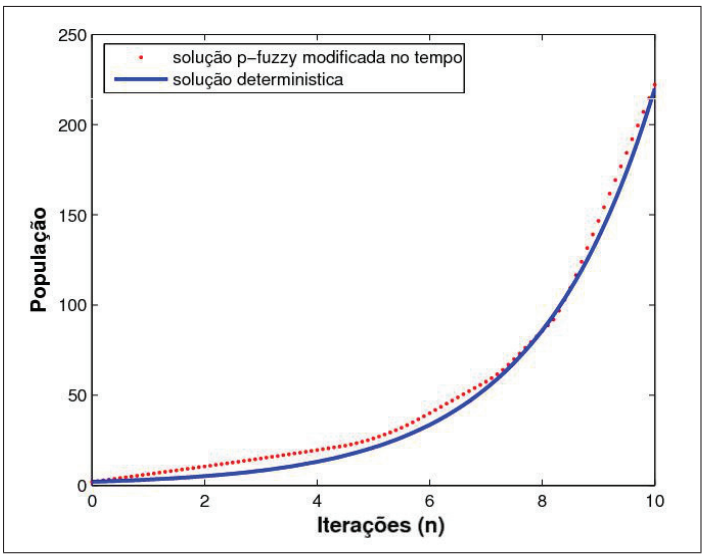

Figura 4.25: Solução do sistema p-fuzzy modificado no tempo e solução determinística.

A Figura 4.25 mostra a solução do sistema p-fuzzy modificado no tempo em relação a solução determinística com população inicial igual a 2.

A rotina computacional para o sistema p-fuzzy modificado no tempo para o modelo de Malthus pode ser encontrada no Apêndice B.1.

4.5.2 Sistema p-Fuzzy Modificado no Tempo para o Modelo do Controle de Pragas

O sistema p-fuzzy para o modelo do Controle de Pragas mantém o nível de população das pragas abaixo de 40, devido ao controle.

Aplicando a teoria dos modificadores fuzzy encontramos a potência $s = 0.9$, e elevando as funções de pertinência de saída a esta potência, obtemos as funções de pertinência modificadas da Figura 4.23.

Na Tabela 4.8 estão descritos os erros, segundo os critérios apresentados nas expressões (4.6) e (4.7), para as simulações que foram feitas, aplicando a teoria dos modificadores fuzzy juntamente com a metodologia descrita na seção 4.3.

Função	E_1	E_2
$f(i) = 1/i$	0.001318	0.3445
$f(i) = 0.5 + 1/i$	0.01693	0.7898
$f(i) = 0.9 + 1/i$	0.02108	0.9963

Tabela 4.8: Resultado das simulações para busca da função $f(i)$ para o Modelo do Controle de Pragas Modificado no Tempo.

Portanto, a melhor função encontrada para este modelo é $f(i) = 0.9 + \frac{1}{i}$, que determina a sequência de potências que alteram as funções de pertinência, de entrada e saída, com $i = 1, \ldots, n$, sendo $n = 400$ o número de iterações. Até a iteração número 3, os modificadores utilizados são restritivos, ou seja, maiores do que um. A partir da iteração número 4, temos que os modificadores são expansivos. A Figura 4.26 representa a evolução da praga com e sem controle, em que pode-se verificar que aplicando o biocida, a população de pragas não ultrapassa o valor de 40, considerando a população no intervalo $[0, 300]$. As vantagens do sistema modificado no tempo em relação ao sistema sem modificações e em relação ao sistema modificado em relação ao erro E_1 são, respectivamente, $V_1 = 0.215841$ e $V_{1'} = 0.02108$, e em relação ao erro E_2 são, respectivamente, $V_2 = 1.02246$ e $V_{2'} = 0.9963$ (Figura 4.27).

Observamos que para os dois critérios utilizados, a função de alteração de potências $f(i)$ que resulta em um controle maior da praga é a mesma.

Segue no Apêndice B.6 a rotina computacional para o sistema p-fuzzy modificado no tempo para o modelo do controle de pragas.

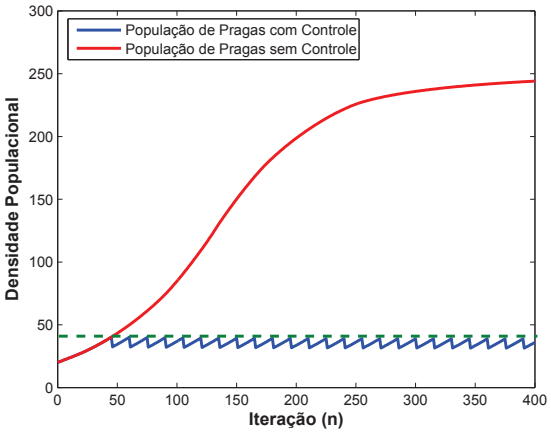

Figura 4.26: Dens. populacional do sistema p-fuzzy modificado.

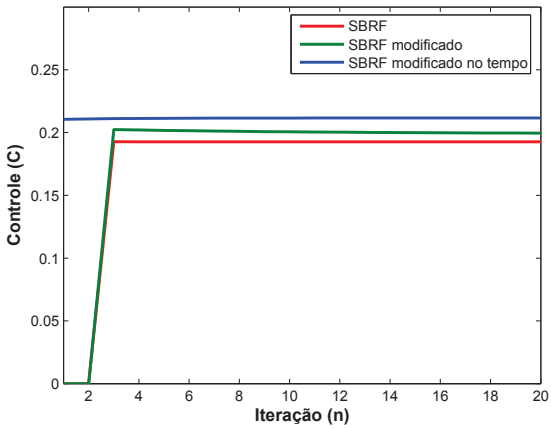

Figura 4.27: Comparação de eficiência entre os sistemas.

Concluímos que, se encontrada a função de alteração de potências adequada, o sistema p-fuzzy com controle modificado no tempo tem maior eficácia para o modelo do controle de pragas (veja [9]). Para o intervalo de aplicação de biocida de 15 iterações (que pode representar o intervalo da aplicação em dias), a quantidade de biocida de cada aplicação pode ser alterada, podendo obter um controle maior da praga e proporcionar melhor qualidade e maior produção da lavoura.

No próximo capítulo apresentamos as conclusões e considerações finais sobre o trabalho, haja vista as conclusões pontuais já terem sido feitas no final de cada capítulo.

Capítulo 5

Considerações Finais

O Capítulo 1 teve como objetivo dar uma visão da introdução da teoria dos conjuntos fuzzy bem como apresentar os Sistemas Baseados em Regras Fuzzy que foram utilizados com ampla extensão neste trabalho. No Capítulo 2, estudou-se a solução de Métodos Numéricos para equações diferenciais ordinárias e tópicos de integração numérica, muito utilizados neste trabalho.

Nos Capítulos 3 e 4, abordamos o principal objeto de estudo do nosso trabalho, que foram os sistemas dinâmicos fuzzy aplicados às dinâmicas populacionais. Concluímos que, a utilização da teoria dos modificadores linguísticos aplicadas em sistemas p-fuzzy proporciona uma vantagem em relação a sistemas p-fuzzy sem modificações, considerando os critérios adotados. Quando elevamos as funções de pertinência a potências diferentes de 1, obtemos vantagens calculadas numericamente. Depois, utilizando as modificações aplicadas a cada iteração, podemos perceber que sistemas modificados no tempo aproximam-se melhor aos modelos determinísticos que os sistemas modificados, deixando claro que modificar no tempo é mais vantajoso para os casos estudados. Dessa forma, vale ressaltar alguns pontos que seguem.

Para os sistemas modificados, a dificuldade é encontrar a potência adequada, que faz com que a aproximação aos modelos clássicos seja melhor. Para o sistema modificado no tempo, o desafio é encontrar a função de modificação de potência que proporciona a maior vantagem.

É importante ressaltar que para os dois critérios utilizados na escolha da melhor potência, as potências adequadas são as mesmas em todos os modelos estudados.

As modificações que trazem vantagens são, em alguns modelos do tipo restritivas, em outros modelos, do tipo expansiva e no caso do modelo do HIV são tanto restritivas como expansivas.

Dessa forma, não podemos generalizar a princípio qual o tipo de modificador é adequado para cada modelo.

Enfim, este trabalho é um passo inicial para estudos sobre modificadores linguísticos aplicados em sistemas p-fuzzy, visto que o significado teórico das vantagens de cada modificação fica em aberto para pesquisas posteriores. Como contribuição deste trabalho, anexamos as rotinas computacionais utilizadas para obter as modificações e as modificações no tempo para os sistemas p-fuzzy (Apêndices A e B).

Outros trabalhos poderão ser desenvolvidos, como utilizar SBRF utilizando outros métodos de defuzzificação como *máximo, média dos máximos, altura ou altura modificada*, para verificar se obtemos resultados semelhantes, fazer modificações no tempo para outros modelos de Controle de Pragas apresentados em [21] e demonstrar teoricamente as vantagens do modelo p-fuzzy modificado e modificado no tempo em relação ao modelo p-fuzzy sem modificações.

Referências Bibliográficas

[1] Anderson, R.M., Medley, G.F., May, R.M. e Johnson, A.M., *A Preliminary Study of the Transmission Dynamics of the Human Immunodeficiency Virus (HIV), the Causative Agent of AIDS*, Journal of Mathematics Applied in Medicine and Biology, 3 (1986), 229-263.

[2] Barros, L. C. e Bassanezi, R. C., *Tópicos de Lógica Fuzzy e Biomatemática*, 2° edição, IMECC, Universidade Estadual de Campinas, São Paulo, 2010.

[3] Bassanezi, R. C. e Ferreira Jr, W. C., *Equações diferenciais com aplicações*, Edit. HARBRA, São Paulo, 1998.

[4] Braga, A e Sousa-Silva, C. R., *Afídeos de citros (Citrus sinensis) e seus predadores na região de São Carlos-SP*, Monografia, Departamento de Ecologia e Biologia Evolutiva da Universidade Federal de São Carlos, 1999.

[5] Cecconelo, M. S., *Modelagem Alternativa para Dinâmica Populacional: Sistemas Dinâmicos Fuzzy*, Dissertação de Mestrado, IMECC, Universidade Estadual de Campinas, São Paulo, 2006.

[6] Cock, M. e Kerre, E. E.,*Fuzzy modifiers based on fuzzy relations*, Information Sciences, 160 (2004), 173-199.

[7] Ferreira, D. P. L., *Sistema p-Fuzzy Aplicado às Equações Diferenciais Parciais*, Dissertação de Mestrado, Famat-UFU, Uberlândia-MG, 2011.

[8] Ferreira, T. F. e Jafelice, R. S. M., *Sistema p-Fuzzy Modificado para o Modelo Presa-Predador*, em Anais da XI Semana da Matemática - SEMAT, Universidade Federal de Uberlândia - UFU, Uberlândia-MG, 2011, [35] 141-144.

[9] Ferreira, T. F. e Jafelice, R. S. M., *Modelo do Controle de Pragas: Sistema p-Fuzzy Modificado e Modificado no Tempo*, Congresso Nacional de Matemática Aplicada e Computacional - CNMAC 2012, Águas de Lindóia-SP, 2012, (aceito).

[10] Jafelice, R. S. M., *Modelagem Fuzzy para Dinâmica de Transferência de Soropositivos para HIV em Doença Plenamente Manifesta*, Tese de Doutorado, FEEC, Universidade Estadual de Campinas, Campinas, 2003.

[11] Jafelice, R.M., Barros, L.C. e Bassanezi, R.C., *Teoria dos Conjuntos Fuzzy com Aplicações*, Uma Publicação da SBMAC - Editora Plêiade,17, 2005.

[12] Jafelice, R. S. M., Barros, L. C. e Bassanezi, R. C., *Usando a Teoria Fuzzy na Modelagem de Fenômenos Biológicos*, Simpósio de Aplicações em Lógica Fuzzy, Sorocaba-SP, 2008.

[13] Klir, G. J. e Yuan, B., *Fuzzy Sets and Fuzzy Logic Theory and applications*, Prentice-Hall PTR, New Jersey, EUA,1995.

[14] Leite, J. C. S., *Sistemas dinâmicos fuzzy aplicados a processos difusivos*, Tese de Doutorado, IMECC-Unicamp, Campinas-SP, 2011.

[15] Lotka, A. J., *Elements of Physical Biology*, Williams and Wilkins, Baltimore, 1925.

[16] Mamdani, E. H. e Assilian, S., *An experiment in linguistic synthesis with a fuzzy logic controller*, Int.J.Man-Machine Studies 7 (1975), 1-13.

[17] Massad, E., Menezes, R. X. de, Silveira, P. S. P. e Ortega, N. R. S., *Métodos Quantitativos em Medicina*, Barueri-SP : Manole, 2004.

[18] Peixoto, M. S., *Sistemas Dinâmicos e Controladores Fuzzy: um estudo da dispersão da morte súbita dos citros em São Paulo*, Tese de Doutorado, IMECC, Universidade Estadual de Campinas, São Paulo, 2005.

[19] Peterman, T., Drotman, D. e Curran, J., *Epidemiology of the acquired immunodeficiency syndrome AIDS*, Epidemiology Reviews, 7 (1985) 7-21.

[20] Ruggiero, M. A. G., Lopes, V. R., *Cálculo Numérico: Aspectos Teóricos e Computacionais*, São Paulo: Makron Books, 1996.

[21] Santos, L.R., *Estratégias para o Controle de Pragas: Sistemas p-Fuzzy com Controle Híbrido*, Dissertação de Mestrado, IMECC, Universidade Estadual de Campinas, São Paulo, 2008.

[22] Silva, J. S., *Análise de Estabilidade de Sistemas Dinâmicos p-Fuzzy com aplicações em Biomatemática*, Tese de Doutorado, IMECC, Universidade Estadual de Campinas, São Paulo, 2005.

[23] Thomas Robert Malthus, Disponível em http://tinyurl.com/7g7jt5p. Acessado em 31 de janeiro de 2012.

[24] Verhulst, P. F., Recherches Mathématiques sur La Loi D'Accroissement de la Population, Nouveaux Mémoires de l'Académie Royale des Sciences et Belles-Lettres de Bruxelles, 18, Art. 1, 1-45, 1845 (Investigações Matemáticas sobre a Lei de Crescimento da População).

[25] Villela, M.F.S., Santos, P.B. e Jafelice, R.S.M., *Diagnóstico Médico Fuzzy de Doenças Infantis*, em Anais da VII Semana da Matemática da Universidade Federal de Uberlândia, Uberlândia-MG, 2007, 94-95.

[26] Volterra, V., *Variazionie fluttuazioni del numero d'individui in specie animali conviventi*, Mem. Acad. Licei. 2, 31-113, 1926. (Variations and fluctuations of a number of individuals in animal species living together. Translation In: R. N. Chapman: Animal Ecology. New York: McGraw Hill, 409-448, 1931.)

[27] Zadeh, L. A., *Fuzzy Sets*, Information and Control 8 (1965), 338-353.

Apêndice A

A.1 Programa para a Função de Pertinência Triangular Modificada – "trimfquadrado"

```
function y = trimfquadrado(x, params)
%TRIMF Triangular membership function.
%   TRIMF(X, PARAMS) returns a matrix which is the triangular
%   membership function evaluated at X. PARAMS = [A B C D] is a 4-element
%   vector that determines the break points of this membership function and potency.
%   Usually we require A <= B <= C.
%
%   Note that this MF always has a height of unity. To have a triangular
%   MF with a height less than unity, use TRAPMF instead.
%
%   For example:
%
%       x = (0:0.2:10)';
%       y1 = trimf(x, [3 4 5 1]);
%       y2 = trimf(x, [2 4 7 1]);
%       y3 = trimf(x, [1 4 9 1]);
%       subplot(211), plot(x, [y1 y2 y3 y4]);
%       y1 = trimf(x, [2 3 5 1]);
%       y2 = trimf(x, [3 4 7 1]);
%       y3 = trimf(x, [4 5 9 1]);
%       subplot(212), plot(x, [y1 y2 y3 y4]);
%       set(gcf, 'name', 'trimfquadrado', 'numbertitle', 'off');
%
%   See also DSIGMF, EVALMF, GAUSS2MF, GAUSSMF, GBELLMF, MF2MF, PIMF, PSIGMF,
%   SIGMF, SMF, TRAPMF, ZMF.

%   Roger Jang, 6-29-93, 10-5-93, 4-14-94.
%   Copyright 1994-2002 The MathWorks, Inc.
%   $Revision: 1.18 $  $Date: 2002/04/14 22:21:19 $


n=params(length(params));
params=params(1:length(params)-1);
```

```
if nargin ~= 2
   error('Two arguments are required by the triangular MF.');
elseif length(params) < 3
   error('The triangular MF needs at least three parameters.');
end

a = params(1); b = params(2); c = params(3);

if a > b,
   error('Illegal parameter condition: a > b');
elseif b > c,
   error('Illegal parameter condition: b > c');
elseif a > c,
   error('Illegal parameter condition: a > c');
end

%*********************************************************************
%  Momento do Programa que as funções são elevadas a potência 'n'
%
%*********************************************************************

y = zeros(size(x));

% Left and right shoulders (y = 0)
index = find(x <= a | c <= x);
y(index) = zeros(size(index));

% Left slope
if (a ~= b)
   index = find(a < x & x < b);
   y(index) = ((x(index)-a)/(b-a)).^n;
end

% right slope
if (b ~= c)
   index = find(b < x & x < c);
   y(index) = ((c-x(index))/(c-b)).^n;
end

% Center (y = 1)
index = find(x == b);
y(index) = ones(size(index));
```

A.2 Programa para aFunção de Pertinência Trapezoidal Modificada – “trapmfquad”

```
function y = trapmfquad(x, params)
%TRAPMF Trapezoidal membership function.
%   TRAPMF(X, PARAMS) returns a matrix which is the trapezoidal
%   membership function evaluated at X. PARAMS = [A B C D E] is a 5-element
%   vector that determines the break points of this membership function and potency.
%   We require that A <= B and C <= D. If B >= C, this membership
%   function becomes a triangular membership function that could have
%   a height less than unity. (See the example below.)
%
%   For example:
%
%       x = (0:0.1:10)';
%       y1 = trapmf(x, [2 3 7 9 1]);
%       y2 = trapmf(x, [3 4 6 8 1]);
%       y3 = trapmf(x, [4 5 5 7 1]);
%       y4 = trapmf(x, [5 6 4 6 1]);
%       plot(x, [y1 y2 y3 y4 y5]);
%       set(gcf, 'name', 'trapmfquad', 'numbertitle', 'off');
%
%   See also DSIGMF, EVALMF, GAUSS2MF, GAUSSMF, GBELLMF, MF2MF, PIMF, PSIGMF,
%   SIGMF, SMF, TRIMF, TRIMFQUADRADO, ZMF.

%   Roger Jang, 6-28-93, 10-5-93, 4-14-94.
%   Copyright 1994-2002 The MathWorks, Inc.
%   $Revision: 1.22 $  $Date: 2002/04/14 22:21:13 $



n=params(length(params));
params=params(1:length(params)-1);

if nargin ~= 2
   error('Two arguments are required by the trapezoidal MF.');
elseif length(params) < 4
   error('The trapezoidal MF needs at least four parameters.');
end

a = params(1); b = params(2); c = params(3); d = params(4);

if a > b,
   error('Illegal parameter condition: a > b');
elseif c > d,
   error('Illegal parameter condition: c > d');
end

y1 = zeros(size(x));
y2 = zeros(size(x));

% Compute y1
```

```
%*********************************************************************
%  Momento do Programa que as funções são elevadas a potência 'n'
%
%*********************************************************************

index = find(x >= b);
if ~isempty(index),
   y1(index) = ones(size(index));
end
index = find(x < a);
if ~isempty(index),
   y1(index) = zeros(size(index));
end
index = find(a <= x & x < b);
if ~isempty(index) & a ~= b,
   y1(index) = ((x(index)-a)/(b-a)).^n;
end

% Compute y2
index = find(x <= c);
if ~isempty(index),
   y2(index) = ones(size(index));
end
index = find(x > d);
if ~isempty(index),
   y2(index) = zeros(size(index));
end
index = find(c < x & x <= d);
if ~isempty(index) & c ~= d,
   y2(index) = ((d-x(index))/(d-c)).^n;
end

% Compute y
y = min(y1, y2);
```

Apêndice B

B.1 Modelo de Malthus

Rotina Computacional para o Sistema p-Fuzzy

```
clear all
close all

%*****************************************
% População Inicial - x
%*****************************************
x=2;

%*****************************************
%Iterações – n
%*****************************************
n=100;

t=0;
out=0;
out1=zeros(n+1,1);
vetorx=zeros(n+1,1);
x1=0;
out1(1)=x1;
vetorx(1)=x;
z=1;
h=0.1;
for i=1:n

%*********************************************************************
% SBRF – malthus1 – ao final do programa
%*********************************************************************

  fismat=readfis('malthus1');
   out=evalfis([x],fismat);
   z=z+1;
   out1(z)=out;

   if (vetorx(z-1)==0)
      x=0;
   else
      x= x+ ((out1(z-1)+out1(z))*h)/2;
```

```
    end
       if (x<=0)
       x=0;
       vetorx(z)=0;
    end
      vetorx(z)=x;
   end
t=0:h:(n*h);
figure;
plot(t,vetorx,'.r')

%************************************************************
% SOLUÇÃO DETERMINISTICA DO MODELO DE MALTHUS
%************************************************************
h1=0:0.1:10;
g=2*exp(0.47*h1);
hold on
plot(h1,g,'b')

%************************************************************
% SBRF malthus1
%************************************************************

% Name='malthus1'
% Type='mamdani'
% Version=2.0
% NumInputs=1
% NumOutputs=1
% NumRules=4
% AndMethod='min'
% OrMethod='max'
% ImpMethod='min'
% AggMethod='max'
% DefuzzMethod='centroid'
%
% [Input1]
% Name='populacao'
% Range=[0 250]
% NumMFs=4
% MF1='media':'trimf',[56.5 90.1 110.780423280423]
% MF2='muitobaixa':'trapmf',[-230 -30.3 8.92857142857144 34.7]
% MF3='alta':'trapmf',[88.2936507936508 119 275 475]
% MF4='baixa':'trimf',[21.494708994709 44.6 67.1]
%
% [Output1]
% Name='dx/dt'
% Range=[0 108]
% NumMFs=4
% MF1='media':'trimf',[20.52 33.05 41.85]
% MF2='muitobaixa':'trapmf',[-100.4 -13.93 2.711 13.28]
% MF3='alta':'trapmf',[36.45 49 120.5 177.8]
% MF4='baixa':'trimf',[9.857 17.86 24.43]
%
% [Rules]
% 2, 2 (1) : 1
% 1, 1 (1) : 1
% 3, 3 (1) : 1
% 4, 4 (1) : 1
```

Rotina Computacional para o Sistema p-Fuzzy Modificado

```
clear all
close all

%*****************************************************
% Potência
%*****************************************************
%m=0.1:0.1:1.9;
m=0.25:0.01:0.35;


for j=1:length(m)


%*****************************************************
% População Inicial - x
%*****************************************************
x=2;

%*****************************************************
% Iterações - n
%*****************************************************
n=100;

t=0;
out=0;
out1=zeros(n+1,1);
vetorx=zeros(n+1,1);
x1=0;
out1(1)=x1;
vetorx(1)=x;
z=1;
h=0.1;

   for i=1:n


%*****************************************************
% Sistema Baseado em Regras
*****************************************************

a(j)=newfis('malthusmodificado');
 getfis(a(j))

Name='malthusmodificado'
Type='mamdani'
Version=2.0
NumInputs=1
NumOutputs=1
NumRules=4
AndMethod='min'
OrMethod='max'
```

```
ImpMethod='min'
AggMethod='max'
DefuzzMethod='centroid'


%****************************************************************
% Funções de Pertinência de Entrada
%****************************************************************
a(j)=addvar(a(j),'input','populacao',[0 300]);
  a(j)=addmf(a(j),'input',1,'muitobaixa','trapmfquad',[-230 -30.3 8.929 34.7 m(j)]);
  a(j)=addmf(a(j),'input',1,'baixa','trimfquadrado',[21.49 44.6 67.1 m(j)]);
  a(j)=addmf(a(j),'input',1,'media','trimfquadrado',[56.5 90.1 110.8 m(j)]);
  a(j)=addmf(a(j),'input',1,'alta','trapmfquad',[88.29 119 375 475 m(j)]);
%figure;
%plotmf(a(j),'input',1)

Name='populacao'
Range=[0 250];
NumMFs=4;
MF1='muitobaixa':'trapmfquad',[-230 -30.3 8.929 34.7 m];
MF2='baixa':'trimfquadrado',[21.49 44.6 67.1 m];
MF3='media':'trimfquadrado',[56.5 90.1 110.8 m];
MF4='alta':'trapmfquad',[88.29 119 375 475 m];

%****************************************************************
% Funções de Pertinência de Saída
%****************************************************************
a(j)=addvar(a(j),'output','dx/dt',[0 108]);
  a(j)=addmf(a(j),'output',1,'muitobaixa','trapmf',[-100.4 -13.93 2.711 13.28 m(j)]);
  a(j)=addmf(a(j),'output',1,'baixa','trimfquadrado',[9.857 17.86 24.43 m(j)]);
  a(j)=addmf(a(j),'output',1,'media','trimfquadrado',[20.52 33.05 41.85 m(j)]);
  a(j)=addmf(a(j),'output',1,'alta','trapmf',[36.45 49 120.5 177.8 m(j)]);
%figure;
%plotmf(a(j),'output',1);

Name='dx/dt'
Range=[0 108];
NumMFs=4;
MF1='muitobaixa':'trapmf',[-100.4 -13.93 2.711 13.28 m];
MF2='baixa':'trimfquadrado',[9.857 17.86 24.43 m];
MF3='media':'trimfquadrado',[20.52 33.05 41.85 m];
MF4='alta':'trapmf',[36.45 49 120.5 177.8 m];

%****************************************************************
%Regras
*****************************************************************
rulelist=[
1 1 1  1
2 2 1  1
3 3 1  1
4 4 1  1];

a(j)=addrule(a(j),rulelist);

showrule(a(j));

out=evalfis([x],a(j));
```

```
   z=z+1;
   out1(z)=out;

   if (vetorx(z-1)==0)
      x=0;
   else
      x= x+ ((out1(z-1)+out1(z))*h)/2;
   end
      if (x<=0)
      x=0;
      vetorx(z)=0;
   end
     vetorx(z)=x;
   end

%****************************************************************
% Solução Determinística
%****************************************************************
t=0:h:(n*h);
h1=0:0.1:10;
g=2*exp(0.47*h1);
figure;
plot(t,vetorx,'.r',h1,g,'b')
legend('solução p-fuzzy','solução deterministica')
title(strcat('potencia = ', num2str(m(j))));
figure;
plotmf(a(j),'input',1)
figure;
plotmf(a(j),'output',1)

%****************************************************************
% Critérios de Escolha do Erro
%****************************************************************
erro=max(abs(vetorx-g'))
erro1=(max(abs(vetorx-g')))./min(abs(vetorx))

v(j)=erro;
v1(j)=erro1;

end

disp('erro')
v

disp('erro1')
v1

k=min(v)
k1=min(v1)

kk=find(v==k);
kk1=find(v1==k1);

m(kk)
m(kk1)
```

Rotina Computacional para o Sistema p-Fuzzy Modificado no Tempo

```
clear all
close all

%*********************************************************
% População Inicial – x
%*********************************************************
x=2;

%*********************************************************
% Iterações – n
%*********************************************************
n=100;

t=0;
out=0;
out1=zeros(n+1,1);
vetorx=zeros(n+1,1);
x1=0;
out1(1)=x1;
vetorx(1)=x;
z=1;
h=0.1;

   for i=1:n

%m(i) = 1/i;
%m(i) = 0.5 + 1/i;
%m(i) = 0.2 + 0.5/i;
%m(i) = 0.28 + 0.5/i;
m(i) = 0.2 + 0.2/i;



%*********************************************************
% SBRF
%*********************************************************

a(i)=newfis('malthusmodificadonotempo');
getfis(a(i))

Name='malthusmodificadonotempo'
Type='mamdani'
Version=2.0
NumInputs=1
NumOutputs=1
NumRules=4
AndMethod='min'
OrMethod='max'
ImpMethod='min'
AggMethod='max'
DefuzzMethod='centroid'
```

```
%***************************************************
% Funções de Pertinência de Entrada
%***************************************************

a(i)=addvar(a(i),'input','populacao',[0 300]);
  a(i)=addmf(a(i),'input',1,'muitobaixa','trapmfquad',[-230 -30.3 8.929 34.7 m(i)]);
  a(i)=addmf(a(i),'input',1,'baixa','trimfquadrado',[21.49 44.6 65 m(i)]);
  a(i)=addmf(a(i),'input',1,'media','trimfquadrado',[56.5 90.1 112.8 m(i)]);
  a(i)=addmf(a(i),'input',1,'alta','trapmfquad',[88.81 119 320 475 m(i)]);
%    figure;
%    plotmf(a(i),'input',1)

Name='populacao'
Range=[0 300];
NumMFs=4;
MF1='muitobaixa':'trapmfquad',[-230 -30.3 8.929 34.7 m];
MF2='baixa':'trimfquadrado',[21.49 44.6 65 m];
MF3='media':'trimfquadrado',[56.5 90.1 112.8 m];
MF4='alta':'trapmfquad',[88.81 119 320 475 m];


%***************************************************
% Funções de Pertinência de Saída
%***************************************************

a(i)=addvar(a(i),'output','dx/dt',[0 108]);
  a(i)=addmf(a(i),'output',1,'muitobaixa','trapmf',[-100.4 -13.93 2.711 13.28 m(i)]);
  a(i)=addmf(a(i),'output',1,'baixa','trimfquadrado',[10 17.86 24.43 m(i)]);
  a(i)=addmf(a(i),'output',1,'media','trimfquadrado',[20.52 33.05 41.05 m(i)]);
  a(i)=addmf(a(i),'output',1,'alta','trapmf',[36.45 49 120.5 177.8 m(i)]);
%    figure;
%    plotmf(a(i),'output',1);

Name='dx/dt'
Range=[0 108];
NumMFs=4;
MF1='muitobaixa':'trapmf',[-100.4 -13.93 2.711 13.28 m];
MF2='baixa':'trimfquadrado',[10 17.86 24.43 m];
MF3='media':'trimfquadrado',[20.52 33.05 41.05 m];
MF4='alta':'trapmf',[36.45 49 120.5 177.8 m];


%***************************************************
% Regras
%***************************************************
rulelist=[
1 1 1  1
2 2 1  1
3 3 1  1
4 4 1  1];

a(i)=addrule(a(i),rulelist);

showrule(a(i));
out=evalfis([x],a(i));


   z=z+1;
```

```
   out1(z)=out;

   if (vetorx(z-1)==0)
      x=0;
   else
      x= x+ ((out1(z-1)+out1(z))*h)/2;
   end
      if (x<=0)
      x=0;
      vetorx(z)=0;
   end
     vetorx(z)=x;
  end

%*********************************************************
% Solução Determinística do Modelo de Malthus
%*********************************************************
t=0:h:(n*h);
h1=0:0.1:10;
g=2*exp(0.47*h1);
figure;
plot(t,vetorx,'.r',h1,g,'b')
legend('solução p-fuzzy modificada no tempo','solução deterministica')


%*********************************************************
% Critério de Escolha do Erro
%*********************************************************

erro=max(abs(vetorx-g'))
erro1=max(abs(vetorx-g'))./min(abs(vetorx))

v(i)=erro;
v1(i)=erro1;
```

B.2 Modelo de Verhulst

Rotina Computacional para o Sistema p-Fuzzy

```
clear all
close all

%*****************************************************
% População Inicial – x
%*****************************************************
x=20);

%*****************************************************
% Iterações – n
%*****************************************************
n=500;

t=0;
out=0;
out1=zeros(n+1,1);
vetorx=zeros(n+1,1);
x1=0;
out1(1)=x1;
vetorx(1)=x;
z=1;
h=1;
for i=1:n
  fismat=readfis('verhust1');
   out=evalfis([x],fismat);
   z=z+1;
   out1(z)=out;

   if (vetorx(z-1)==0)
      x=0;
   else
      x= x+ ((out1(z-1)+out1(z))*h)/2;
   end
      if (x<=0)
      x=0;
      vetorx(z)=0;
   end
     vetorx(z)=x;
  end
t=0:h:(n*h);
figure;
plot(t,vetorx,'.r')
```

Rotina Computacional para o Sistema p-Fuzzy Modificado

```
clear all
close all

%************************************************************
% Potência
%************************************************************
%m=0.1:0.1:1.9;
m=1.05:0.01:1.15;

for j=1:length(m)

%************************************************************
% População Inicial – x
%************************************************************
x=20;

%************************************************************
% Iterações – n
%************************************************************
n=500;

t=0;
out=0;
out1=zeros(n+1,1);
vetorx=zeros(n+1,1);
x1=0;
out1(1)=x1;
vetorx(1)=x;
z=1;
h=1;

   for i=1:n

%************************************************************
% SBRF
%************************************************************
a(j)=newfis('verhustmodificado');
 getfis(a(j))

Name='verhustmodificado'
Type='mamdani'
Version=2.0
NumInputs=1
NumOutputs=1
NumRules=6
AndMethod='min'
OrMethod='max'
ImpMethod='min'
AggMethod='max'
DefuzzMethod='centroid'
```

```
%****************************************************************
% Funções de Pertinência de Entrada
%****************************************************************
a(j)=addvar(a(j),'input','populacao',[0 300]);
  a(j)=addmf(a(j),'input',1,'mediabaixa','trimfquadrado1',[25 75 125 m(j)]);
  a(j)=addmf(a(j),'input',1,'mediaalta','trimfquadrado1',[125 175 225 m(j)]);
  a(j)=addmf(a(j),'input',1,'media','trimfquadrado1',[75 125 175 m(j)]);
  a(j)=addmf(a(j),'input',1,'alta','trimfquadrado1',[175 225 275 m(j)]);
  a(j)=addmf(a(j),'input',1,'baixa','trapmfquad2',[-30.41 -13.77 30 75 m(j)]);
  a(j)=addmf(a(j),'input',1,'altissima','trapmfquad2',[225 275 320 350 m(j)]);

Name='populacao'
Range=[0 300]
NumMFs=6
MF1='mediabaixa':'trimfquadrado1',[25 75 125 m]
MF2='mediaalta':'trimfquadrado1',[125 175 225 m]
MF3='media':'trimfquadrado1',[75 125 175 m]
MF4='alta':'trimfquadrado1',[175 225 275 m]
MF5='baixa':'trapmfquad2',[-30.41 -13.77 30 75 m]
MF6='altissima':'trapmfquad2',[225 275 320 350 m]

%****************************************************************
% Funções de Pertinência de Saída
%****************************************************************
a(j)=addvar(a(j),'output','dx/dt',[-1 2]);
  a(j)=addmf(a(j),'output',1,'altapositiva','trapmf',[1 1.5 2 3 m(j)]);
  a(j)=addmf(a(j),'output',1,'mediapositiva','trimf',[0.5 1 1.5 m(j)]);
  a(j)=addmf(a(j),'output',1,'baixanegativa','trapmf',[-0.8 -0.3 0 0 m(j)]);
  a(j)=addmf(a(j),'output',1,'baixapositiva','trapmf',[0 0 0.3 1 m(j)]);

Name='dx/dt'
Range=[-1 2]
NumMFs=4
MF1='altapositiva':'trapmf',[1 1.5 2 3 m]
MF2='mediapositiva':'trimf',[0.5 1 1.5 m]
MF3='baixanegativa':'trapmf',[-0.8 -0.3 0 0 m]
MF4='baixapositiva':'trapmf',[0 0 0.3 1 m]

%*********************************************************
% Regras
%*********************************************************
rulelist=[
5 4 1  1
1 2 1  1
3 1 1  1
2 2 1  1
4 4 1  1
6 3 1  1];

a(j)=addrule(a(j),rulelist);

showrule(a(j));
out=evalfis([x],a(j));

   z=z+1;
   out1(z)=out;

   if (vetorx(z-1)==0)
```

```
        x=0;
    else
        x= x+ ((out1(z-1)+out1(z))*h)/2;
    end
        if (x<=0)
        x=0;
        vetorx(z)=0;
    end
       vetorx(z)=x;
    end
t=0:h:(n*h);

%*****************************************************************
% Solução Determinística do Modelo de Verhulst
%*****************************************************************
p0=20;
p00=255.3447;
r=0.01785;
t1=0:1:500;
g=(p00*p0)./((p00-p0).*exp(-r*t1)+p0);
figure;
plot(t,vetorx,'.r',t1,g,'b')
legend('solução p-fuzzy','solução deterministica')
title(strcat('potencia = ', num2str(m(j))));

figure;
plotmf(a(j),'input',1)
figure;
plotmf(a(j),'output',1)

%*******************************************************
% Critérios para Escolha do Erro
%*******************************************************
erro=max(abs(vetorx-g'))
erro1=(max(abs(vetorx-g')))./min(abs(vetorx))

v(j)=erro;
v1(j)=erro1;

end

disp('erro')
v

disp('erro1')
 v1

k=min(v)
k1=min(v1)

kk=find(v==k);
kk1=find(v1==k1);

m(kk)
m(kk1)
```

B.3 Modelo de Montroll

Rotina Computacional para o Sistema p-Fuzzy

```
clear all
close all

%************************************************************
% População Inicial – x
%************************************************************
x=5;

%************************************************************
% Iterações – n
%************************************************************
n=450;

t=0;
out=0;
out1=zeros(n+1,1);
vetorx=zeros(n+1,1);
x1=0;
out1(1)=x1;
vetorx(1)=x;
z=1;
h=0.2;
for i=1:n

%************************************************************
% SBRF – montroll – ao final do programa
%************************************************************
  fismat=readfis('montroll');
   out=evalfis([x],fismat);
   z=z+1;
   out1(z)=out;

   if (vetorx(z-1)==0)
      x=0;
   else
      x= x+ ((out1(z-1)+out1(z))*h)/2;
   end
      if (x<=0)
      x=0;
      vetorx(z)=0;
   end
     vetorx(z)=x;
  end
t=0:h:(n*h);
figure;
```

```
plot(t,vetorx,'.r')
hold on

%************************************************************
% Solução Determinística do Modelo de Montroll
%************************************************************
e=0:0.2:90;
alpha12=1.2;
lambda=0.1;
pi=204;
p0=5;

for i=1:length(e)

r12(i)=pi*(exp(alpha12*lambda*e(i))/((exp(alpha12*lambda*e(i)))+((pi/p0)^alpha12)-1))^(1/alpha12);

end

for i=1:length(t)
   p(i)=find(t==t(i));

    w12(i)=r12(p(i));

end

plot(t,w12)
legend('p-fuzzy','\alpha=1.2')


%************************************************************
% SBRF – montroll
%************************************************************

% Name='montroll'
% Type='mamdani'
% Version=2.0
% NumInputs=1
% NumOutputs=1
% NumRules=6
% AndMethod='min'
% OrMethod='max'
% ImpMethod='min'
% AggMethod='max'
% DefuzzMethod='centroid'

% [Input1]
% Name='populacao'
% Range=[0 240]
% NumMFs=6
% MF1='baixa':'trimf',[24.4 50.4761904761905 77.1]
% MF2='mediaalta':'trimf',[109.444444444444 140.444444444444 166.031746031746]
% MF3='media':'trimf',[55.6 84.1269841269841 117]
% MF4='alta':'trimf',[156 196 219.365079365079]
% MF5='muitobaixa':'trapmf',[-24.3 -11 10.5 43.4920634920635]
% MF6='altissima':'trapmf',[200 223.174603174603 256 280]
%

% [Output1]
```

```
% Name='dx/dt'
% Range=[-4 12]
% NumMFs=4
% MF1='altapos':'trapmf',[6 9 12 14]
% MF2='mediapos':'trimf',[2 5 8]
% MF3='baixaneg':'trimf',[-3 0 0]
% MF4='baixapos':'trimf',[0 0 3]
%
% [Rules]
% 5, 4 (1) : 1
% 1, 2 (1) : 1
% 3, 1 (1) : 1
% 2, 2 (1) : 1
% 4, 4 (1) : 1
% 6, 3 (1) : 1
```

Rotina Computacional para o Sistema p-Fuzzy Modificado

```
clear all
close all

%************************************************
% Potência
%************************************************
%m=0.1:0.1:1.9;
m=0.75:0.01:0.85;

for j=1:length(m)

%************************************************
% População Inicial – x
%************************************************
x=5;


%************************************************
% Iterações – n
%************************************************
n=450;

t=0;
out=0;
out1=zeros(n+1,1);
vetorx=zeros(n+1,1);
x1=0;
out1(1)=x1;
vetorx(1)=x;
z=1;
h=0.2;

for i=1:n
```

```
a(j)=newfis('montrollmodificado');
 getfis(a(j))


Name='montrollmodificado'
Type='mamdani'
Version=2.0
NumInputs=1
NumOutputs=1
NumRules=6
AndMethod='min'
OrMethod='max'
ImpMethod='min'
AggMethod='max'
DefuzzMethod='centroid'


%*******************************************************
% Funções de Pertinência de Entrada
%*******************************************************
a(j)=addvar(a(j),'input','populacao',[0 240]);
   a(j)=addmf(a(j),'input',1,'baixa','trimfquadrado1',[24.4 50.4761904761905 77.1 m(j)]);
   a(j)=addmf(a(j),'input',1,'mediaalta','trimfquadrado1',[109.444444444444 140.444444444444
166.031746031746 m(j)]);
   a(j)=addmf(a(j),'input',1,'media','trimfquadrado1',[55.6 84.1269841269841 117 m(j)]);
   a(j)=addmf(a(j),'input',1,'alta','trimfquadrado',[156 196 219.365079365079 m(j)]);
   a(j)=addmf(a(j),'input',1,'muitobaixa','trapmfquad1',[-24.3 -11 10.5 43.4920634920635 m(j)]);
   a(j)=addmf(a(j),'input',1,'altissima','trapmfquad1',[200 223.174603174603 256 280 m(j)]);

Name='populacao'
Range=[0 240]
NumMFs=6
MF1='baixa':'trimfquadrado1',[24.4 50.4761904761905 77.1 m]
MF2='mediaalta':'trimfquadrado1',[109.444444444444 140.444444444444 166.031746031746 m]
MF3='media':'trimfquadrado1',[55.6 84.1269841269841 117 m]
MF4='alta':'trimfquadrado1',[156 196 219.365079365079 m]
MF5='muitobaixa':'trapmfquad1',[-24.3 -11 10.5 43.4920634920635 m]
MF6='altissima':'trapmfquad1',[200 223.174603174603 256 280 m]


%*******************************************************
% Funções de Pertinência de Saída
%*******************************************************
a(j)=addvar(a(j),'output','dx/dt',[-4 12]);
   a(j)=addmf(a(j),'output',1,'altapositiva','trapmfquad',[6 9 12 14 m(j)]);
   a(j)=addmf(a(j),'output',1,'mediapositiva','trimfquadrado',[2 5 8 m(j)]);
   a(j)=addmf(a(j),'output',1,'baixanegativa','trimfquadrado',[-3 0 0 m(j)]);
   a(j)=addmf(a(j),'output',1,'baixapositiva','trimfquadrado',[0 0 3 m(j)]);

Name='dx/dt'
Range=[-4 12]
NumMFs=4
MF1='altapositiva':'trapmfquad',[6 9 12 14 m]
MF2='mediapositiva':'trimfquadrado',[2 5 8 m]
MF3='baixanegativa':'trimfquadrado',[-3 0 0 m]
MF4='baixapositiva':'trimfquadrado',[0 0 3 m]
```

```
%*********************************************************
% Regras
%*********************************************************
rulelist=[
5 4 1  1
1 2 1  1
3 1 1  1
2 2 1  1
4 4 1  1
6 3 1  1];

a(j)=addrule(a(j),rulelist);

showrule(a(j));

out=evalfis([x],a(j));

z=z+1;
   out1(z)=out;
   if (vetorx(z-1)==0)
      x=0;
   else
      x= x+ ((out1(z-1)+out1(z))*h)/2;
   end
      if (x<=0)
      x=0;
      vetorx(z)=0;
   end
     vetorx(z)=x;
  end
t=0:h:(n*h);

%*************************************************************
% Solução Analítica para o Modelo de Montroll
%*************************************************************
e=0:0.2:90;
alpha12=1.2;
lambda=0.1;
pi=204;
p0=5;

for i=1:length(e)

   r12(i)=pi*(exp(alpha12*lambda*e(i))/((exp(alpha12*lambda*e(i)))+((pi/p0)^alpha12)-1))^(1/alpha12);

end

for i=1:length(t)
   p(i)=find(t==t(i));

   w12(i)=r12(p(i));

end

%*************************************************************
% Critérios para Escolha do Erro
%*************************************************************
erro=max(abs(w12-vetorx'))
```

```
erro1=(max(abs(w12-vetorx'))).\min(abs(w12))

v(j)=erro
v1(j)=erro1

figure;
plot(t,vetorx,'.r',t,w12,'b')
legend('solução p-fuzzy','solução deterministica')
title(strcat('potencia = ', num2str(m(j))));

end

disp('erro')
v
disp('erro1')
 v1

k=min(v)
k1=min(v1)

kk=find(v==k);
kk1=find(v1==k1);

m(kk)
m(kk1)
```

B.4 Modelo da AIDS

Rotina Computacional para o Sistema p-Fuzzy

```
clear all
close all

%*********************************************************
% População Inicial – x
%*********************************************************
x=0.001;



%*********************************************************
% Iterações – n
%*********************************************************
n=800;
```

```
t=0;
out=0;
out1=zeros(n+1,1);
vetorx=zeros(n+1,1);
x1=0;
out1(1)=x1;
vetorx(1)=x;
z=1;
h=0.01;
for i=1:n

%*********************************************************
% SBRF – aids – descrito no final do programa
%*********************************************************
  fismat=readfis('aids');
   out=evalfis([x],fismat);
   z=z+1;
   out1(z)=out;

   if (vetorx(z-1)==0)
      x=0;
   else
      x= x+ ((out1(z-1)+out1(z))*h)/2;
   end
      if (x<=0)
      x=0;
      vetorx(z)=0;
   end
     vetorx(z)=x;
  end
t=0:h:(n*h);
figure;
plot(t,vetorx,'.r',t1,w,'*b')


%*********************************************************
% Solução Determinística – Dados de Peterman (1985)
%*********************************************************
h1=0:0.1:7;
for i=1:length(h)
    pd(i)=exp(-(0.237*(h1(i)^2))/2);
    p1(i)=1-pd(i);
    p2(i)=0.237*h1(i)*pd(i);
end

 t1=[0.6 1.2 1.5 2.1 2.6 3.1 3.6 4 4.5 5 5.5];
 z=[0.048 0.078 0.172 0.115 0.155 0.088 0.155 0.038 0.08 0.018 0.04];
 w(1)=0.048;
 for i=1:(length(z)-1)
    w(i+1)=w(i)+z(i+1);
 end


 w1=ones(1,length(z))-w;

t=0:h:(n*h);
figure;
```

```
vetorx=vetorx-0.001;
subplot(2,1,2);
plot(t,vetorx);
hold on
plot(t1,w,'*r');
nn=1-vetorx;
subplot(2,1,1);
plot(t,nn,t1,w1,'*r')


%************************************************
% SBRF
%************************************************
% Name='aids'
% Type='mamdani'
% Version=2.0
% NumInputs=1
% NumOutputs=1
% NumRules=6
% AndMethod='min'
% OrMethod='max'
% ImpMethod='min'
% AggMethod='max'
% DefuzzMethod='centroid'

% [Input1]
% Name='populacao'
% Range=[0.001 1.1]
% NumMFs=6
% MF1='mediabaixa':'trimf',[0.1129 0.2758 0.4586]
% MF2='mediaalta':'trimf',[0.459 0.642 0.8253]
% MF3='media':'trimf',[0.2758 0.459 0.642]
% MF4='alta':'trimf',[0.642 0.8253 1.009]
% MF5='baixa':'trapmf',[-0.1104 -0.04944 0.1109 0.2758]
% MF6='altissima':'trapmf',[0.8253 1.009 1.174 1.284]

% [Output1]
% Name='dy/dt'
% Range=[-0.2 1]
% NumMFs=4
% MF1='altapos':'trimf',[0.401 0.447435897435898 0.483]
% MF2='mediapos':'trimf',[0.204 0.326 0.457692307692308]
% MF3='baixaneg':'trapmf',[-0.16 -0.03462 0 0]
% MF4='baixapos':'trapmf',[0.00256410256410256 0.00256410256410256 0.22948717948718
0.298564102564103]

% [Rules]
% 5, 4 (1) : 1
% 1, 2 (1) : 1
% 3, 1 (1) : 1
% 2, 2 (1) : 1
% 4, 4 (1) : 1
% 6, 3 (1) : 1
```

Rotina Computacional para o Sistema p-Fuzzy Modificado

```
clear all
close all

%************************************************
% Potência
%************************************************
%m=0.1:0.1:1.9;
m=1.05:0.01:1.15;

for j=1:length(m)

%************************************************
% População Inicial – x
%************************************************
x=0.001;

%************************************************
% Iterações – n
%************************************************
n=800;

t=0;
out=0;
out1=zeros(n+1,1);
vetorx=zeros(n+1,1);
x1=0;
out1(1)=x1;
vetorx(1)=x;
z=1;
h=0.01;

for i=1:n


%************************************************
% SBRF
%************************************************
   a(j)=newfis('aidsmodificado');
 getfis(a(j))

 Name='aidsmodificado'
Type='mamdani'
Version=2.0
NumInputs=1
NumOutputs=1
NumRules=6
AndMethod='min'
OrMethod='max'
ImpMethod='min'
AggMethod='max'
DefuzzMethod='centroid'
```

```
%************************************************
% Funções de Pertinência de Entrada
%************************************************
a(j)=addvar(a(j),'input','populacao',[0.001 1.1]);
   a(j)=addmf(a(j),'input',1,'mediabaixa','trimfquadrado',[0.1129 0.2758 0.4586 m(j)]);
   a(j)=addmf(a(j),'input',1,'mediaalta','trimfquadrado',[0.459 0.642 0.8253 m(j)]);
   a(j)=addmf(a(j),'input',1,'media','trimfquadrado',[0.2758 0.459 0.642 m(j)]);
   a(j)=addmf(a(j),'input',1,'alta','trimfquadrado',[0.642 0.8253 1.009 m(j)]);
   a(j)=addmf(a(j),'input',1,'baixa','trapmfquad',[-0.1104 -0.04944 0.1109 0.2758 m(j)]);
   a(j)=addmf(a(j),'input',1,'altissima','trapmfquad',[0.8253 1.009 1.174 1.284 m(j)]);

Name='populacao'
Range=[0.001 1.1]
NumMFs=6
MF1='mediabaixa':'trimfquadrado',[0.1129 0.2758 0.4586 m]
MF2='mediaalta':'trimfquadrado',[0.459 0.642 0.8253 m]
MF3='media':'trimfquadrado',[0.2758 0.459 0.642 m]
MF4='alta':'trimfquadrado',[0.642 0.8253 1.009 m]
MF5='baixa':'trapmfquad',[-0.1104 -0.04944 0.1109 0.2758 m]
MF6='altissima':'trapmfquad',[0.8253 1.009 1.174 1.284 m]


%************************************************
% Funções de Pertinência de Saída
%************************************************
a(j)=addvar(a(j),'output','dx/dt',[-0.2 1]);
   a(j)=addmf(a(j),'output',1,'mediapositiva','trimfquadrado1',[0.204 0.326 0.457692307692308 m(j)]);
   a(j)=addmf(a(j),'output',1,'baixanegativa','trapmfquad',[-0.16 -0.03462 0 0 m(j)]);
   a(j)=addmf(a(j),'output',1,'baixapositiva','trapmfquad',[0.00256410256410256 0.00256410256410256
0.22948717948718 0.298564102564103 m(j)]);
   a(j)=addmf(a(j),'output',1,'altapositiva','trapmfquad',[0.401 0.4474 1 2 m(j)]);

Name='dy/dt'
Range=[-0.2 1]
NumMFs=4
MF1='mediapositiva':'trimfquadrado1',[0.204 0.326 0.457692307692308 m]
MF2='baixanegativa':'trapmfquad',[-0.16 -0.03462 0 0 m]
MF3='baixapositiva':'trapmfquad',[0.00256410256410256 0.00256410256410256 0.22948717948718
0.298564102564103 m]
MF4='altapositiva':'trapmfquad',[0.401 0.4474 1 2 m]


%************************************************
% Regras
%************************************************
rulelist=[
5 3 1  1
1 1 1  1
2 1 1  1
4 3 1  1
6 2 1  1];

a(j)=addrule(a(j),rulelist);

showrule(a(j));

out=evalfis([x],a(j));
```

```
  z=z+1;
   out1(z)=out;

   if (vetorx(z-1)==0)
      x=0;
   else
      x= x+ ((out1(z-1)+out1(z))*h)/2;
   end
      if (x<=0)
      x=0;
      vetorx(z)=0;
   end
     vetorx(z)=x;
  end
t=0:h:(n*h);


%*********************************************************
% Solução Determinística – Dados de Peterman (1985)
%*********************************************************
h1=0:0.1:7;

for i=1:length(h)
    pd(i)=exp(-(0.237*(h1(i)^2))/2);
    p1(i)=1-pd(i);
    p2(i)=0.237*h1(i)*pd(i);
end

 t1=[0.6 1.2 1.5 2.1 2.6 3.1 3.6 4 4.5 5 5.5];
 z=[0.048 0.078 0.172 0.115 0.155 0.088 0.155 0.038 0.08 0.018 0.04];
 w(1)=0.048;
 for i=1:(length(z)-1)
    w(i+1)=w(i)+z(i+1);
 end

 w1=ones(1,length(z))-w;

t=0:h:(n*h);

for i=1:length(t1)
   p(i)=find(t==t1(i));
   w1(i)=vetorx(p(i));
end

figure;
plot(t,vetorx,'.r',t1,w,'*b')
legend('solução p-fuzzy modificado','solução deterministica')
title(strcat('potencia = ', num2str(m(j))));

%*****************************************************
% Critérios para Escola do Erro
%*****************************************************
erro=max(abs(w1-w))
erro1=(max(abs(w1-w)))./min(abs(w1))


v(j)=erro;
```

```
v1(j)=erro1;

end

disp('erro')
v
disp('erro1')
v1

k=min(v)
k1=min(v1)

kk=find(v==k);
kk1=find(v1==k1);

m(kk)
m(kk1)
```

B.5 Modelo Presa-Predador

Rotina Computacional para o Sistema p-Fuzzy

```
clear all
close all

%************************************************************
% População Inicial das Presas e Predadores – x e y
%************************************************************
x=100;
y=5;

%************************************************************
% Iterações – n
%************************************************************
n=460;

t=0;
out=0;
out1=zeros(n+1,1);
out2=zeros(n+1,1);
vetorx=zeros(n+1,1);
vetory=zeros(n+1,1);
x1=0;
x2=0;
out1(1)=x1;
```

```
out2(1)=x2;
vetorx(1)=x;
vetory(1)=y;
z=1;
h=1;
for i=1:n

%************************************************************
% SBRF – presapredador – Descrito no final do programa
%************************************************************
  fismat=readfis('presapredador');
   out=evalfis([x' y'],fismat);
   z=z+1;
   out1(z)=x*out(1);
   out2(z)=y*out(2);
   if (vetorx(z-1)==0)
      x=0;
   else
      x= x+ out1(z-1)*h;
   end
   if (vetory(z-1)==0)
      y=0;
   else
      y=y+ out2(z-1)*h;
   end
   if (x<=0)
      x=0;
      vetorx(z)=0;
   end
   if (y<=0)
      y=0;
      vetory(z)=0;
   end
   vetorx(z)=x;
       vetory(z)=y;
   end
t=0:h:(n*h);
figure;
plot(t,vetorx,'*r')
hold on
plot(t,vetory,'*r')

%****************************************
% Critério para escolha do Erro
%****************************************
erro1=max(abs(vetorx-y(:,1)));
erro2=max(abs(vetory-y(:,2)));

v(j)=erro1;
w(j)=erro2;

r=v+w
```

```
%************************************************
% SBRF
%************************************************
% Name='presapredador'
% Type='mamdani'
% Version=2.0
% NumInputs=2
% NumOutputs=2
% NumRules=16
% AndMethod='min'
% OrMethod='max'
% ImpMethod='min'
% AggMethod='max'
% DefuzzMethod='centroid'

% [Input1]
% Name='X'
% Range=[0 250]
% NumMFs=4
% MF1='A2':'trimf',[10 40 125]
% MF2='A3':'trimf',[40 125 217.261904761905]
% MF3='A1':'trapmf',[-225 -25 6.2830687830688 14.9]
% MF4='A4':'trapmf',[145.171957671958 244 259 269]

% [Input2]
% Name='Y'
% Range=[0 15]
% NumMFs=4
% MF1='B1':'trimf',[-6 0 5.357]
% MF2='B3':'trimf',[5.357 10.29 15]
% MF3='B4':'trimf',[10.29 15 19.29]
% MF4='B2':'trimf',[0 5.357 10.29]
%
% [Output1]
% Name='1/x(dx/dt)'
% Range=[-0.05 0.05]
% NumMFs=4
% MF1='N1':'trimf',[-0.04545 0 0]
% MF2='P1':'trimf',[0 0 0.05]
% MF3='N2':'trapmf',[-0.0684 -0.0594 -0.0437830687830688 -0.000265]
% MF4='P2':'trapmf',[0 0.04091 0.05909 0.06818]
%
% [Output2]
% Name='1/y(dy/dt)'
% Range=[-0.03 0.04]
% NumMFs=4
% MF1='N1':'trimf',[-0.0302777777777778 0.00481 0.00481]
% MF2='P1':'trimf',[0.005 0.005 0.038]
% MF3='N2':'trapmf',[-0.05 -0.04 -0.0293518518518518 0.005]
% MF4='P2':'trapmf',[0.005 0.0391666666666667 0.047 0.103]

% [Rules]
% 3 1, 4 3 (1) : 1
% 1 1, 4 1 (1) : 1
% 2 1, 4 2 (1) : 1
% 4 1, 4 4 (1) : 1
% 3 4, 2 3 (1) : 1
% 1 4, 2 1 (1) : 1
```

```
% 2 4, 2 2 (1) : 1
% 4 4, 2 4 (1) : 1
% 3 2, 1 3 (1) : 1
% 1 2, 1 1 (1) : 1
% 2 2, 1 2 (1) : 1
% 4 2, 1 4 (1) : 1
% 3 3, 3 3 (1) : 1
% 1 3, 3 1 (1) : 1
% 2 3, 3 2 (1) : 1
% 4 3, 3 4 (1) : 1
```

Rotina Computacional para o Sistema p-Fuzzy Modificado

```
clear all
close all

%*************************************************************
% Potência
%*************************************************************
%m=0.1:0.1:1.9;
%m=0.75:0.01:0.85;
m=0.755:0.001:0.765;

for j=1:length(m)


%*************************************************************
% População Inicial das Presas e Predadores – s e y
%*************************************************************
x=100;
y=5;

%*************************************************************
% Iterações – n
%*************************************************************
n=460;

t=0;
out=0;
out1=zeros(n+1,1);
out2=zeros(n+1,1);
vetorx=zeros(n+1,1);
vetory=zeros(n+1,1);
x1=0;
x2=0;
out1(1)=x1;
out2(1)=x2;
vetorx(1)=x;
vetory(1)=y;
z=1;
h=1;
```

```
for i=1:n

a(j)=newfis('presapredadormodificado');
 getfis(a(j))

Name='presapredadormodificado'
Type='mamdani'
Version=2.0
NumInputs=2
NumOutputs=2
NumRules=16
AndMethod='min'
OrMethod='max'
ImpMethod='min'
AggMethod='max'
DefuzzMethod='centroid'

%************************************************************
% Funções de Pertinência de Entrada
%************************************************************
a(j)=addvar(a(j),'input','presa',[0 250]);
   a(j)=addmf(a(j),'input',1,'A2','trimfquadrado1',[10 40 125 m(j)]);
   a(j)=addmf(a(j),'input',1,'A3','trimfquadrado1',[40 125 217.261904761905 m(j)]);
   a(j)=addmf(a(j),'input',1,'A1','trapmfquad2',[-225 -25 6.2830687830688 14.9 m(j)]);
   a(j)=addmf(a(j),'input',1,'A4','trapmfquad2',[145.171957671958 244 259 269 m(j)]);

Name='presa'
Range=[0 250]
NumMFs=4
MF1='A2':'trimfquadrado1',[10 40 125 m]
MF2='A3':'trimfquadrado1',[40 125 217.261904761905 m]
MF3='A1':'trapmfquad2',[-225 -25 6.2830687830688 14.9 m]
MF4='A4':'trapmfquad2',[145.171957671958 244 259 269 m]

a(j)=addvar(a(j),'input','predador',[0 15]);
   a(j)=addmf(a(j),'input',2,'B1','trimfquadrado1',[-6 0 5.357 m(j)]);
   a(j)=addmf(a(j),'input',2,'B3','trimfquadrado1',[5.357 10.29 15 m(j)]);
   a(j)=addmf(a(j),'input',2,'B4','trimfquadrado1',[10.29 15 19.29 m(j)]);
   a(j)=addmf(a(j),'input',2,'B2','trimfquadrado1',[0 5.357 10.29 m(j)]);

Name='predador'
Range=[0 15]
NumMFs=4
MF1='B1':'trimfquadrado1',[-6 0 5.357 m]
MF2='B3':'trimfquadrado1',[5.357 10.29 15 m]
MF3='B4':'trimfquadrado1',[10.29 15 19.29 m]
MF4='B2':'trimfquadrado1',[0 5.357 10.29 m]

%************************************************************
% Funções de Pertinência de Saída
%************************************************************
a(j)=addvar(a(j),'output','(1/x)(dx/dt)',[-0.05 0.05]);
   a(j)=addmf(a(j),'output',1,'N1','trimf',[-0.04545 0 0 m(j)]);
   a(j)=addmf(a(j),'output',1,'P1','trimf',[0 0 0.05 m(j)]);
   a(j)=addmf(a(j),'output',1,'N2','trapmf',[-0.0684 -0.0594 -0.0437830687830688 -0.000265 m(j)]);
   a(j)=addmf(a(j),'output',1,'P2','trapmf',[0 0.04091 0.05909 0.06818 m(j)]);
```

```
Name='(1/x)(dx/dt)'
Range=[-0.05 0.05]
NumMFs=4
MF1='N1':'trimf',[-0.04545 0 0 m]
MF2='P1':'trimf',[0 0 0.05 m]
MF3='N2':'trapmf',[-0.0684 -0.0594 -0.0437830687830688 -0.000265 0 m]
MF4='P2':'trapmf',[0 0.04091 0.05909 0.06818 m]

a(j)=addvar(a(j),'output','(1/y)(dy/dt)',[-0.03 0.04]);
   a(j)=addmf(a(j),'output',2,'N1','trimf',[-0.025 0.005 0.005 m(j)]);
   a(j)=addmf(a(j),'output',2,'P1','trimf',[0.005 0.005 0.038 m(j)]);
   a(j)=addmf(a(j),'output',2,'N2','trapmf',[-0.05 -0.04 -0.025 0.005 m(j)]);
   a(j)=addmf(a(j),'output',2,'P2','trapmf',[0.005 0.037 0.047 0.103 m(j)]);

Name='(1/y)(dy/dt)'
Range=[-0.03 0.04]
NumMFs=4
MF1='N1':'trimf',[-0.025 0.005 0.005 m]
MF2='P1':'trimf',[0.005 0.005 0.038 m]
MF3='N2':'trapmf',[-0.05 -0.04 -0.025 0.005 m]
MF4='P2':'trapmf',[0.005 0.037 0.047 0.103 m]

%**************************************************
% Regras
%**************************************************
rulelist=[
3 1 4 3 1  1
1 1 4 1 1  1
2 1 4 2 1  1
4 1 4 4 1  1
3 4 2 3 1  1
1 4 2 1 1  1
2 4 2 2 1  1
4 4 2 4 1  1
3 2 1 3 1  1
1 2 1 1 1  1
2 2 1 2 1  1
4 2 1 4 1  1
3 3 3 3 1  1
1 3 3 1 1  1
2 3 3 2 1  1
4 3 3 4 1  1];

a(j)=addrule(a(j),rulelist);

showrule(a(j));

out=evalfis([x y],a(j));
    z=z+1;
    out1(z)=x*out(1);
    out2(z)=y*out(2);
    if (vetorx(z-1)==0)
        x=0;
    else
        x= x+ out1(z-1)*h;
    end
    if (vetory(z-1)==0)
        y=0;
```

```
    else
        y=y+ out2(z-1)*h;
    end
    if (x<=0)
        x=0;
        vetorx(z)=0;
    end
    if (y<=0)
        y=0;
        vetory(z)=0;
    end
    vetorx(z)=x;
         vetory(z)=y;
end
[t,y]=ode45('rosananovo',[0:1:460],[100;5]);
figure;
plot(t,y(:,1),t,y(:,2));
legend('presa','predador')
xlabel('iteracoes');
ylabel('populacao');
hold on
t=0:h:(n*h);
plot(t,vetorx,'*r')
plot(t,vetory,'*r')

%*****************************************************************
% Critérios para Escolha do Erro
%*****************************************************************
erro1=max(abs(vetorx-y(:,1)));
v(j)=erro1;

erro11a=(max(abs(vetorx-y(:,1))))./min(abs(vetorx));
v1(j)=erro11a;

erro2=max(abs(vetory-y(:,2)));
w(j)=erro2;

erro21b=(max(abs(vetory-y(:,2))))./min(abs(vetory));
w1(j)=erro21b;

r=v+w
r1=v1+w1

k=min(r)
k1=min(r1)

kk=find(r==k);
kk1=find(r1==k1);

m(kk)
m(kk1)

figure;
plot(y(:,1),y(:,2),'');
xlabel('presa');
ylabel('predador');

end
```

B.6 Modelo do Controle de Pragas

Rotina Computacional para o Sistema p-Fuzzy

```
% CRESCIMENTO POPULACIONAL INIBIDO P-FUZZY
% CASO UNIDIMENSIONAL
%*************************************************************************


function [p cont1] = ControleCaso1(x0,apl,K)

%*************************************************************************
%         DADOS DO SBRF – Descritos no final do Programa
%*************************************************************************

dados_cont=readfis('praga1');
dados_var=readfis('praga');

%*************************************************************************
%         DADOS INICIAIS
%*************************************************************************
if nargin==3
   p(1)=x0; %População Inicial
   aplic=apl;
   n=K; %iterações
elseif nargin==2
   p(1)=x0; %População Inicial
   aplic=apl; %Intervalo de aplicação (DIAS) = ');
   N=400; %iterações
elseif nargin==1
   p(1)=x0; %População Inicial
   aplic=15; % Intervalo de aplicação (DIAS) = ');
   n=400; % iterações
else
   p(1)=input('x0=>'); %População Inicial
   aplic=15; % Intervalo de aplicação (DIAS) = ');
   n=400; % iterações
end
p_noncont(1)=p(1);

%*************************************************************************
%         Controle Fuzzy
%*************************************************************************
k=1;
for i=1:n
   var(i)=evalfis(p(i),dados_var);
   var_noncont(i)=evalfis(p_noncont(i),dados_var);
   j=rem(i-1,aplic);
   if j==0;
```

```
        cont1(k)=evalfis([p(i),var(i)],dados_cont);
        p(i+1)=(p(i)+var(i))*(1-cont1(k));
        k=k+1;
    else
        p(i+1)=p(i)+ var(i);
    end
    p_noncont(i+1)=p_noncont(i)+var_noncont(i);
end

t=0:n;
figure(1)
plot(t,p,'k',t,p_noncont,':k')
xlabel('Iteração','fontsize',13);ylabel('Densidade Populacional','fontsize',13);
RangeY=dados_var.input(1).range;
axis([0 n RangeY]);
figure(2)
plot(cont1,'r')
RangeContY=dados_cont.output(1).range;
RangeContX=[1 (length(cont1)+2)];
axis([RangeContX RangeContY]);
xlabel('Iteração (j)');
ylabel('Controle (C)');

%********************************************************************************
% SBRF – praga
%********************************************************************************
% Name='praga'
% Type='mamdani'
% Version=2.0
% NumInputs=1
% NumOutputs=1
% NumRules=6
% AndMethod='min'
% OrMethod='max'
% ImpMethod='min'
% AggMethod='max'
% DefuzzMethod='centroid'

% [Input1]
% Name='x'
% Range=[0 300]
% NumMFs=6
% MF1='B':'trapmf',[-75 -25 25 75]
% MF2='MB':'trimf',[25 75 125]
% MF3='M':'trimf',[75 125 175]
% MF4='MA':'trimf',[125 175 225]
% MF5='A':'trimf',[175 225 275]
% MF6='ALT':'trapmf',[225 275 325 375]

% [Output1]
% Name='C'
% Range=[-1 2]
% NumMFs=4
% MF1='BN':'trapmf',[-1 -0.25 0 0]
% MF2='BP':'trapmf',[0 0 0.25 1]
% MF3='MP':'trimf',[0.5 1 1.5]
% MF4='AP':'trimf',[1 1.5 2]
%
```

```
% [Rules]
% 1, 2 (1) : 1
% 2, 3 (1) : 1
% 3, 4 (1) : 1
% 4, 3 (1) : 1
% 5, 2 (1) : 1
% 6, 1 (1) : 1


%************************************************************************
% SBRF – praga1
%************************************************************************
% Name='praga1'
% Type='mamdani'
% Version=2.0
% NumInputs=2
% NumOutputs=1
% NumRules=24
% AndMethod='min'
% OrMethod='max'
% ImpMethod='min'
% AggMethod='max'
% DefuzzMethod='centroid'

% [Input1]
% Name='x'
% Range=[0 300]
% NumMFs=6
% MF1='B':'trapmf',[-75 -25 25 75]
% MF2='MB':'trimf',[25 75 125]
% MF3='M':'trimf',[75 125 175]
% MF4='MA':'trimf',[125 175 225]
% MF5='A':'trimf',[175 225 275]
% MF6='ALT':'trapmf',[225 275 325 375]

% [Input2]
% Name='deltax'
% Range=[-1 2]
% NumMFs=4
% MF1='BN':'trapmf',[-1 -0.25 0 0]
% MF2='BP':'trapmf',[0 0 0.25 1]
% MF3='MP':'trimf',[0.5 1 1.5]
% MF4='AP':'trimf',[1 1.5 2]

% [Output1]
% Name='C'
% Range=[0 0.9]
% NumMFs=4
% MF1='C0':'trapmf',[-0.2 -0.1 0 0]
% MF2='CB':'trapmf',[0 0.21 0.25 0.43]
% MF3='CM':'trapmf',[0.245 0.43 0.48 0.625]
% MF4='CA':'trapmf',[0.45 0.65 1 1.4]
```

```
% [Rules]
% 1 1, 1 (1) : 1
% 2 1, 1 (1) : 1
% 3 1, 1 (1) : 1
% 4 1, 2 (1) : 1
% 5 1, 3 (1) : 1
% 6 1, 4 (1) : 1
% 1 2, 1 (1) : 1
% 2 2, 1 (1) : 1
% 3 2, 2 (1) : 1
% 4 2, 3 (1) : 1
% 5 2, 3 (1) : 1
% 6 2, 4 (1) : 1
% 1 3, 1 (1) : 1
% 2 3, 2 (1) : 1
% 3 3, 3 (1) : 1
% 4 3, 3 (1) : 1
% 5 3, 4 (1) : 1
% 6 3, 4 (1) : 1
% 1 4, 2 (1) : 1
% 2 4, 3 (1) : 1
% 3 4, 3 (1) : 1
% 4 4, 4 (1) : 1
% 5 4, 4 (1) : 1
% 6 4, 4 (1) : 1
```

Rotina Computacional para o Sistema p-Fuzzy Modificado

```
close all
clear all

%*************************************************************************
% CRESCIMENTO POPULACIONAL INIBIDO P-FUZZY MODIFICADO
% CASO UNIDIMENSIONAL
%*************************************************************************
% function [p cont] = ControleCaso1(x0,apl,K)

%*************************************************************************
%              VARIAÇÃO DA POTENCIA
%*************************************************************************
p(1)=input('x0=>'); %População Inicial
   aplic=15; % Intervalo de aplicação (DIAS) = ');
   n=400; % iterações

p_noncont(1)=p(1);

%m=0.1:0.1:1.9;
m=0.9:0.01:0.96;
```

```
for j=1:length(m)

%*********************************************************
% SBRF - C
%*********************************************************
a(j)=newfis('praga1');
 getfis(a(j))

Name='praga1'
Type='mamdani'
Version=2.0
NumInputs=2
NumOutputs=1
NumRules=24
AndMethod='min'
OrMethod='max'
ImpMethod='min'
AggMethod='max'
DefuzzMethod='centroid'

%*********************************************************************
% Funções de Pertinência de Entrada
%*********************************************************************
a(j)=addvar(a(j),'input','x',[0 300]);
   a(j)=addmf(a(j),'input',1,'B','trapmfquad',[-75 -25 25 75 m(j)]);
   a(j)=addmf(a(j),'input',1,'MB','trimfquadrado',[25 75 125 m(j)]);
   a(j)=addmf(a(j),'input',1,'M','trimfquadrado',[75 125 175 m(j)]);
   a(j)=addmf(a(j),'input',1,'MA','trimfquadrado',[125 175 225 m(j)]);
   a(j)=addmf(a(j),'input',1,'A','trimfquadrado',[175 225 275 m(j)]);
   a(j)=addmf(a(j),'input',1,'ALT','trapmfquad',[225 275 325 375 m(j)]);

Name='x'
Range=[0 300]
NumMFs=6
MF1='B':'trapmfquad',[-75 -25 25 75 m]
MF2='MB':'trimfquadrado',[25 75 125 m]
MF3='M':'trimfquadrado',[75 125 175 m]
MF4='MA':'trimfquadrado',[125 175 225 m]
MF5='A':'trimfquadrado',[175 225 275 m]
MF6='ALT':'trapmfquad',[225 275 325 375 m]

a(j)=addvar(a(j),'input','deltax',[-1 2]);
   a(j)=addmf(a(j),'input',2,'BN','trapmfquad',[-1 -0.25 0 0 m(j)]);
   a(j)=addmf(a(j),'input',2,'BP','trapmfquad',[0 0 0.25 1 m(j)]);
   a(j)=addmf(a(j),'input',2,'MP','trimfquadrado',[0.5 1 1.5 m(j)]);
   a(j)=addmf(a(j),'input',2,'AP','trimfquadrado',[1 1.5 2 m(j)]);

Name='deltax'
Range=[-1 2]
NumMFs=4
MF1='BN':'trapmfquad',[-1 -0.25 0 0 m]
MF2='BP':'trapmfquad',[0 0 0.25 1 m]
MF3='MP':'trimfquadrado',[0.5 1 1.5 m]
MF4='AP':'trimfquadrado',[1 1.5 2 m]
```

```
%******************************************************
% Funções de Pertinência de Saída
%******************************************************
a(j)=addvar(a(j),'output','C',[0 0.9]);
   a(j)=addmf(a(j),'output',1,'C0','trapmfquad',[-0.2 -0.1 0 0 m(j)]);
   a(j)=addmf(a(j),'output',1,'CB','trapmfquad',[0 0.21 0.25 0.43 m(j)]);
   a(j)=addmf(a(j),'output',1,'CM','trapmfquad',[0.245 0.43 0.48 0.625 m(j)]);
   a(j)=addmf(a(j),'output',1,'CA','trapmfquad',[0.45 0.65 1 1.4 m(j)]);

Name='C'
Range=[0 0.9]
NumMFs=4
MF1='C0':'trapmfquad',[-0.2 -0.1 0 0 m]
MF2='CB':'trapmfquad',[0 0.21 0.25 0.43 m]
MF3='CM':'trapmfquad',[0.245 0.43 0.48 0.625 m]
MF4='CA':'trapmfquad',[0.45 0.65 1 1.4 m]

%******************************************************
% Regras
%******************************************************
rulelist=[
1 1 1 1  1
2 1 1 1  1
3 1 1 1  1
4 1 2 1  1
5 1 3 1  1
6 1 4 1  1
1 2 1 1  1
2 2 1 1  1
3 2 2 1  1
4 2 3 1  1
5 2 3 1  1
6 2 4 1  1
1 3 1 1  1
2 3 2 1  1
3 3 3 1  1
4 3 3 1  1
5 3 4 1  1
6 3 4 1  1
1 4 2 1  1
2 4 3 1  1
3 4 3 1  1
4 4 4 1  1
5 4 4 1  1
6 4 4 1  1];

a(j)=addrule(a(j),rulelist);

showrule(a(j));
```

```
%************************************************
% SBRF – deltax
%************************************************
b(j)=newfis('praga');
 getfis(b(j))

Name='praga'
Type='mamdani'
Version=2.0
NumInputs=1
NumOutputs=1
NumRules=6
AndMethod='min'
OrMethod='max'
ImpMethod='min'
AggMethod='max'
DefuzzMethod='centroid'

%************************************************
% Funções de Pertinência de Entrada
%************************************************
b(j)=addvar(b(j),'input','x1',[0 300]);
   b(j)=addmf(b(j),'input',1,'B','trapmfquad',[-75 -25 25 75 m(j)]);
   b(j)=addmf(b(j),'input',1,'MB','trimfquadrado',[25 75 125 m(j)]);
   b(j)=addmf(b(j),'input',1,'M','trimfquadrado',[75 125 175 m(j)]);
   b(j)=addmf(b(j),'input',1,'MA','trimfquadrado',[125 175 225 m(j)]);
   b(j)=addmf(b(j),'input',1,'A','trimfquadrado',[175 225 275 m(j)]);
   b(j)=addmf(b(j),'input',1,'ALT','trapmfquad',[225 275 325 375 m(j)]);

Name='x1'
Range=[0 300]
NumMFs=6
MF1='B':'trapmfquad',[-75 -25 25 75 m]
MF2='MB':'trimfquadrado',[25 75 125 m]
MF3='M':'trimfquadrado',[75 125 175 m]
MF4='MA':'trimfquadrado',[125 175 225 m]
MF5='A':'trimfquadrado',[175 225 275 m]
MF6='ALT':'trapmfquad',[225 275 325 375 m]

%************************************************
% Funções de Pertinência de Saída
%************************************************
b(j)=addvar(b(j),'output','C1',[-1 2]);
   b(j)=addmf(b(j),'output',1,'BN','trapmfquad',[-1 -0.25 0 0 m(j)]);
   b(j)=addmf(b(j),'output',1,'BP','trapmfquad',[0 0 0.25 1 m(j)]);
   b(j)=addmf(b(j),'output',1,'MP','trimfquadrado',[0.5 1 1.5 m(j)]);
   b(j)=addmf(b(j),'output',1,'AP','trapmfquad',[1 1.5 2 3 m(j)]);

Name='C1'
Range=[-1 2]
NumMFs=4
MF1='BN':'trapmfquad',[-1 -0.25 0 0 m]
MF2='BP':'trapmfquad',[0 0 0.25 1 m]
MF3='MP':'trimfquadrado',[0.5 1 1.5 m]
MF4='AP':'trapmfquad',[1 1.5 2 3 m]
```

```
%****************************************
% Regras
%****************************************
rulelist1=[
1 2 1  1
2 3 1  1
3 4 1  1
4 3 1  1
5 2 1  1
6 1 1  1];

b(j)=addrule(b(j),rulelist1);

showrule(b(j));


%************************************************
%          SBRF
%************************************************
k=1;
for i=1:n
   var(i)=evalfis([p(i)],b(j));
   var_noncont(i)=evalfis([p_noncont(i)],b(j));
   g=rem(i-1,aplic);
   if g==0;
      cont(k)=evalfis([p(i),var(i)],a(j));
      p(i+1)=(p(i)+var(i))*(1-cont(k));
      k=k+1;
   else
      p(i+1)=p(i)+ var(i);
   end
   p_noncont(i+1)=p_noncont(i)+var_noncont(i);
end

t=0:n;
limite=40;
figure
plot(t,p,'b',t,p_noncont,':r',1,limite,'g')
title(strcat('potencia = ', num2str(m(j))));
xlabel('Iteração','fontsize',13);ylabel('Densidade Populacional','fontsize',13);
RangeY=[0 300];
RangeY=b(j).input(1).range
axis([0 n RangeY]);
H(j,:)=cont;
[p cont1]=ControleCaso1(20,15,400);
 contpot1=cont1;
 kk=0:1:(k-2);
limite=40;
figure
plot(kk,contpot1,':r',kk,cont,'k')
legend('eficiencia do biocida com sistema p-fuzzy','eficiencia do biocida com sistema p-fuzzy modificado')
hold on
title(strcat('potencia = ', num2str(m(j))));
RangeContY=a(j).output(1).range;;
RangeContX=[1 (length(cont)+2)];
axis([RangeContX RangeContY]);
xlabel('Iteração (j)');
ylabel('Controle (C)');
```

```
 Dif(j,:)=H(j,:)-contpot1;

%************************************************
% Critérios para escolha do Erro
%************************************************
 N1(j)=max(abs(Dif(j,:)));
 N2(j)=(max(abs(Dif(j,:)))./max(abs(contpot1));

cont2=H(sm1,:)

 figure;
 plot(kk,Dif(j,:))

end

disp('erro1')
N1

disp('erro2')
 N2

k1=max(N1)
k2=max(N2)

kk1=find(N1==k1);
kk2=find(N2==k2);

m(kk1)
m(kk2)
```

Rotina Computacional para o Sistema p-Fuzzy Modificado no Tempo

```
clear all
close all

%*************************************************************************
% CRESCIMENTO POPULACIONAL INIBIDO P-FUZZY
% CASO UNIDIMENSIONAL
%*************************************************************************
% function [p cont] = ControleCaso1(x0,apl,K)

p(1)=20; %População Inicial
   aplic=15; % Intervalo de aplicação (DIAS) = ');
   n=400; % iterações

p_noncont(1)=p(1);
```

```
%*********************************************************************
% Função que altera as potências e modifica o sistema no tempo
%*********************************************************************

for i=1:n

%m(i)= 1/i;
%m(i)= 0.5 + 1/i;
m(i)= 0.9 + 1/i;

%*********************************************************************
% SBRF
%*********************************************************************
a(i)=newfis('praga1');
 getfis(a(i))

Name='praga1'
Type='mamdani'
Version=2.0
NumInputs=2
NumOutputs=1
NumRules=24
AndMethod='min'
OrMethod='max'
ImpMethod='min'
AggMethod='max'
DefuzzMethod='centroid'

%*********************************************************************
% Funções de Pertinência de Entrada
%*********************************************************************
a(i)=addvar(a(i),'input','x',[0 300]);
   a(i)=addmf(a(i),'input',1,'B','trapmfquad',[-75 -25 25 75 m(i)]);
   a(i)=addmf(a(i),'input',1,'MB','trimfquadrado',[25 75 125 m(i)]);
   a(i)=addmf(a(i),'input',1,'M','trimfquadrado',[75 125 175 m(i)]);
   a(i)=addmf(a(i),'input',1,'MA','trimfquadrado',[125 175 225 m(i)]);
   a(i)=addmf(a(i),'input',1,'A','trimfquadrado',[175 225 275 m(i)]);
   a(i)=addmf(a(i),'input',1,'ALT','trapmfquad',[225 275 325 375 m(i)]);

Name='x'
Range=[0 300]
NumMFs=6
MF1='B':'trapmfquad',[-75 -25 25 75 m]
MF2='MB':'trimfquadrado',[25 75 125 m]
MF3='M':'trimfquadrado',[75 125 175 m]
MF4='MA':'trimfquadrado',[125 175 225 m]
MF5='A':'trimfquadrado',[175 225 275 m]
MF6='ALT':'trapmfquad',[225 275 325 375 m]

a(i)=addvar(a(i),'input','deltax',[-1 2]);
   a(i)=addmf(a(i),'input',2,'BN','trapmfquad',[-1 -0.25 0 0 m(i)]);
   a(i)=addmf(a(i),'input',2,'BP','trapmfquad',[0 0 0.25 1 m(i)]);
   a(i)=addmf(a(i),'input',2,'MP','trimfquadrado',[0.5 1 1.5 m(i)]);
   a(i)=addmf(a(i),'input',2,'AP','trapmfquad',[1 1.5 2 3 m(i)]);
```

```
Name='deltax'
Range=[-1 2]
NumMFs=4
MF1='BN':'trapmfquad',[-1 -0.25 0 0 m]
MF2='BP':'trapmfquad',[0 0 0.25 1 m]
MF3='MP':'trimfquadrado',[0.5 1 1.5 m]
MF4='AP':'trapmfquad',[1 1.5 2 3 m]

%****************************************************************
% Funções de Pertinência de Saída
%****************************************************************
a(i)=addvar(a(i),'output','C',[0 0.9]);
   a(i)=addmf(a(i),'output',1,'C0','trapmfquad',[-0.2 -0.1 0 0 m(i)]);
   a(i)=addmf(a(i),'output',1,'CB','trapmfquad',[0 0.21 0.25 0.43 m(i)]);
   a(i)=addmf(a(i),'output',1,'CM','trapmfquad',[0.245 0.43 0.48 0.625 m(i)]);
   a(i)=addmf(a(i),'output',1,'CA','trapmfquad',[0.45 0.65 1 1.4 m(i)]);

Name='C'
Range=[0 0.9]
NumMFs=4
MF1='C0':'trapmfquad',[-0.2 -0.1 0 0 m]
MF2='CB':'trapmfquad',[0 0.21 0.25 0.43 m]
MF3='CM':'trapmfquad',[0.245 0.43 0.48 0.625 m]
MF4='CA':'trapmfquad',[0.45 0.65 1 1.4 m]

%****************************************************************
% Regras
%****************************************************************
rulelist=[
1 1 1 1  1
2 1 1 1  1
3 1 1 1  1
4 1 2 1  1
5 1 3 1  1
6 1 4 1  1
1 2 1 1  1
2 2 1 1  1
3 2 2 1  1
4 2 3 1  1
5 2 3 1  1
6 2 4 1  1
1 3 1 1  1
2 3 2 1  1
3 3 3 1  1
4 3 3 1  1
5 3 4 1  1
6 3 4 1  1
1 4 2 1  1
2 4 3 1  1
3 4 3 1  1
4 4 4 1  1
5 4 4 1  1
6 4 4 1  1];

a(i)=addrule(a(i),rulelist);

showrule(a(i));
```

```
%*******************************************************
% SBRF
%*******************************************************
b(i)=newfis('praga');
 getfis(b(i))

Name='praga'
Type='mamdani'
Version=2.0
NumInputs=1
NumOutputs=1
NumRules=6
AndMethod='min'
OrMethod='max'
ImpMethod='min'
AggMethod='max'
DefuzzMethod='centroid'

%*******************************************************
% Funções de Pertinência de Entrada
%*******************************************************
b(i)=addvar(b(i),'input','x1',[0 300]);
   b(i)=addmf(b(i),'input',1,'B','trapmfquad',[-75 -25 25 75 m(i)]);
   b(i)=addmf(b(i),'input',1,'MB','trimfquadrado',[25 75 125 m(i)]);
   b(i)=addmf(b(i),'input',1,'M','trimfquadrado',[75 125 175 m(i)]);
   b(i)=addmf(b(i),'input',1,'MA','trimfquadrado',[125 175 225 m(i)]);
   b(i)=addmf(b(i),'input',1,'A','trimfquadrado',[175 225 275 m(i)]);
   b(i)=addmf(b(i),'input',1,'ALT','trapmfquad',[225 275 325 375 m(i)]);

Name='x1'
Range=[0 300]
NumMFs=6
MF1='B':'trapmfquad',[-75 -25 25 75 m]
MF2='MB':'trimfquadrado',[25 75 125 m]
MF3='M':'trimfquadrado',[75 125 175 m]
MF4='MA':'trimfquadrado',[125 175 225 m]
MF5='A':'trimfquadrado',[175 225 275 m]
MF6='ALT':'trapmfquad',[225 275 325 375 m]

%*******************************************************
% Funções de Pertinência de Saída
%*******************************************************
b(i)=addvar(b(i),'output','C1',[-1 2]);
   b(i)=addmf(b(i),'output',1,'BN','trapmfquad',[-1 -0.25 0 0 m(i)]);
   b(i)=addmf(b(i),'output',1,'BP','trapmfquad',[0 0 0.25 1 m(i)]);
   b(i)=addmf(b(i),'output',1,'MP','trimfquadrado',[0.5 1 1.5 m(i)]);
   b(i)=addmf(b(i),'output',1,'AP','trapmfquad',[1 1.5 2 3 m(i)]);

Name='C1'
Range=[-1 2]
NumMFs=4
MF1='BN':'trapmfquad',[-1 -0.25 0 0 m]
MF2='BP':'trapmfquad',[0 0 0.25 1 m]
MF3='MP':'trimfquadrado',[0.5 1 1.5 m];
MF4='AP':'trapmfquad',[1 1.5 2 3 m];
```

```
%*****************************************************
% Regras
%*****************************************************
rulelist1=[
1 2 1  1
2 3 1  1
3 4 1  1
4 3 1  1
5 2 1  1
6 1 1  1];

b(i)=addrule(b(i),rulelist1);

showrule(b(i));

%************************************************************
%          Controle Fuzzy
%************************************************************
k=1;
for j=1:n
   var(j)=evalfis([p(j)],b(i));
   var_noncont(j)=evalfis([p_noncont(j)],b(i));
   g=rem(j-1,aplic);
   if g==0;
      cont(k)=evalfis([p(j),var(j)],a(i));
      p(j+1)=(p(j)+var(j))*(1-cont(k));
      k=k+1;
   else
      p(j+1)=p(j)+ var(j);
   end
   p_noncont(j+1)=p_noncont(j)+var_noncont(j);
end

end

t=0:n;
figure
plot(t,p,'b',t,p_noncont,':r')
%title(strcat('potencia = ', num2str(m(i))));
xlabel('Iteração','fontsize',13);ylabel('Densidade Populacional','fontsize',13);
legend('com controle','sem controle')
%RangeY=[0 300];
RangeY=b(i).input(1).range
axis([0 n RangeY]);
 [cont2]=ControleCaso1modificado2(20,15,400);

kk=0:1:(k-2);

 contpot1=[0 0 0 0.19270505383284 0.19268199287205 0.19266498901395 0.19265246254318
0.19264324036032 0.19263645402807 0.19263146189044 0.19262779052510 0.19262509099951
0.19262310633292 0.19262164737172 0.19262057494499 0.19261978668810 0.19261920732549
0.19261878151110 0.19261846855692 0.19261823855352 0.19261806951617 0.19261794528592
0.19261785398625 0.19261778688831 0.19261773757689 0.19261770133717 0.19261767470409];
```

```
figure
plot(kk,contpot1,':r',kk,cont,'k',kk,cont2,':g')
legend('p-fuzzy','p-fuzzy modificado','p-fuzzy modificado no tempo')
% hold on
% title(strcat('potencia = ', num2str(m(i))));
RangeContY=a(i).output(1).range;
RangeContX=[1 (length(cont)+2)];
axis([RangeContX RangeContY]);
xlabel('Iteração (j)');
ylabel('Controle (C)');

Dif2=cont2-cont

figure;
 plot(kk,Dif2)
 legend('Vetor diferença entre Modificado e Modificado no Tempo')


%*****************************************************************
% Critérios para escolha do Erro
%*****************************************************************
N1=max(abs(Dif2));

N2=(max(abs(Dif2)))./max(abs(cont));
```