MINISTÉRIO DA CIÊNCIA, TECNOLOGIA E INOVAÇÃO
INSTITUTO NACIONAL DE PESQUISAS ESPACIAIS

sid.inpe.br/mtc-m21b/2014/06.12.01.41-TDI

PREVISIBILIDADE EM SISTEMAS CAÓTICOS UTILIZANDO SISTEMAS NEURO-DIFUSOS

Pettras Leonardo Bueno dos Santos

Dissertação de Mestrado do Curso de Pós-Graduação em Computação Aplicada, orientada pelos Drs. Sandra Aparecida Sandri, e Haroldo Fraga de Campos Velho, aprovada em 15 de abril de 2014.

URL do documento original:
<http://urlib.net/8JMKD3MGP5W34M/3GF6ESE>

INPE
São José dos Campos
2014

PUBLICADO POR:

Instituto Nacional de Pesquisas Espaciais - INPE
Gabinete do Diretor (GB)
Serviço de Informação e Documentação (SID)
Caixa Postal 515 - CEP 12.245-970
São José dos Campos - SP - Brasil
Tel.:(012) 3208-6923/6921
Fax: (012) 3208-6919
E-mail: pubtc@sid.inpe.br

MINISTÉRIO DA CIÊNCIA, TECNOLOGIA E INOVAÇÃO
INSTITUTO NACIONAL DE PESQUISAS ESPACIAIS

sid.inpe.br/mtc-m21b/2014/06.12.01.41-TDI

PREVISIBILIDADE EM SISTEMAS CAÓTICOS UTILIZANDO SISTEMAS NEURO-DIFUSOS

Pettras Leonardo Bueno dos Santos

Dissertação de Mestrado do Curso de Pós-Graduação em Computação Aplicada, orientada pelos Drs. Sandra Aparecida Sandri, e Haroldo Fraga de Campos Velho, aprovada em 15 de abril de 2014.

URL do documento original:
<http://urlib.net/8JMKD3MGP5W34M/3GF6ESE>

INPE
São José dos Campos
2014

Dados Internacionais de Catalogação na Publicação (CIP)

Santos, Pettras Leonardo Bueno dos.

Sa59p Previsibilidade em sistemas caóticos utilizando sistemas neuro-difusos / Pettras Leonardo Bueno dos Santos. – São José dos Campos : INPE, 2014.

xxiv + 90 p. ; (sid.inpe.br/mtc-m21b/2014/06.12.01.41-TDI)

Dissertação (Mestrado em Computação Aplicada) – Instituto Nacional de Pesquisas Espaciais, São José dos Campos, 2014.

Orientadores : Drs. Sandra Aparecida Sandri, e Haroldo Fraga de Campos Velho.

1. Lógica Fuzzy. 2. Sistemas caóticos. 3. Neuro-fuzzy. 4. Bred vectors. I.Título.

CDU 681.513.6

Aprovado (a) pela Banca Examinadora em cumprimento ao requisito exigido para obtenção do Título de ***Mestre*** em

Computação Aplicada

Dr. Elbert Einstein Nehrer Macau

Presidente / INPE / São José dos Campos - SP

Dra. Sandra Aparecida Sandri

Orientador(a) / INPE / SJCampos - SP

Dr. Haroldo Fraga de Campos Velho

Orientador(a) / INPE / São José dos Campos - SP

Dr. José Antonio Aravéquia

Membro da Banca / INPE / São José dos Campos - SP

Dr. Fernando Antonio Campos Gomide

Convidado(a) / UNICAMP / Campinas - SP

Este trabalho foi aprovado por:

( ) maioria simples

( ) unanimidade

Aluno (a): ***Pettras Leonardo Bueno dos Santos***

São José dos Campos, 15 de Abril de 2014

"Quando a mente se abre ao impossível, às vezes se encontra a verdade".

Walter Bishop

Á minha mãe Olinda, à minha irmã Ana Paula e à minha esposa Diana

AGRADECIMENTOS

À Capes, pelo incentivo financeiro durante 1 ano do desenvolvimento deste trabalho.

Aos meus orientadores Sandra Aparecida Sandri e Haroldo Fraga de Campos Velho, que me receberam de braços abertos e sempre acreditaram no meu trabalho, apoiando e conduzindo esta pesquisa.

A todos os professores responsáveis pela minha formação acadêmica.

Aos meus amigos, Érica Gouvea, Érica Ferreira, Luis, Juliana Anochi, Marlon, Sabrina, Saymon, Rudinei, André, Daniel, Diego, Juliano, Ligia, Marcos, Marina, Peterson e Toni, pessoas que convivi durante esses anos de estudos e que jamais esquecerei; em especial aos meus amigos Jonas Mendonça e Eduardo Luz que me ajudaram a produzir a versão em LATEX deste texto.

Aos meus familiares, em especial à minha mãe Olinda Bueno dos Santos que sempre acreditou no meu potencial e sempre foi uma grande incentivadora.

À minha esposa Diana, que com amor, incentivo, companheirismo, dedicação e paciência completa a minha vida, e tem sido o meu pilar de sustentação. Além disso, sem a sua perseverança e incentivo, sei que não teria conseguido terminar este trabalho.

Enfim, agradeço a todas as pessoas que de alguma forma me ajudaram a dar mais este passo na construção da minha história.

RESUMO

Os sistemas caóticos apresentam sensibilidade às condições iniciais. Pequenas alterações nestas condições iniciais podem levar a resultados muito diferentes da trajetória original do sistema. Esta característica faz com que seja muito difícil prever o comportamento destes sistemas, principalmente porque em várias aplicações práticas, as condições iniciais são obtidas com instrumentos de medida, os quais estão sujeitos a erros de precisão. A previsibilidade do comportamento de sistemas caóticos é uma área de grande importância porque muitos fenômenos do mundo real apresentam algum tipo de comportamento caótico. O objetivo deste trabalho é realizar a previsibilidade em sistemas caóticos e também gerar automaticamente um conjunto de regras interpretáveis sobre o sistema. O problema de previsibilidade pode ser formulado como um problema de classificação. Deste modo, podemos utilizar alguns dentre vários tipos de classificadores que estão disponíveis atualmente. A técnica de *"bred vectors"* é utilizada para avaliar o sistema não linear e os sistemas neuro-difusos gerados pelas ferramentas ANFIS e GUAJE são empregados para classificar a dinâmica. A partir do treinamento com pares entrada/saída (magnitude do bred vector / classe de dinâmica), o ANFIS gera um sistema difuso do tipo *Takagi-Sugeno*. Tal sistema pode ser utilizado com um conjunto de testes para verificar a eficácia do classificador neuro-difuso. Os resultados da classificação com o sistema ANFIS são comparados com a classificação utilizando Redes Neurais Artificiais (RNAs). Tanto o ANFIS quanto as RNAs apresentam bons resultados, porém geram sistemas de difícil interpretabilidade. Para gerar sistemas com boa interpretabilidade e manter uma acurácia alta, utilizamos o *software* GUAJE, o qual permite gerar um conjunto de regras para um sistema difuso do tipo *Mamdani*. Como objeto de estudo, utilizamos dois sistemas que apresentam comportamento caótico na forma de um atrator estranho, o Sistema de Lorenz (associado à dinâmica da atmosfera) e o modelo não linear de três ondas acopladas (física solar).

PREDICTABILITY IN CHAOTIC SYSTEMS USING NEURO-FUZZY SYSTEMS

ABSTRACT

Chaotic systems have much sensitivity to initial conditions. Small changes in these initial conditions can lead to very different results from the original trajectory of the system. This feature makes it very difficult to predict the behavior of systems, mainly because in many practical applications, the initial conditions are obtained with measurement instruments, which are subject to precision errors. The predictability of the behavior of chaotic systems is an area of great importance because many real world phenomena have some kind of chaotic behavior. The aim of this work is predictability in chaotic systems and also to automatically generate a set of interpretable rules on the system. The first step is to model the predictability problem in a classification problem. From there, we can use some of several classifiers available currently. In this study, we use the technique of "bred vectors" to generate pairs of input/output required for the neuro-fuzzy system ANFIS. From these pairs of training the ANFIS generates a system of Fuzzy Takagi-Sugeno type, this system can be used with a set of tests to check the effectiveness of the model, in order to compare the results we also use Artificial Neural Networks (ANN). Both ANFIS as the ANNs present good results, but generate systems with difficult interpretability. To generate systems with good interpretability and maintaining a high accuracy, we use the software GUAJE, which allows generating a set of rules for a type Mamdani Fuzzy system. As an object of study, we used two systems that present chaotic behavior in the form of a strange attractor, the Lorenz System and nonlinear three-wave coupled model.

LISTA DE FIGURAS

Pág.

LISTA DE TABELAS

Pág.

LISTA DE ABREVIATURAS E SIGLAS

ANFIS	–	Adaptive-Network-Based Fuzzy Inference System
CPTEC	–	Centro de Previsão de Tempo e Estudos Climáticos
LLE	–	Local Lyapunov Exponents
LMS	–	Least Mean Square
MLP	–	Multilayer Perceptron
Neurônio MCP	–	Neurônio de McCulloch e Pitts
PO	–	Perturbed Observation
RNA	–	Redes Neurais Artificiais

LISTA DE SÍMBOLOS

σ	–	Parâmetro das Equações do Sistema de Lorenz
ρ	–	Parâmetro das Equações do Sistema de Lorenz
β	–	Parâmetro das Equações do Sistema de Lorenz
α	–	Constante nas funções de ativação de RNAs
$\in$	–	Pertence ao conjunto
$\notin$	–	Não pertence ao conjunto
Ω	–	Conjunto universo
μ_A	–	Função de pertinência
$\varphi(a, b)$	–	Função com parâmetros a e b
$\top$	–	Operador de intersecção
$\perp$	–	Operador de união
$\tilde{}$	–	Operador negação
μ_A (x)	–	Função de ativação na camada 1 do sistema ANFIS
w_i	–	Função de ativação na camada 2 do sistema ANFIS
$(\overline{w_i})$	–	Função de ativação na camada 3 do sistema ANFIS
O_i	–	Função de ativação na camada 4 do sistema ANFIS
S	–	Função de ativação na camada 5 do sistema ANFIS
Σ	–	Somatório

SUMÁRIO

Pág.

1 INTRODUÇÃO

Os sistemas caóticos apresentam, em geral, sensibilidade às condições iniciais. Pequenas alterações nas condições iniciais podem levar a resultados muito diferentes. Esta característica faz com que seja muito difícil prever o comportamento destes sistemas, a partir de dados obtidos com instrumentos de medidas principalmente no longo prazo, isto se deve à precisão limitada associada aos sensores que permitem a realização das medidas. Muitas aplicações práticas dependem da aquisição de dados através de instrumentos de medidas, os quais estão sujeitos a erros de precisão na leitura dos dados. Estes erros, por menores que sejam, alteram significativamente nossa capacidade de estimativa de estados futuros associados à uma trajetória caótica a partir de suas medias.

A previsibilidade do comportamento de sistemas caóticos é uma área de grande importância, porque muitos fenômenos do mundo real apresentam algum tipo de comportamento caótico. Por exemplo, em Economia (variações irregulares em preços de ações), Meteorologia (mudanças repentinas no tempo), Biologia (complexidade em sistemas fisiológicos e bioquímicos), Física (séries temporais de fenômenos não lineares), entre outros. Muitos trabalhos tem sido desenvolvidos com o objetivo de conseguir algum tipo de previsibilidade em sistemas caóticos. Por exemplo, em (TOTH; KALNAY, 1997), (NEWMAN et al., 2003), (EVANS, 2004) e (CINTRA; VELHO, 2008), os autores utilizam a técnica conhecida como *"bred vectors"* para prever o comportamento de sistemas caóticos. Já em (GUÉGAN; LEROUX, 2011) e (ECKHARDTA B.; YAOA, 1993) são utilizados o Expoentes Locais de Lyapunov - LLE (do inglês, *Local Lyapunov Exponents*). Em (MENDONCA; BONATTI, 2002), os autores explicam o sistema de previsão de tempo global por *"ensemble"* do Centro de Previsão de Tempo e Estudos Climáticos (CPTEC) - INPE. (PALMER, 1998) utilizou a técnica *"singular vectors"*. Em (HAMILL; SNYDER, 2000), os autores fazem um estudo comparativo entre *"bred vectors"*, *"singular vectors"* e PO (*"Perturbed Observation"*). Nos trabalhos de (CINTRA; VELHO, 2008) e (EVANS, 2004), os autores chegaram a um conjunto de duas regras sobre o comportamento do Modelo de Lorenz (Atrator de Lorenz).

Neste trabalho, pretendemos gerar um conjunto de regras para prever o comportamento do sistema, isto é, previsão de localização de trajetória no atrator estranho, tanto para o sistema de Lorenz quanto para o modelo não linear de três ondas acopladas. A princípio, trabalhamos com o Sistema Neuro-Difuso ANFIS (*Adaptive Neuro-based difuso Inference System*), proposto por (JANG, 1993), porém utilizamos

também Redes Neurais Artificiais (RNAs) supervisionadas para fins de comparação e também corroborar os resultados obtidos com o ANFIS. Redes neurais já foram usadas com *"bred vectors"* para predizer o comportamento do sistema de Lorenz em regime caótico (PASINI; PELINO, 2005). É dito que o conhecimento nas RNAs está nos pesos que são ajustados durante o treinamento, porém esse conhecimento é difícil de interpretar, fazendo com que a rede neural funcione como uma "caixa preta": os dados são apresentados e a rede os classifica sem que o usuário saiba exatamente como esses dados foram classificados. No caso do ANFIS, são geradas regras, porém essas não são facilmente interpretáveis, o ANFIS gera um modelo difuso do tipo *Takagi-Sugeno*, o qual também funciona como um classificador, mas com regras de difícil interpretação pelo usuário.

Para gerar um classificador com regras mais facilmente interpretáveis, utilizamos o Software GUAJE. O GUAJE permite criar um sistema neuro-difuso do tipo *Mamdani*, que trabalha com regras interpretáveis. Essas regras são geradas automaticamente com o GUAJE através de seus algoritmos.

Esta dissertação está organizada da seguinte maneira: no Capítulo 1 foram realizadas a apresentação e a introdução do trabalho; no Capítulo 2 são tratados os Sistemas Dinâmicos, apresentando algumas definições, um breve histórico, até chegar aos atratores estranhos do sistema de Lorenz e do modelo não linear de três ondas acopladas; Redes Neurais Artificiais (RNAs), neurônio MCP (McCulloch e Pitts), topologias das RNAs e seus algoritmos de aprendizagem são abordados no Capítulo 3; no Capítulo 4 são apresentados os conceitos da teoria dos conjuntos difusos e algumas de suas operações e dos Sistemas de Inferência *difuso*; finalmente, no Capítulo 5, já com os conceitos de RNAs e Sistemas Difusos discutidos, apresentaremos os conceitos dos sistemas híbridos Neuro-Difusos e o sistemas ANFIS e GUAJE; no Capítulo 6, é discutida a técnica de *"bred vectors"* como ferramenta para a previsibilidade em sistemas caóticos e a metodologia deste trabalho; os resultados obtidos são apresentados e discutidos no Capítulo 7 e, finalmente, no capítulo 8 apresentamos as conclusões e perspectivas futuras.

2 SISTEMAS DINÂMICOS E ATRATORES CAÓTICOS

2.1 Definição

Um Sistema pode ser definido como um conjunto de objetos agrupados por alguma interação ou interdependência, de modo que existam relações de causa e efeito nos fenômenos que ocorrem com os elementos deste conjunto. Um Sistema é dito dinâmico quando algumas grandezas que caracterizam seus objetos variam no tempo (MONTEIRO, 2006), assim podemos dizer que um Sistema Dinâmico apresenta propriedades que variam no tempo. Um Sistema Dinâmico pode ser descrito por um conjunto de equações diferenciais, com o objetivo de caracterizar qualitativamente e quantitativamente suas soluções. Tais equações podem ser obtidas utilizando as leis físicas que governam um determinado sistema, por exemplo, as leis de Newton para sistemas mecânicos ou as leis de Kirchhoff para sistemas elétricos.

2.2 O problema de $N - corpos$

Utilizando as leis de Newton, é possível deduzir as equações que descrevem o movimento, por exemplo, dos planetas do sistema solar (MONTEIRO, 2006). Dados N pontos de massa não nula em um espaço tridimensional, sujeitos apenas à atração gravitacional mútua, o ‘"problema de $N - corpos$"consiste em predizer as posições destes corpos através das soluções das equações que descrevem o movimento deste sistema. Em seu livro *Essai philosophique sur les probabilités* (publicado em 1812), Pierre Simon Laplace $(1749 - 1827)$ afirma que: "Se alguma ’inteligência’ pudesse conhecer a posição $\vec{x}$ e a velocidade $\vec{v}$ de cada partícula do universo em um dado instante t, assim como a massa e a força que age sobre cada uma das partículas, então esta ’inteligência’ poderia prever o futuro do universo para sempre". Apesar de a frase parecer muito otimista, esse era um pensamento muito corrente na época. A frase de Laplace remete ao problema de $N-corpos$. Newton resolveu analiticamente este problema para $N = 2$. Porém, para $N = 3$ ainda não foi possível encontrar uma solução analítica para o problema.

2.3 Atratores

Atrator pode ser compreendido como uma região limitada no espaço onde todas as trajetórias de um sistema dinâmico estão confinadas. Na teoria de caos, os pesquisadores lidam com conceitos de atrator estranho (citar paper do Ruelle). Entretanto, alguns autores mostraram que existem sistemas com dinâmica confinada em um atrator estranho, mas não são caóticos (GREBOGI, 1984).

Um atrator caótico é definido por (GREBOGI, 1984) (página 262) da seguinte maneira:

> *Definition. A chaotic attractor is one for which typical orbits on the attractor have a positive Lyapunov exponent.*
> *In the above definition we have used the idea of "typical"orbits on the attractor. That is, we assume that, for almost any initial condition in the basin of attraction of the attractor, the largest Lyapunov exponents generated by those (typical) initial conditions exist and are identical. Further we use the following definitions of an attractor and a basin of attraction:*

A idéia de atrator caótico é feita para diferenciar do conceito de atrator estranho, assim definido por (GREBOGI, 1984) (página 263):

> *Definition: An attractor is a compact set with a neighborhood such that, for almost every initial condition in this neighborhood, the limit set of the orbit as time tends to $+\infty$ is the attractor.*
> *We define a strange attractor as follows:*
> *Definition. A strange attractor is an attractor which is not a finite set of points and is not piecewise differentiable. We say that it is piecewise differentiable if it is either a piecewise differentiable curve or surface, or a volume bounded by a piecewise differentiable closed surface.*
> *Definition. An attractor is a compact set with a neighborhood such that, for almost every initial condition in this neighborhood, the limit set of the orbit as time tends to $+\infty$ is the attractor.*
> *Definition. The basin of attraction of an attractor is the closure of the set of initial conditions which approach the attractor as time tends to $+\infty$.*

A motivação para estas definições esta no fato de se caracterizar que nem todo atrator estranho é um atrator caótico. Neste trabalho, o sistema de Lorenz e o sistema de 3 ondas acopladas em física solar são atratores estranhos e possuem regimes caóticos.

2.4 Sistema de Lorenz

Edward Lorenz (1917 – 2008) desenvolveu em seus trabalhos, (LORENZ, 1963), (LORENZ, 1965b) e (LORENZ, 1965a), um modelo matemático simplificado para descrever movimentos atmosféricos. A partir destes trabalhos percebeu que: pequenas variações nos valores iniciais das variáveis de seu modelo resultavam em valores muito divergentes. Em sistemas dinâmicos, estes resultados "instáveis" dizem respeito à evolução temporal como função de seus parâmetros e variáveis. Lorenz em sua pesquisa de sistemas dinâmicos usou três equações diferenciais ordinárias para representar graficamente o comportamento do sistema através de computadores, descreveu um sistema relativamente simples com um padrão de comportamento não trivial e verificou que a partir de estados iniciais ligeiramente diferentes, o sistema de equações diferenciais resultava em soluções completamente diferentes entre si, após algum tempo de evolução temporal. O sistema de Lorenz consiste de três equações diferenciais ordinárias de primeira ordem, acopladas (LORENZ, 1963) e (CINTRA et al., 2010):

$$\frac{dx}{dt} = -\sigma x - y \tag{2.1}$$

$$\frac{dy}{dt} = -\rho x - y - xz \tag{2.2}$$

$$\frac{dz}{dt} = xy - \beta z \tag{2.3}$$

Nas equações acima, utilizando os parâmetros $\sigma = 10$, $\rho = 28$ e $\beta = 8/3$, suas soluções numéricas levam a um atrator imerso em um espaço tridimensional com coordenadas (x, y, z), (LORENZ, 1963). Neste trabalho, sempre que nos referirmos ao Sistema de Lorenz, estamos considerando estes valores de parâmetros. A Figura 2.1 ilustra o Atrator Lorenz, cuja dinâmica é sabidamente caótica para determinados conjuntos de parâmetros (OTT, 2002).

2.5 Modelo de Três Ondas Acopladas Não Linear

O Modelo Três Ondas Acopladas Não Linear é de interesse geral em muitos ramos da física, como fusão nuclear, astrofísica, geofísica espacial, óptica não-linear e mecânica dos fluidos. No regime saturado deste modelo, as soluções podem ser caóticas

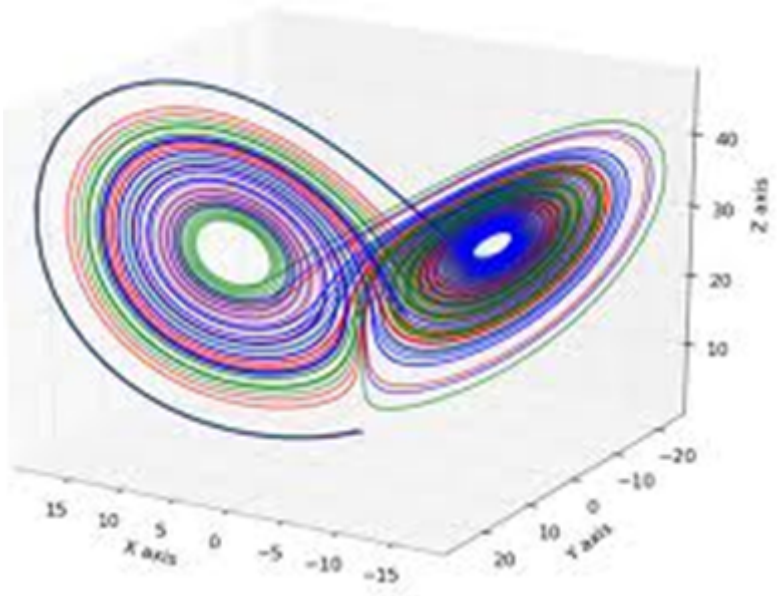

Figura 2.1 - Sistema Atrator de Lorenz

encontrado, neste caso, soluções de onda que podem evoluir na forma de um comportamento que se considera como caótico através de várias vias tais como período de duplicação ou intermitente.

A dedução teórica deste sistema pode ser feita seguindo os mesmos passos usados para deduzir o sistema de Lorenz - Equações 2.1-2.3. O modelo mais simples para descrever a dinâmica temporal ressonante de acoplamento não linear de três ondas pode ser obtido considerando-se somente as equações das amplitudes de onda. Além disso, as ondas podem ser assumidas monocromáticas, com os campos eléctricos na forma:

$$E_\alpha(x,t) = \frac{1}{2}A_\alpha(x,t)\exp\{i(k_\alpha x) - \omega_\alpha\} \tag{2.4}$$

Onde $\alpha = 1, 2, 3$ e a escala de tempo das interações não lineares é muito maior do que o período das ondas lineares (desacoplada). Para que as interações de três ondas ocorreram, as frequências de onda ω_α e os vetores de onda k_α devem satisfazer as condições de ressonância:

$$\omega_3 \cong \omega_1 - \omega_2, \quad k_3 = k_1 - k_2 \tag{2.5}$$

Sob estas circunstâncias, a dinâmica temporal não linear do sistema pode ser regulada pelo seguinte conjunto de três equações diferenciais autônomas de primeira ordem.

$$\frac{dA_1}{d\tau} = v_1 A_1 - A_2 A_3 \tag{2.6}$$

$$\frac{dA_2}{d\tau} = i\delta A_2 + v_2 A_2 - A_1 A_3 \tag{2.7}$$

$$\frac{dA_3}{d\tau} = v_3 A_3 - A_2 A_1 \tag{2.8}$$

Onde a variável $\tau = k(x - vt)$, com v e k sendo velocidade da onda e o vetor da onda, respectivamente, e $\delta = \frac{((\omega_1 - \omega_2 - \omega_3))}{x}$ é a frequência linear normalizada e $v_\alpha = \frac{\omega_\alpha}{x}$ calcula o comportamento de onda linear em uma escala de tempo longa. A onda A_1 é linearmente instável ($v_1 > 0$) e as outras duas ondas, A_2 e A_3 são linearmente amortecidas $v_2 = v_3 \equiv -v < 0$ e em seguida é definido $x = v_1$.

A Figura 2.2 ilustra esse atrator caótico, um regime é uma linha reta e o outro regime é caracterizado por uma curva.

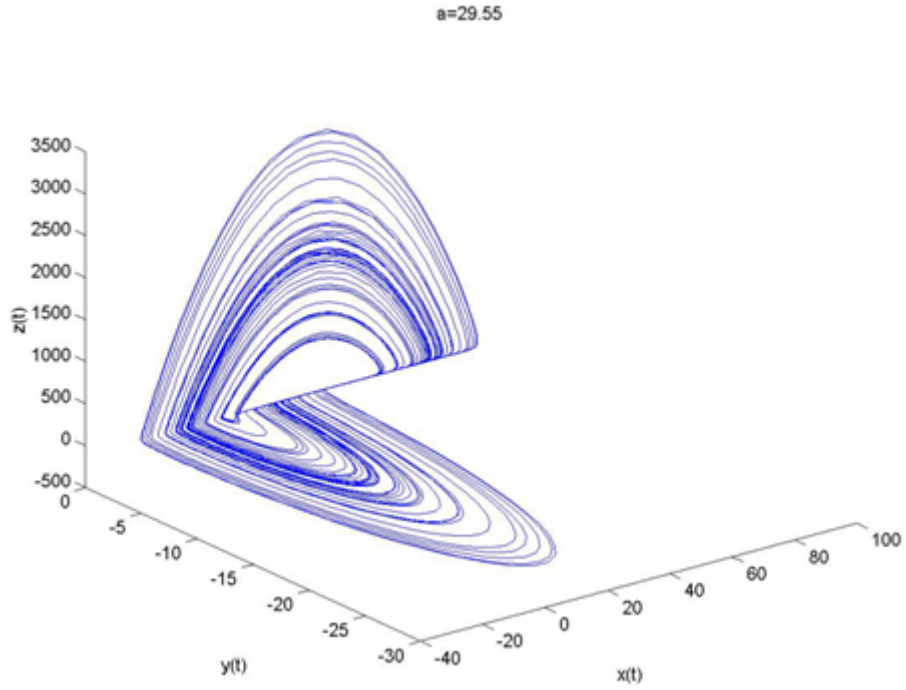

Figura 2.2 - Sistema Atrator Modelo Três Ondas

2.6 Sistemas Caóticos e Previsibilidade

Os Sistemas Caóticos, em geral, são sensíveis a variações de suas condições iniciais. Neste sentido, muitos fenômenos do mundo real das mais diversas áreas apresentam comportamento que se assume ser caótico (OTT, 2002). Em vista disto, é importante que se desenvolvam métodos que, de alguma forma, consigam trazer algum tipo de previsibilidade a estes sistemas.

Muitos trabalhos tem sido desenvolvidos com o objetivo de conseguir algum tipo de previsibilidade em sistemas caóticos. Por exemplo, em (TOTH; KALNAY, 1997), (NEWMAN et al., 2003), (EVANS, 2004) e (CINTRA; VELHO, 2008) os autores utilizam a técnica conhecida como *"bred vectors"* para prever o comportamento de sistemas caóticos. Já em (GUÉGAN; LEROUX, 2011) e (ECKHARDTA B.; YAOA, 1993) são utilizados o Expoentes Locais de Lyapunov - LLE (do inglês, *Local Lyapunov Exponents*). Em (MENDONCA; BONATTI, 2002) os autores explicam o sistema de previsão de tempo global por ensemble do Centro de Previsão de Tempo e Estudos Climáticos (CPTEC) - INPE. Em (PALMER, 1998) é utilizada a técnica *"singular vectors"*. Em (HAMILL; SNYDER, 2000) os autores fazem um estudo comparativo entre *"bred vectors"*, *"singular vectors"* e PO (*"Perturbed Observation"*).

Nos trabalhos desenvolvidos em (EVANS, 2004) e em (CINTRA; VELHO, 2008) os autores utilizaram uma técnica proposta por (TOTH; KALNAY, 1997) conhecida como *"bred vectors"* ou *"breeding method"* para realizar a previsão de comportamento em sistemas caóticos, principalmente no Atrator de Lorenz. Neste trabalho utilizamos os *"bred vectors"* para gerar as entradas que alimentam os sistemas que utilizamos, conforme será discutido no capítulo 6.

3 REDES NEURAIS ARTIFICIAIS

3.1 O neurônio biológico

Fisiologicamente o neurônio é dividido em três partes: corpo da célula (ou soma), dendritos e axônio Figura 3.1. Os dendritos têm a função de levar os estímulos (informação) recebidos de outros neurônios até o corpo celular, onde a informação é processada e levada até o axônio, que por sua vez a passa adiante através dos dendritos de outros neurônios. O ponto onde ocorre o contato do axônio de um neurônio com um dendrito de outro neurônio é chamado sinapse. As sinapses ocorrem em regiões eletroquimicamente ativas, compreendidas entre duas membranas celulares: a membrana pré-sináptica, que é por onde chega um estímulo de outra célula, e a membrana pós-sináptica que é a membrana do dendrito. Na região intersináptica o estímulo nervoso que chega é transferido à membrana dendrital através de neurotransmissores (KOVACS, 1996).

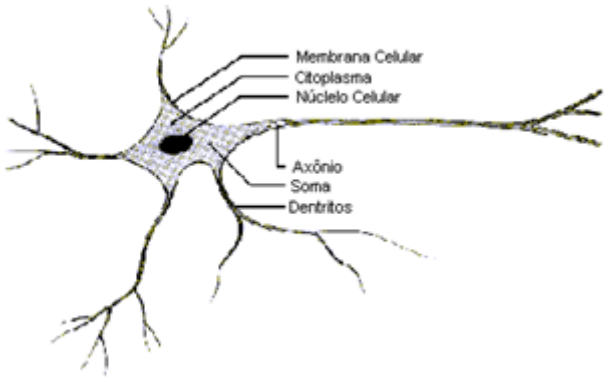

Figura 3.1 - Componentes de um neurônio biológico

O cérebro humano possui cerca de 10^{11} neurônios (nodos), onde cada um se comunica com milhares de outros continuamente e em paralelo, através das conexões sinápticas, formando uma rede neural (BRAGA et al., 2000).

Os sinais oriundos dos neurônios pré-sinápticos são passados para o corpo do neurônio, onde são comparados com outros sinais recebidos. Se a potência do sinal em um intervalo curto de tempo é suficientemente alto a célula "dispara" (um sinal excitatório que faz com que o neurônio transfira energia), produzindo um impulso eletro-químico que é transmitido para as células seguintes (neurônios pós-sinápticos). Este sistema relativamente "simples" é responsável pela maioria das funções realiza-

das pelo nosso cérebro, coordena várias atividades do organismo e tem a capacidade de resolver problemas complexos: são cerca de 10^{11} nodos do cérebro humano trabalhando conectados e em paralelo, vindo daí, a grande inspiração para a teoria de redes neurais artificiais (BRAGA et al., 2000).

3.2 Histórico das Redes Neurais Artificiais

As Redes Neurais Artificiais (RNAs) surgiram em meados da década de 40, época em que os primeiros computadores ocupavam uma sala inteira. Os pioneiros na teoria de RNAs foram o psiquiatra e neuro-anatomista Warren McCulloch e o estatístico Walter Pitts, que propuseram um modelo de um neurônio artificial, o qual será discutido na Seção 3.4, como uma unidade de processamento que recebe vários estímulos de entrada e gera um único sinal de saída.

Alguns anos após o trabalho de McCulloch e Pitts [1] , Donald Hebb mostrou como a plasticidade da aprendizagem de RNAs pode ser conseguida através do ajuste dos pesos de entrada dos nodos. Essa regra de Hebb foi interpretada matematicamente e hoje é muito utilizada em algoritmos de aprendizagem. Posteriormente, Widrow e Hoff criaram uma regra de aprendizado conhecida como regra Delta, também muito utilizada atualmente, a qual se baseia no método do gradiente para minimização do erro na saída de um neurônio com resposta linear (BRAGA et al., 2000).

Em 1958, Frank Rosenblatt propôs um modelo, conhecido como perceptron, composto por uma estrutura de rede tendo como unidades básicas nodos MCP (que serão discutidos na seção 3.4), contendo uma regra de aprendizado. Através deste modelo Rosenblatt demonstrou que se forem acrescidas sinapses ajustáveis, as RNAs com nodos MCP podem ser treinadas para reconhecimento de padrões (BRAGA et al., 2000).

Em 1969, Minsky e Papert demonstraram que problemas não linearmente separáveis eram impossíveis de serem resolvidos com o uso do perceptron, isto é, problemas onde não é possível obter a solução dividindo o espaço em duas regiões através de uma reta. A repercussão do trabalho de Minsky e Papert foi tanta que o estudo sobre RNAs ficou um pouco esquecido, apesar de alguns pesquisadores continuarem trabalhando na área. No entanto, em 1982, John Hopfield publicou um trabalho que despertou novamente o interesse dos pesquisadores para as RNAs. Hopfield

[1] O paper de McCulloch e Pitts (*"A logical calculus of the ideas immanent in nervous activity"*, Bull. Math. Biophysics, Vol. 5 (1943), pp. 115-133) é explicitamente citado e comentado no famoso relatório de John von Neumann: *"First Draft of a Report on the EDVAC"(University of Pennsylvania, 30-June-1945)*(GODFREY; HENDRY, 1993)

mostrou a relação entre redes recorrentes auto-associativas e sistemas físicos, abrindo espaço também para teorias correntes em Física Teórica. Além disso, a descrição do algoritmo de treinamento *backpropagation* feita pelos pesquisadores Rumelhard, Hinton e Williams, mostrou que as RNAs são capazes de resolver problemas não linearmente separáveis, dando novo impulso aos estudos sobre as RNAs na década de 80.

3.3 O que são Redes Neurais Artificiais

A teoria de Redes Neurais Artificiais consiste em uma forma de computação inspirada no modelo do cérebro humano. Uma RNA é composta por uma ou mais unidades básicas (nodos) que simulam os neurônios biológicos (BRAGA et al., 2000). Estes nodos são interligados através de um grande número de conexões, que estão associadas a pesos que armazenam o conhecimento adquirido pela RNA. Uma das principais aplicações de uma RNA é no reconhecimento de padrões através de exemplos previamente apresentados a um programa computacional, e a partir destes conseguir uma generalização para amostras que não foram apresentadas ao programa. Por exemplo, uma RNA pode ser usada para reconhecimento de caracteres onde são apresentados os seguintes exemplos: "A", "a", "A", "**A**". A partir daí, uma rede treinada com estas entradas poderá ser capaz de reconhecer letras "similares" com a letra "A", isto é, a rede é capaz de reconhecer o padrão da letra "A" através de um treinamento apropriado.

3.4 O Neurônio MCP

O modelo de neurônio proposto por McCulloch e Pitts em 1943 (neurônio MCP) era relativamente simples, baseado no conhecimento sobre o neurônio biológico da época. Na sua representação matemática o modelo tem n entradas $(x_1, x_2, ..., x_n)$, que representam os dendritos do neurônio biológico, e uma saída, que representa o axônio (Figura 3.2).

Para simular as sinapses, são associados pesos $(w_1, w_2, ..., w_n)$ às entradas, os quais podem ser positivos ou negativos, correspondendo à sinapses excitatórias e inibitórias, respectivamente. Uma função de ativação é aplicada para determinar a saída, e essa função utiliza o valor da soma ponderada dos sinais de entrada dos pesos. O neurônio dispara então um sinal de saída (igual a um) quando o resultado da função de ativação ultrapassar um determinado limiar de excitação Θ (*threshold*), caso contrário a saída é zero.

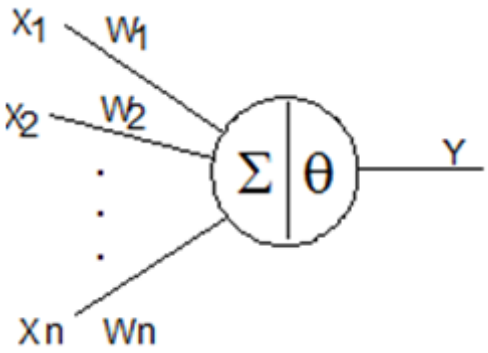

Figura 3.2 - Neurônio de McCulloch e Pitts

A partir do MCP foram criados outros modelos que produzem uma saída não necessariamente binária, dependendo das diferentes funções de ativação utilizadas. Alguns exemplos de funções de ativação são:

- Função linear (Figura 3.3a): $y = \alpha x$
- Função rampa (Figura 3.3b):

$$y = \begin{cases} \alpha & \text{se} \quad x > y \\ x & \text{se} \quad x = y \\ -\alpha & \text{se} \quad x < y \end{cases}$$

- Função degrau (Figura 3.3c):

$$y = \begin{cases} \alpha & \text{se} \quad x > 0 \\ 0 & \text{se} \quad x = y \\ -\alpha & \text{se} \quad x < 0 \end{cases}$$

- Função sigmoidal (Figura 3.3d): $y = \frac{1}{1+e^{-x/T}}$, na qual o parâmetro T determina a inclinação no ponto de inflexão da curva.

3.5 Arquiteturas de Redes Neurais Artificiais

Um dos principais problemas das RNAs é a definição da arquitetura utilizada, pois esta pode restringir o problema a ser resolvido pela rede. Por exemplo, redes com uma única camada de nodos MCP conseguem resolver apenas problemas linearmente separáveis (BRAGA et al., 2000); (HAYKIN, 1999).

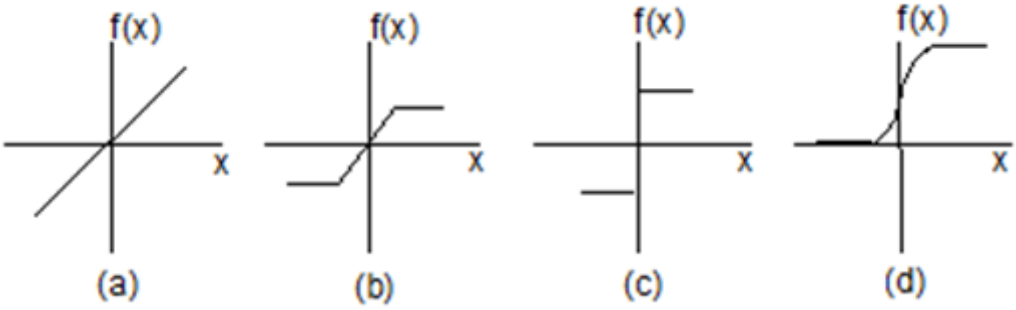

Figura 3.3 - Exemplos de funções de ativação

Para classificar as diferentes arquiteturas das RNAs, os parâmetros considerados são: número de camadas da rede, número de nodos em cada camada, tipo de função de ativação (e seus parâmetros intrínsecos), tipo de conexão entre os nodos (ver abaixo) e topologia da rede. Há também parâmetros da fase de aprendizado de redes supervisionadas - por exemplo: taxa de aprendizado e taxa de momento. As RNAs podem ser classificadas quanto:

a) Ao número de camadas:

- Redes com camada única: só existe um neurônio entre qualquer entrada e qualquer saída da rede.
- Redes com múltiplas camadas: existe mais de um neurônio entre alguma entrada e alguma saída.

b) Ao tipo de conexões entre os nodos:

- *feedforward* (acíclica): a saída de um neurônio na $i-$ésima camada da rede não pode ser usada como entrada de neurônios em camadas de índice menor ou igual a i.
- *feedback* (cíclica): a saída de um neurônio na $i-$ésima camada da rede é usada como entrada de neurônios em camadas de índice menor ou igual a i.

c) À sua conectividade:

- Rede completamente conectada: a saída de um neurônio é entrada para todos os nodos da camada seguinte.

- Rede fracamente (ou parcialmente) conectada: a saída de um neurônio é entrada para um ou mais nodos (porém não todos) da camada seguinte.

Para se obter uma boa generalização, é preciso fornecer para a RNA a maior quantidade de informações possível sobre o problema a ser resolvido. Entretanto, por questões de complexidade computacional procura-se reduzir ao mínimo o número de nodos e a quantidade de conexões entre os mesmos, sendo importante a definição de algoritmos que otimizem os pesos e a arquitetura de uma dada rede (BRAGA et al., 2000).

Para RNAs supervisionadas, é possível configurar automaticamente a rede, onde o problema é formulado como um problema de otimização (CARVALHO et al., 2011); (SAMBATTI et al., 2012).

3.6 Técnicas de Aprendizado das RNAs

As Redes Neurais Artificiais possuem a capacidade de aprender através de exemplos e a partir daí generalizar utilizando padrões que não foram foram apresentados à rede durante o treinamento. A aprendizagem consiste em um processo iterativo de ajuste dos pesos das conexões entre os nodos, guardando ao final do processo o conhecimento que a rede adquiriu. Este processo de aprendizagem é chamado de treinamento. O tipo de algoritmo utilizado para realizar o treinamento da rede é chamado de algoritmo de aprendizado.

Há também diversos métodos para o treinamento das RNAs, os quais podem ser agrupados em dois principais paradigmas: Aprendizado Supervisionado e Aprendizado Não Supervisionado (BRAGA et al., 2000).

O aprendizado supervisionado é chamado desta forma porque a entrada e a saída desejadas são fornecidas por um supervisor externo. A rede tem sua saída calculada e comparada com a saída desejada, recebendo assim informações sobre o erro em relação à saída desejada. A cada padrão de entrada submetido à rede compara-se a resposta desejada com a resposta calculada, e os pesos das conexões são ajustados para minimizar o erro. A soma dos erros quadráticos de todas as saídas é normalmente utilizada como medida de desempenho da rede, e também como função de custo a ser minimizada pelo algoritmo de treinamento (BRAGA et al., 2000).

Já no aprendizado não supervisionado, não há um supervisor externo para acom-

panhar o processo de aprendizado. Neste tipo de técnica, somente os padrões de entrada estão disponíveis para a rede. Durante o processo de treinamento, a rede procura estabelecer uma harmonia com as regularidades estatísticas dos dados de entrada, e a partir daí desenvolver uma habilidade para formar representações internas, podendo codificar características de entrada e criar novas classes ou grupos automaticamente. Este tipo de aprendizado só é possível quando há redundância nos dados de entrada. Sem redundância seria impossível encontrar quaisquer padrões ou características dos dados de entrada (BRAGA et al., 2000).

3.7 *Perceptron* de Camada Única

Em 1958, Frank Rosenblatt propôs um modelo, conhecido como perceptron, composto por uma estrutura de rede tendo como unidades básicas os nodos MCP contendo uma regra de aprendizado. Rosenblatt demonstrou que um nodo MCP treinado com o algoritmo de aprendizado do *perceptron* sempre converge se o problema a ser resolvido for linearmente separável (teorema da convergência do *perceptron*). A topologia original descrita por Rosenblatt era composta por unidades de entrada, por um nível intermediário formado pelas unidades de associação, e por um nível de saída formado pelas unidades de resposta (Figura 3.4).

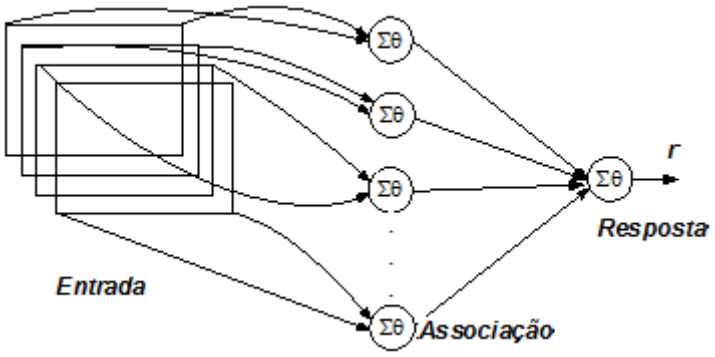

Figura 3.4 - Topologia de um perceptron simples com uma única saída

Embora esta topologia original possua três níveis, ela ficou conhecida como *perceptron* de camada única, pois somente a camada de saída tem propriedades adaptativas. As unidades intermediárias de associação, apesar de compostas por nodos MCP, possuem pesos fixos definidos antes do treinamento. Houve boa receptividade inicial na comunidade científica da época, mas o perceptron não teve vida longa, pois estava restrito a somente de problemas linearmente separáveis. Como mencionado,

apenas na década de 80 é que as Redes Neurais Artificiais ganharam novo impulso, principalmente devido às descrições da rede de *Hopfield* em 1982 e do algoritmo *backpropagation* (que será discutido na próxima seção).

3.8 *Perceptron* de Múltiplas Camadas

Como já foi discutido, RNAs com apenas uma camada resolvem problemas linearmente separáveis, mas para a solução de problemas não linearmente separáveis é necessário uma rede com uma ou mais camadas intermediárias Figura 3.5. Basicamente, uma RNA com uma camada intermediária pode aproximar qualquer função contínua, e a utilização de duas camadas intermediárias permite a aproximação de qualquer função matemática (BRAGA et al., 2000). Entretanto, deve ser ressaltado que dependendo da distribuição dos dados a rede pode convergir para um mínimo local ou demorar muito para encontrar a solução esperada.

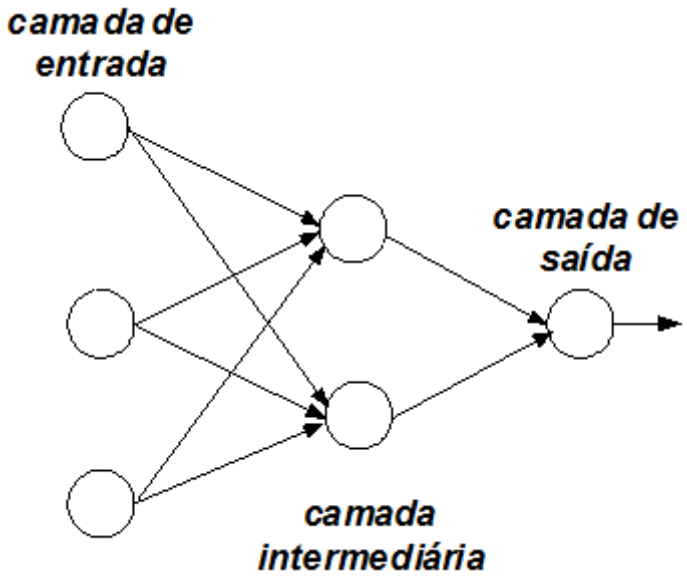

Figura 3.5 - Rede MLP com uma camada intermediária

Para a solução de problemas práticos de reconhecimento de padrões, é necessário definir para a rede um número suficiente de unidades intermediárias para a solução do problema. Contudo, é preciso ter cuidado para evitar o excesso ou a falta de unidades intermediárias. O excesso de unidades pode levar a rede a "memorizar" os padrões de treinamento, ao invés de extrair características gerais permitindo a generalização de padrões não vistos durante o treinamento. Este problema é conhecido como *"overfitting"* (BRAGA et al., 2000). Por outro lado, um número muito reduzido

de unidades pode levar a rede a gastar tempo excessivo tentando encontrar uma solução, problema conhecido como *"underfitting"*. A definição da topologia de uma RNA do tipo Perceptron de Múltiplas Camadas (ou MLP: *Multi-Layer Perceptron*) está intimamente ligada com o número de camadas intermediárias e de nós nesssas camadas, o que geralmente é definido empiricamente. Este número depende fortemente da distribuição dos padrões de treinamento e validação da rede (BRAGA et al., 2000).

Existem vários algoritmos para treinamento de redes MLP. Entretanto, o algoritmo de aprendizado para treinamento mais conhecido destas redes é o algoritmo de retropropagação do erro (*backpropagation*). O *backpropagation* é um algoritmo supervisionado que utiliza pares de entrada e saída desejada. O ajuste dos pesos da rede é feito através de um mecanismo de correção de erros. O treinamento é realizado em fases para frente (*forward*) e para trás (*backward*), com cada uma percorrendo a rede em um sentido, como ilustrado na Figura 3.6. Durante a fase *forward* é definida uma saída para um determinado padrão de entrada, e durante a fase *backward* os pesos das conexões são ajustados de acordo com a saída desejada e a saída fornecida pela rede (BRAGA et al., 2000).

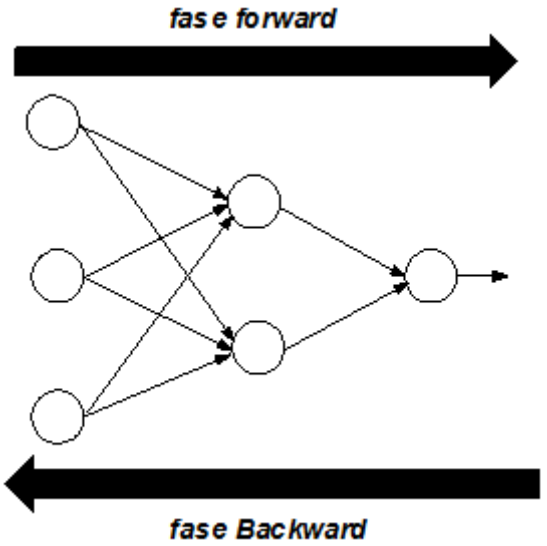

Figura 3.6 - Fluxo de processamento do algoritmo *backpropagation*

O ajuste de pesos no algoritmo *backpropagation* é feito de forma a definir o erro dos nós das camadas intermediárias, possibilitando o ajuste de seus pesos (BRAGA et al., 2000). Sua eficiência gerou uma forte expansão no número de trabalhos teóricos e

de aplicação envolvendo RNAs.

Para definir a topologia de uma RNA e também treinar a rede objetivando minimizar os problemas de *"overfitting"* e *"underfitting"*, algumas alternativas têm sido abordadas. Tais alternativas incluem variações do *backpropagation* como o *rprop* (*Resilient Backpropagation*) e o *quickprop* ((BRAGA et al., 2000)), aplicação de algoritmos de otimização e meta-heurísticas como os Algoritmos Genéticos (MONTANA; DAVIS, 1989); (KITANO, 1990); (ALBUQUERQUE et al., 2004); (MOTA, 2007) e Algoritmo de Colisão de Múltiplas-Partículas (ANOCHI et al., 2012).

4 SISTEMAS DIFUSOS (*FUZZY*)

4.1 Lógica Difusa ou Nebulosa

O termo *Fuzzy* significa algo nebuloso, impreciso ou vago (RODRIGUES; DIMURO, 2011). Assim, podemos dizer que a Lógica Difusa (*Fuzzy*) trabalha com problemas cujos dados apresentam algum tipo de incerteza ou imprecisão. Estas duas características são intrinsecamente ligadas e opostas entre si: quanto mais se aumenta a incerteza mais se diminui a imprecisão e vice-versa (SANDRI; CORREA, 1999). Por exemplo, se afirmamos que um evento ocorrerá entre $8h$ e $9h$, temos certo grau de incerteza e imprecisão neste intervalo. Porém, se desejarmos uma informação mais precisa, tenderemos a aumentar a incerteza, afirmando, por exemplo, que o evento ocorrerá às $8:30h$ com uma probabilidade diferente de 1. Uma informação imprecisa, também pode ser vaga, por exemplo, ao afirmarmos que o evento ocorrerá "por volta de $8:30h$' (SANDRI; CORREA, 1999).

A teoria dos conjuntos e a teoria de probabilidades são utilizadas para tratar a imprecisão e incerteza, respectivamente. Porém, estas teorias nem sempre conseguem captar a riqueza da informação fornecida por seres humanos. A teoria dos conjuntos não é capaz de tratar o aspecto vago da informação e a teoria de probabilidades, na qual a probabilidade de um evento determina completamente a probabilidade do evento contrário, é mais adaptada para tratar de informações "frequentistas"do que aquelas fornecidas por seres humanos (SANDRI; CORREA, 1999).

4.2 Conjuntos Difusos

Na teoria dos conjuntos, um elemento x em um conjunto universo X, pertence a um conjunto A ou não. Essa pertinência é binária e pode ser representada matematicamente com a função indicador (TANSCHEIT, 2013):

$$x_A(x) = \begin{cases} 1 & \text{se} \quad x \in A \\ 0 & \text{se} \quad x \notin A \end{cases}$$

Na função acima, $x_A(x)$ indica a pertinência do elemento x em relação ao conjunto A e os símbolos $\in$ e $\notin$ denotam pertence e não pertence, respectivamente.

Na teoria dos conjuntos difusos, introduzida por (ZADEH, 1965), um conjunto nebuloso A de um universo Ω é definido por uma função de pertinência $A : \Omega \to [0, 1]$

[1]. Essa função associa a cada elemento x de Ω o grau $\mu_A(x)$, com o qual x pertence a A. A função de pertinência $A(x)$ indica o grau de compatibilidade entre x e o conceito expresso por A (SANDRI; CORREA, 1999):

- $A(x) = 1$, indica que x é completamente compatível com A.
- $A(x) = 0$, indica que x é completamente incompatível com A.
- $0 < A(x) < 1$, indica que x é parcialmente compatível com A, com grau de compatibilidade $A(x)$.

Um conjunto A da teoria dos conjuntos pode ser visto como um caso particular de um conjunto nebuloso, no qual $A : \Omega \to \{0, 1\}$, ou seja, a pertinência é do tipo completamente compatível (pertence) ou completamente incompatível (não pertence), e não gradual como para os conjuntos difusos (SANDRI; CORREA, 1999).

4.3 Operações com conjuntos difusos

Da mesma forma que existem operações sobre conjuntos, também é possível definir operações sobre conjuntos difusos, por exemplo, União, Intersecção e Negação. Para isso, apresentaremos os conceitos de Comutatividade, Associatividade e Monotonicidade. Seja uma função definida como:$\Phi : [0, 1]^2 \to [0, 1]$, dizemos que tal função é:

- Comutativa. Se, e somente se: $\Phi(a, b) = \Phi(b, a)$;
- Associativa. Se, e somente se: $\Phi(a, \Phi(b, c)) = \Phi(\Phi(a, b), c)$;
- Monotônica. Se, e somente se: $\Phi(a, b) \leq \Phi(c, d) \to a \leq c \; e \; b \leq d$;

Um operador de Intersecção $\top : [0, 1]^2 \to [0, 1]$ deve satisfazer as propriedades de Comutatividade, Associatividade e Monotonicidade, além do seguinte Elemento Neutro: $\top(a, 1) = a$.

Um operador de União $\perp[0, 1]^2 \to [0, 1]$ deve satisfazer as propriedades de Comutatividade, Associatividade e Monotonicidade, além do seguinte Elemento Neutro: $\perp(a, 0) = a$.

[1]Neste texto será utilizado o símbolo A para denotar um conjunto difuso e também para denotar uma função de pertinência. Alguns autores utilizam A para denotar o conjunto e μ_A para a função de pertinência, ou ainda, A para o conjunto e $\tilde{A}$ para a função.

Na teoria dos conjuntos difusos as famílias de operadores de Intersecção são conhecidas como t-normas ou normas $\top$ e as famílias de operadores de União são conhecidas como t-conormas ou conormas $\perp$ (SANDRI; CORREA, 1999).

O principal operador de negação (ou complemento) pode ser definido como: $a = 1 - a$, $a \in [0, 1]$. Porém outros operadores podem ser definidos e utilizados ((SANDRI; CORREA, 1999)).

4.4 Variáveis Linguísticas

Uma variável linguística é uma variável cujos valores são nomes de conjuntos difusos (TANSCHEIT, 2013). Por exemplo, a altura de alguns indivíduos em um determinado processo pode ser uma variável linguística assumindo valores: baixa, média e alta. Estes valores são representados por intermédio de conjuntos difusos, representados por funções de pertinência. A Figura 4.1 ilustra este exemplo (TANSCHEIT, 2013).

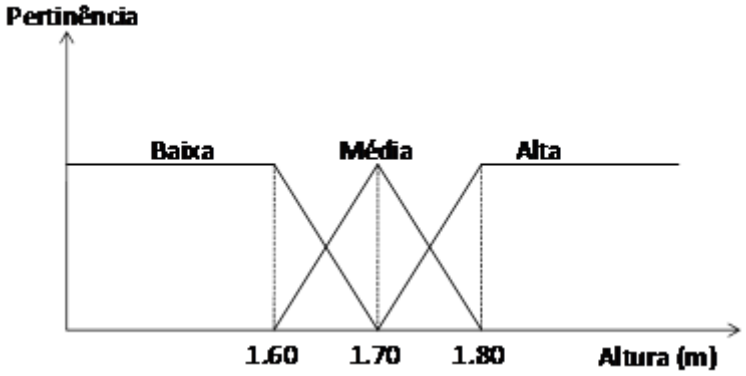

Figura 4.1 - Exemplo de variável linguística (altura)

A principal função das variáveis linguísticas é fornecer uma maneira sistemática para uma caracterização aproximada de fenômenos complexos ou definidos com um baixo grau de formalismo. Em essência, a utilização do tipo de descrição linguística empregada por seres humanos, e não de variáveis quantificadas, permite o tratamento de sistemas que são muito complexos para serem analisados através de termos matemáticos convencionais (TANSCHEIT, 2013).

4.5 Sistema de Inferência Difuso

A Figura 4.2 ilustra um Sistema de Inferência baseado em regras difusas descrito em (TANSCHEIT, 2013).

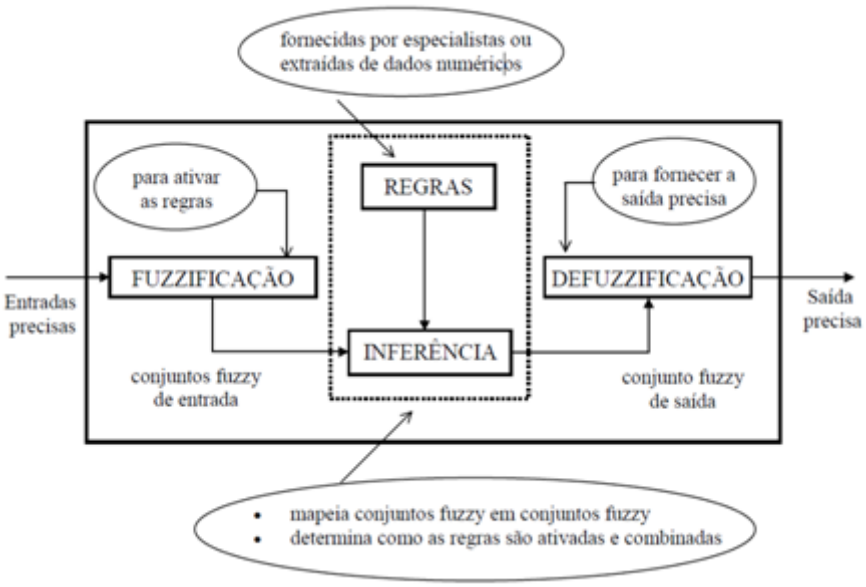

Figura 4.2 - Exemplo de Sistema de Inferência Baseado em Regras Difusas

Neste sistema, as entradas são precisas (não difusas), resultantes de um conjunto de dados (adquiridos através de medições, por exemplo). Em função disto, é necessário mapear os valores de entrada para conjuntos difusos (por exemplo, utilizando variáveis linguísticas), este processo é conhecido como codificação (*fuzzificação* na Figura 4.2). A partir daí, são aplicadas regras de inferência nestes conjuntos difusos. Tais regras, em geral, são fornecidas por algum especialista (pode ser um especialista humano, algum tipo de sistema ou mesmo algum modelo matemático), estas regras podem ter vários termos na premissa, ligadas por conjunção, como foi discutido na seção 4.3. Após a aplicação das regras, na maioria dos sistemas difusos, é necessário realizar um processo de decodificação (*defuzzificação* na Figura 4.2) dos valores obtidos. Tal processo consiste em mapear os valores obtidos (codificados) para valores precisos (não codificados), para serem interpretados de acordo com seus significados em relação aos valores de entrada do sistema, existem diversos métodos de *defuzzificação*, como: Centróide (ou centro de massa), Princípio da máxima associação (método da altura), Método da média ponderada, Média da associação máxima e Centro das somas.

4.6 O Modelo Takagi-Sugeno

O modelo Takagi-Sugeno foi proposto originalmente por (TAKAGI; SUGENO, 1985), porém foi aprimorado por (SUGENO; KANG, 1986), sendo conhecido também como modelo Takagi-Sugeno-Kang. Este modelo é um sistema de inferência capaz de des-

crever sistemas dinâmicos não-lineares através de um conjunto de sistemas dinâmicos lineares.

A principal característica de um sistema difuso é a representação de informações (conhecimento) condicionadas por regras do tipo SE-ENTÃO (VITOR, 2011).

Especificamente, o sistema difuso Takagi-Sugeno é descrito pelas regras de inferência do tipo SE-ENTÃO, que representam localmente relações lineares entre a entrada e a saída de um sistema (TEIXEIRA et al., 2000). Por exemplo:

$$\begin{aligned} SE\ x\ \text{é}\ A \quad & (premissa) \\ ENTÃO\ y = ax + b \quad & (consequente) \end{aligned}$$

No exemplo acima, é possível notar que a parte relativa aos consequentes das regras é expressa por uma função polinomial, possibilitando representar modelos numéricos e uma forma mais adequada a aplicações numéricas. Para isso, no sistema Takagi-Sugeno é atribuído um peso médio dos valores nos consequentes das regras.

No exemplo apresentado ($y = ax + b$), os parâmetros a e b devem ser escolhidos de maneira a se adequarem ao problema, porém não existe um modelo fixo de como realizar este ajuste. Considerando que um modelo pode ter várias regras e cada uma com muitos parâmetros, esta tarefa pode se tornar muito custosa. O modelo ANFIS (que será discutido no capítulo seguinte) é um Sistema de Inferência Difuso baseado no modelo Takagi-Sugeno que realiza este ajuste automaticamente através do treinamento de uma Rede Neural Artificial.

A Figura 4.3 mostra um exemplo disponível em (MATHWORKS, 2011) de um sistema difuso Takagi-Sugeno para calcular a gorjeta em um restaurante. As entradas têm duas variáveis linguísticas: qualidade do serviço (ruim, boa ou excelente) e qualidade da comida (ruim ou boa) e a saída é o valor da gorjeta (baixa, média ou alta).

4.7 Modelo de Mamdani

Em 1974, (MAMDANI, 1974) propôs um método para sistemas de controle difuso, neste método as regras de inferência possuem relações difusas tanto nos antecedentes (premissa) como nos consequentes.

Assim, as regras no modelo Mamdani são do tipo:

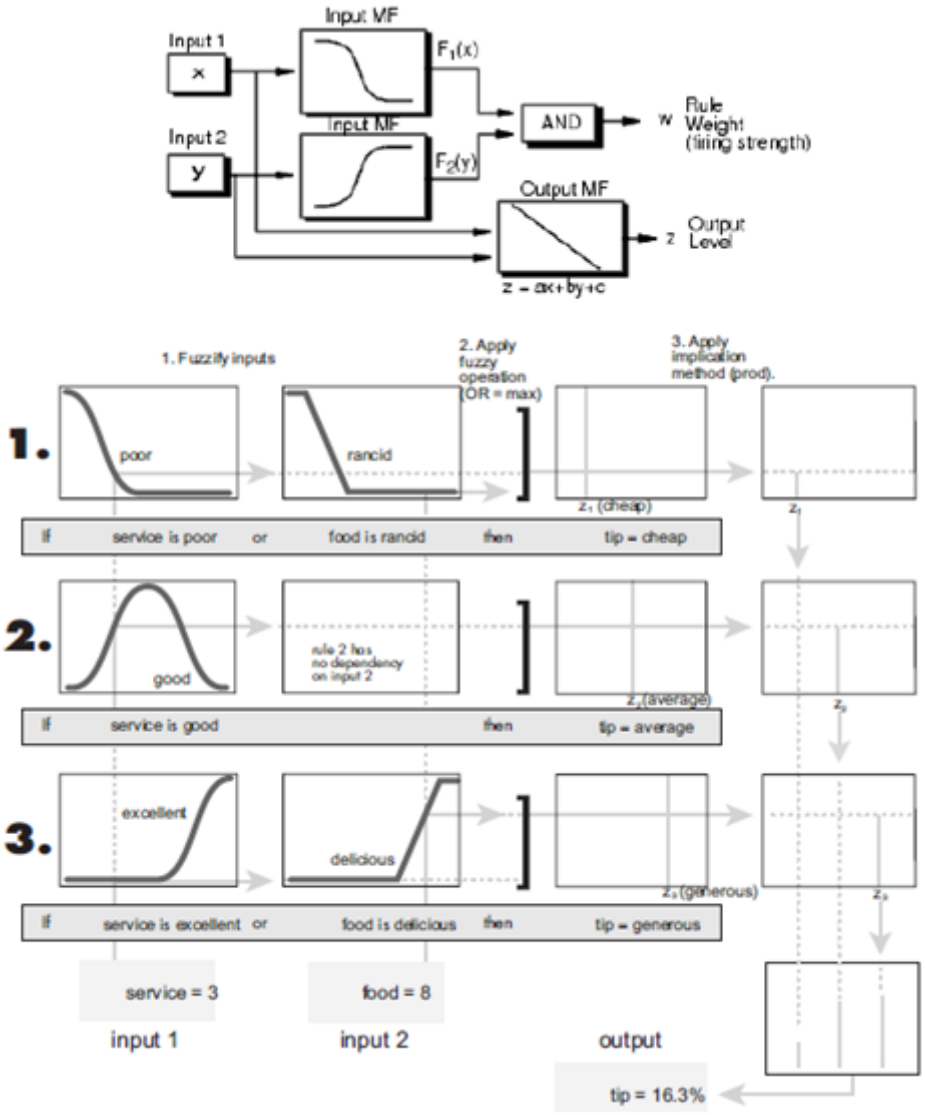

Figura 4.3 - Exemplo de sistema Takagi-Sugeno.
Fonte: (MATHWORKS, 2011)

$$SE\ x \text{ é } A \quad (premissa)$$
$$ENTÃO\ y \text{ é } B \quad (consequente)$$

No exemplo acima, x e y são, respectivamente, as variáveis de entrada e saída. A e B são variáveis linguísticas associadas aos conjuntos difusos que descrevem linguisticamente estas variáveis.

A diferença fundamental entre o modelo Mamdani e o modelo Takagi-Sugeno, é que no modelo Mamdani a parte relativa aos consequentes das regras é expressa por um

conjunto difuso, já no modelo Takagi-Sugeno a parte relativa aos consequentes das regras é expressa por uma função polinomial.

No modelo Mamdani as variáveis de entrada são transformadas em conjuntos difusos (*fuzzificação*) e, posteriormente, os conjuntos difusos gerados na saída são transformados em grandezas numéricas proporcionais (*defuzzificação*).

A Figura 4.4 mostra um exemplo disponível em (MATHWORKS, 2011) de um sistema difuso Mamdani para calcular a gorjeta em um restaurante. As entradas tem duas variáveis linguísticas: qualidade do serviço (ruim, boa ou excelente) e qualidade da comida (ruim ou boa) e a saída é o valor da gorjeta (baixa, média ou alta). No exemplo da Figura 4.4, o método de *defuzzificação* utilizado foi o centróide (ou centro de massa), porém, como foi mencionado na seção 4.5, existem outras técnicas de *defuzzificação*.

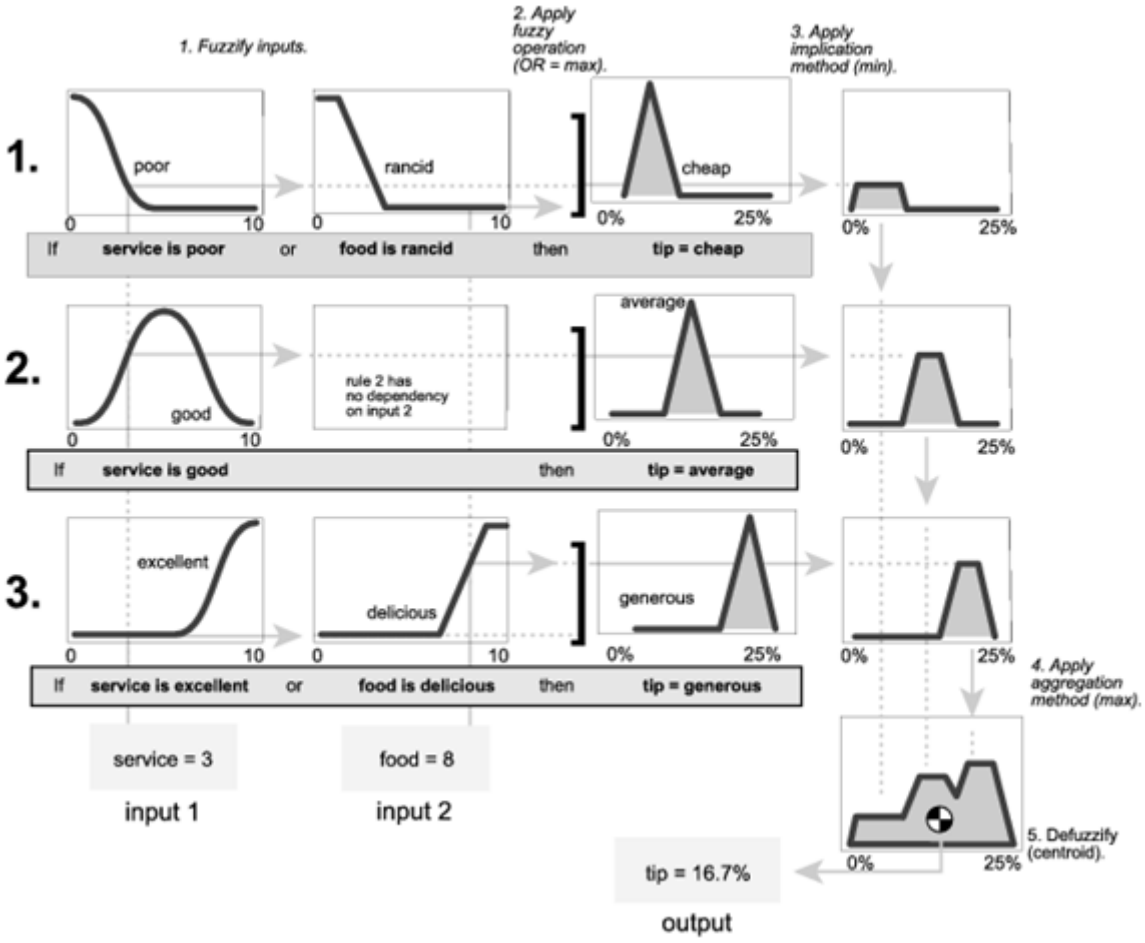

Figura 4.4 - Exemplo de sistema Mamdani.
Fonte: (MATHWORKS, 2011)

5 SISTEMAS NEURO-DIFUSOS

Modelos *Fuzzy* e Redes Neurais Artificiais (RNAs) são duas classes de modelos não lineares cuja importância tem crescido sistematicamente ao longo dos anos. Ambas apresentam uma propriedade fundamental que impulsionou seu desenvolvimento e aplicações nas mais diversas áreas são aproximadores universais. Porém, o conhecimento de uma Rede Neural está nos valores dos pesos que são ajustados após o treinamento, esse conhecimento é difícil de ser analisado, fazendo com que as Redes Neurais funcionem como uma "caixa preta", os dados são apresentados e a RNA os classifica, sem que o usuário saiba exatamente como esses dados foram classificados. No caso dos sistemas difusos, o conhecimento está nas regras que podem ser criadas por um especialista ou geradas automaticamente através de algum algoritmo. O sistema ANFIS (descrito na seção a seguir) cria um modelo difuso do tipo Takagi-Sugeno, o qual gera um conjunto completo de regras, porém essas regras não são de fácil interpretabilidade. Os sistemas difusos do tipo Mamdani trabalham com regras mais interpretáveis. Para gerar um conjunto regras interpretáveis automaticamente, utilizamos o Software GUAJE (*Java Environment for Generating Understandable and Accurate*), que será descrito na seção a seguir.

As Redes Neurais Artificiais consistem em uma ótima ferramenta para trabalhar com problemas relacionados a reconhecimento de padrões, porém apresentam certa dificuldade para explicar como as respostas são obtidas, funcionando como uma espécie de "caixa-preta". Já os Sistemas de Inferência Difusos, mesmo trabalhando com informações imprecisas, apresentam maior facilidade para explicar como os resultados são obtidos. Um Sistema Neuro-Difuso pode utilizar o aprendizado das Redes Neurais Artificiais para ajustar os parâmetros de um Sistema Difuso, com o objetivo de potencializar as vantagens de cada modelo e minimizar suas desvantagens.

O termo neuro-difuso surgiu em meados da década de 1980 e significa uma mescla de RNAs com Sistemas de Inferência Difusos, resultando em um sistema inteligente híbrido que potencializa as características destes dois importantes paradigmas (REZENDE, 2003). A rigor, qualquer sistema que combine os paradigmas de sistemas difusos e sistemas conexionistas poderia ser chamado de "Neuro-Difuso", como, por exemplo, a utilização de um controlador nebuloso para alterar dinamicamente a taxa de aprendizado de uma rede neural. No entanto, o termo é utilizado para um tipo específico de sistema que de certa forma engloba os dois paradigmas. Nestes sistemas, os termos e regras de um sistema nebuloso são aprendidos mediante a apresentação de pares (entrada, saída desejada) (SANDRI; CORREA, 1999).

5.1 Modelo Neuro-Fuzzy ANFIS

O modelo ANFIS (do inglês, *Adaptive-Network-Based Fuzzy Inference System*) é uma Rede Neural Artificial, proposta por (JANG, 1993), que consiste em implementar um Sistema de Inferência Difuso através da própria RNA, de forma que os algoritmos de aprendizado possam ser utilizados para ajustar o Sistema de Inferência Difuso (JACINTHO, 2010).A Figura 5.1 representa um exemplo de sistema ANFIS.

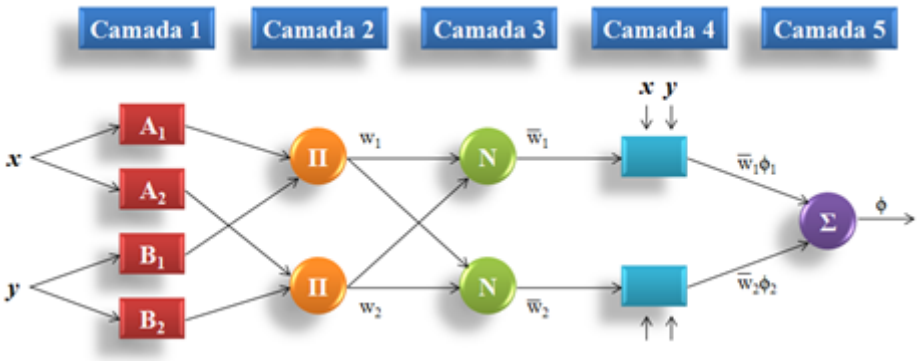

Figura 5.1 - Exemplo de Sistema ANFIS.
Fonte: Adaptada de (JACINTHO, 2010)

O modelo apresentado na Figura 5.1 é formado por cinco camadas, as quais serão descritas a seguir:

- Camada 1 - Sendo x e y os valores de entrada para o sistema, os nós desta camada (A_1, A_2, B_1 e B_2) representam o grau de pertinência destes nós em relação aos valores de entrada, ou seja, esta camada realiza a fase de *fuzzificação* do sistema. Qualquer função contínua e diferenciável no intervalo $[0, 1]$ pode ser utilizada como função de ativação nesta camada, por exemplo, a função sino (equação 5.1), na qual os valores a, b e c são parâmetros.

$$\mu_A(x) = \frac{1}{1 + [(\frac{x-c}{a})^2]^b} \tag{5.1}$$

- Camada 2 - As saídas dos nós desta camada representam o "grau de disparo de uma regra". A Equação 5.2 é utilizada como função de ativação nesta camada, porém qualquer operador de norma T que represente o operador AND pode ser utilizado como função de ativação nesta camada (JANG,

1993).

$$w_i = \top(\mu_{A_i}(x), \mu_{B_i}(x)),\ i = 1, 2 \tag{5.2}$$

- Camada 3 - Cada nó desta camada funciona como um operador de normalização do grau de disparo das regras, definido, de acordo com (JANG, 1993), como a Equação 5.3.

$$\overline{w_i} = \frac{w_i}{w_1 + w_2},\ i = 1, 2 \tag{5.3}$$

- Camada 4 - Esta camada calcula o produto entre as saídas da camada 3 (grau de disparo normalizado) e os valores de entrada x e y, ponderados pelos valores (p, q e r), os quais (JANG, 1993) chamou de parâmetros consequentes, de acordo com a equação 5.4.

$$\overline{w_i} z_i = \overline{w_i}(p_i x + q_i y + r_i),\ i = 1, 2 \tag{5.4}$$

- Camada 5 - Esta camada realiza a fase de *defuzzificação* do sistema. É formada por apenas um nó que realiza o somatório das saídas da camada 4, de acordo com a Equação 5.5.

$$z = \sum \overline{w_i} z_i = \frac{\sum w_i z_i}{\sum w_i},\ i = 1, 2 \tag{5.5}$$

O sistema ANFIS, apresentado na Figura 4.4, utiliza o algoritmo *backpropagation* para calcular os parâmetros antecedentes das regras (Equação 5.1) e o algoritmo LMS (do inglês, *Least Mean Square*)[1] para calcular os parâmetros consequentes (Equação 5.4). Cada iteração do algoritmo é dividida em duas partes (JACINTHO, 2010):

a) Os dados de entrada são propagados e os parâmetros consequentes são calculados de acordo com o LMS, enquanto os parâmetros antecedentes são fixados durante todo o ciclo através do conjunto de treinamento.

b) As taxas de erro são retropropagadas e o *backpropagation* é utilizado para atualizar os parâmetros antecedentes, enquanto os parâmetros consequentes fixados.

[1] A descrição e formalização do algoritmo LMS pode ser encontrada em (HAYKIN, 1999)

5.2 Modelo Neuro-fuzzy GUAJE

A interpretabilidade de um sistema difuso envolve algum conhecimento específico do usuário final, o qual interpreta sua descrição linguística como objetivo de conceber o significado do comportamento do sistema. Em consequência, caracterizar e avaliar a interpretabilidade são uma tarefa muito subjetiva e que depende fortemente da perspectiva do usuário (ALONSO, 2011).

A interpretabilidade é tomada em consideração juntamente com o processo de modelagem de uma maneira geral. Claro que, a acurácia não é esquecida, porque todo o tipo de sistema deve alcançar, pelo menos, um mínimo de precisão, sendo completamente inútil contrário. Assim, certa perda de precisão pode ser tolerada em troca de um modelo mais interpretável (ALONSO; MAGDALENA, 2010).

O GUAJE (*Java Environment for Generating Understandable and Accurate Models*) é um *software* livre que implementa uma metodologia de modelagem difusa chamada *Highly Interpretable Linguistic Knowledge* (HILK) (ALONSO et al., 2008); (ALONSO; MAGDALENA, 2011b). Foi projetado e desenvolvido com o objetivo de produzir sistemas altamente interpretáveis e manter uma boa relação entre interpretabilidade e acurácia. O processo de modelagem é constituído pelas seguintes etapas (ALONSO et al., 2012):

- Pré-processamento dos dados: inclui visualização dos dados, análise e amostragem.
- Seleção de recursos: com o foco em identificar as variáveis de entrada mais significativas.
- Particionamento: consiste em caracterizar cada variável de entrada como uma variável linguística com um número justificável de termos linguísticos. Sua semântica deve ser compatível com o conhecimento do especialista e é descrita por meio de fortes partições difusas com o objetivo de maximizar a interpretabilidade.
- Definição das regras: Uma vez definidas todas as variáveis de entrada, o passo seguinte é definir uma semântica global como base das regras. Como resultado, todas as regras compartilham os mesmos termos linguísticos previamente definidos. Assim, o comportamento do sistema pode ser descrito como um conjunto de regras linguísticas do tipo "se-então". As regras podem ser criadas automaticamente com técnicas de aprendizado de máquina

e/ou fornecidas diretamente por um especialista.

- Verificação da base de regras: esta etapa de modelagem é responsável por verificar a consistência da base de regras previamente definida.
- Melhoria da base de conhecimento: o objetivo desta fase é melhorar a relação entre interpretabilidade e acurácia. A primeira etapa consiste em uma simplificação linguística com o objetivo de aumentar a interpretabilidade, enquanto preserva a acurácia já alcançada, em seguida é feito um refinamento na partição destinada a aumentar a acurácia, porém sem comprometer a interpretabilidade.
- Validação da base de conhecimento: verifica se o sistema difuso gerado satisfaz as expectativas.
- Avaliação da qualidade: avalia os resultados quanto à interpretabilidade e acurácia.

O fluxo de trabalho das etapas acima é descrito por (ALONSO; MAGDALENA, 2011a) da seguinte forma. Na primeira etapa, os dados experimentais disponíveis devem ser pré-processados e convertidos para o formato utilizado pelo GUAJE. Em seguida, um processo de mineração de dados é necessário para selecionar as variáveis de entrada mais significativas. A seguir, a etapa de particionamento é baseada na definição de variáveis linguísticas caracterizadas por partições difusas, em seguida as partições geradas por um especialista e as partições geradas automaticamente a partir de dados experimentais são comparadas. As melhores partições, de acordo com a distribuição de dados e conhecimentos técnicos, são selecionadas para cada variável de entrada. Então, dois conjuntos de regras linguísticas descrevem o comportamento do sistema de forma a combinar as variáveis linguísticas geradas anteriormente. São regras do tipo se-então, onde as premissas e as conclusões são expressas por meio de proposições linguísticas. Em seguida, os conjuntos de regras são integrados após as verificações de integridade e consistência. Em seguida, a base de conhecimento resultante pode ser melhorada tanto em relação à interpretabilidade quanto à precisão. Finalmente, após validar o sistema difuso final (base de conhecimento + mecanismo de inferência), é possível gerar o código nativo para executar o modelo com o próprio GUAJE ou até mesmo exportar o modelo gerado para algum dos seguintes sistemas integrados ao GUAJE: FisPro, XFuzzy ou MATLAB.

6 *BRED VECTORS* E PREVISIBILIDADE

6.1 Definição de *bred vectors*

Como foi discutido na Seção 2.6, nos trabalhos desenvolvidos por (EVANS, 2004) e (CINTRA; VELHO, 2008) os autores utilizaram uma técnica proposta por (TOTH; KALNAY, 1997) conhecida como *"bred vectors"* ou *"breeding method"* para avaliar a previsão do comportamento em sistemas caóticos, principalmente no Modelo de Lorenz apresentado no Capítulo 2. O método *"bred vectors"* consiste nos seguintes passos (TOTH; KALNAY, 1997):

a) Adicionar uma perturbação pequena e arbitrária para a análise do sistema (estado inicial), sendo t_0 o tempo inicial.

b) Integrar ambos os modelos, o que sofreu a perturbação e o que não sofreu para um curto período $(t_1 - t_0)$.

c) Subtrair os resultados obtidos entre os modelos (este resultado é o *"bred vector"*).

d) Normalizar o *"bred vector"*.

e) Adicionar uma perturbação à próxima análise (tempo t_1).

f) Repetir $(b) - (e)$ nos tempos subseqüentes.

Nota-se que, uma vez que a perturbação inicial é introduzida no passo (a), o desenvolvimento do campo de perturbação é determinado dinamicamente pelo fluxo de evolução do sistema (TOTH; KALNAY, 1997). Através das sucessivas aplicações e comparações com os resultados esperados (sem perturbação) é possível inferir alguns padrões e até mesmo algumas regras sobre o comportamento do sistema. A Figura 6.1 exemplifica o conceito de *"bred vector"*.

A técnica de *"bred vectors"* para previsibilidade em sistemas caóticos tem sido bastante utilizada atualmente, principalmente em pesquisas relacionadas à atmosfera, clima ou tempo, ver (EVANS, 2004), (CINTRA; VELHO, 2008), (TOTH; KALNAY, 1997) e (NEWMAN et al., 2003).

Através da Figura 2.1 é possível notar que o Sistema de Lorenz forma duas regiões de atração, as quais os autores de (EVANS, 2004) chamaram de Regime Quente (*"Warm Regime"*) e Regime Frio (*"Cold Regime"*). No referido trabalho os autores compararam os resultados com e sem perturbação e chegaram a duas regras sobre o

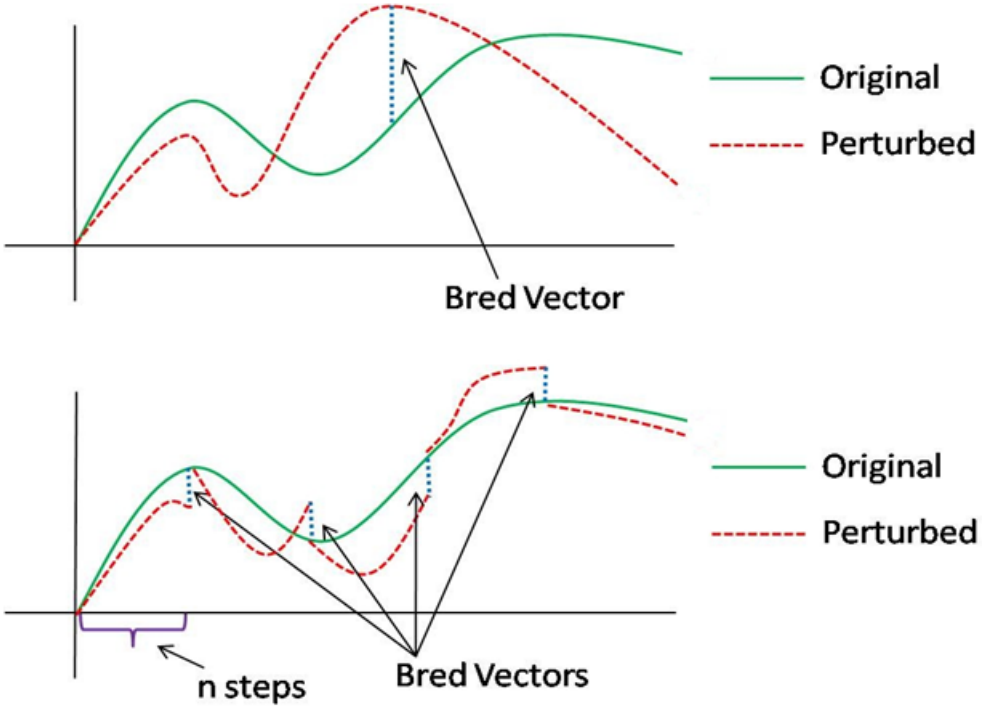

Figura 6.1 - Bred Vectors

comportamento do Sistema de Lorenz (a Figura 6.2 mostra uma forma de visualizar estas regras):

Regra 1: Quando a taxa de crescimento (valor do *"bred vector"*) for superior a 0,064 sobre um período de 8 passos, conforme indicado pela presença de uma ou mais estrelas vermelhas (Figura 6.2) o atual regime terminará após completar a órbita atual.

Regra 2: O comprimento do novo regime (número de voltas) é proporcional ao número de estrelas vermelhas. Por exemplo, a presença de cinco ou mais estrelas no antigo regime, indicando um forte crescimento sustentado, implica que o novo regime vai durar quatro órbitas ou mais.

Na Figura 6.2, cada estrela colorida é uma amostra do tamanho do *"bred vectors"* após 8 passos (pontos), ou seja, a distância do ponto com perturbação em relação ao ponto da trajetória original. Cada cor significa uma seguinte faixa de valores:

- Azul: *"bred vectors"* menor que 0. Neste caso significa que a trajetória perturbada está se aproximando da trajetória original.

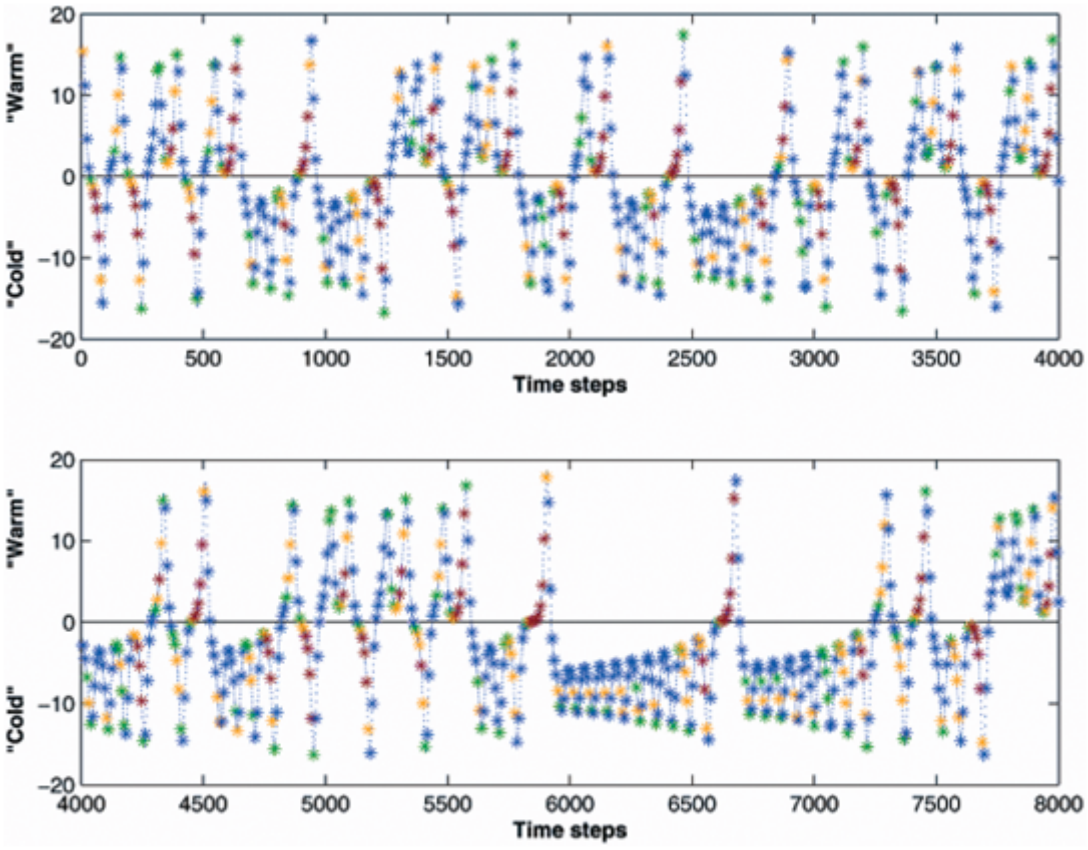

Figura 6.2 - Série temporal $x(t)$ em função do número de passos

- Verde: *"bred vectors"* maior que 0 e menor ou igual 0,032.
- Amarelo: *"bred vectors"* maior que 0,032 e menor ou igual 0,064.
- Vermelho: *"bred vectors"* maior que 0,064.

Cada volta que gera um pico no gráfico representa uma órbita no atrator (Figura 2.1). Os autores chegaram às regras citadas através de observação e contando manualmente o número de estrelas em cada volta, assim conseguiram identificar padrões e concluíram as regras.

O objetivo deste trabalho é utilizar uma ferramenta que automatize esta tarefa e consiga reconhecer padrões apresentados na Figura 6.2. Para isso, utilizamos o sistema Neuro-Difuso ANFIS, que trabalha com pares de entrada/saída desejada e através de um processo de treinamento utilizando estes pares gera um sistema difuso do tipo Takagi-Sugeno. Tal sistema pode ser utilizado para calcular uma saída, dado uma determinada entrada, em geral, uma entrada que não foi utilizada na etapa de treinamento, pois o objetivo é conseguir uma generalização que será utilizada como

classificação. Neste trabalho utilizamos como entradas o número de estrelas de cada cor em cada uma das órbitas da Figura 6.2, e como saída a informação se o a trajetória vai mudar de região e, caso mude, o número de órbitas que vai durar a nova região.

6.2 Metodologia

Como foi discutido na seção anterior, nos trabalhos de (CINTRA; VELHO, 2008) e de (EVANS, 2004) os autores chegaram a duas regras sobre o comportamento de um sistema caótico, nestes casos o Sistema de Lorenz, utilizando a técnica conhecida como *"bred vectors"*. O objetivo deste trabalho é identificar padrões em sistemas caóticos, e a partir destes padrões, realizar uma previsibilidade sobre o comportamento do sistema. Desta forma, o problema de previsibilidade em sistemas dinâmicos com comportamento caótico se torna um problema de classificação e reconhecimento de padrões. Como classificador, foi utilizado o Sistema Neuro-Difuso ANFIS, discutido na seção 5.1, e também foram utilizadas Redes Neurais Artificiais do tipo MLP (*Multi Layer Perceptron*). A ideia é estimar limiares através do crescimento *"bred vectors"* e com isso estimar a alteração do lugar das órbitas dentro do atrator estranho.

Note que na Figura 6.2 foram gerados 8000 pontos na trajetória, através destes 8000 pontos existem em torno de 100 órbitas (cada órbita é uma volta que gera um pico no gráfico). Cada uma destas órbitas foi utilizada como um par de entradas/saída desejada no nosso sistema, e cada um destes pares é um padrão a ser utilizado para treinamento, validação ou teste. Com o objetivo de conseguir uma boa generalização, geramos em torno de 100 mil pontos na trajetória e assim conseguimos uma amostra com 1275 padrões (órbitas). Dividimos a amostra em três partes: treinamento (800 padrões), validação (200 padrões) e teste (275 padrões). Cada padrão possui 4 entradas, sendo cada entrada o número de estrelas de uma determinada cor em cada órbita. A saída desejada é definida de acordo com o número de voltas que permanecerão na região atual ou, caso for mudar de região, de acordo com o número de voltas que permanecerão na nova região. Os testes com o ANFIS foram realizados com o software MATLAB versão R2010a, o qual possui uma implementação do ANFIS que trabalha com arquivos de treinamento, validação e testes com pares de entradas / saídas desejadas e que ao final do treinamento gera um arquivo com extensão ".fis", que é um sistema difuso do tipo Takagi-Sugeno utilizado no MATLAB. Para fins de comparação e também para corroborar os resultados, foi utilizado outro método de reconhecimento de padrões, as Redes Neurais artificiais, neste caso, utilizamos o si-

mulador WEKA (WEKA, 2012), trabalhando sempre com uma camada oculta com o número de neurônios nesta camada calculado pela seguinte fórmula: (número de variáveis de entrada + número de classes)/2. O algoritmo de treinamento utilizado foi o *backpropagation* padrão com taxa de aprendizado 0,3 e termo de *momentum* 0,2, além disso, foram utilizadas sempre 5000 épocas para o treinamento da rede. Mais informações sobre as Redes Neurais Artificiais podem ser encontrados em (HAYKIN, 1999) e em (BRAGA et al., 2000).

7 RESULTADOS

Como foi dito no capítulo anterior, na Figura 6.2, cada estrela colorida é uma amostra do tamanho do *"bred vector"* após 8 passos (pontos), ou seja, a distância do ponto com perturbação em relação ao ponto da trajetória original, e cada cor representa uma faixa (intervalo de valores) de *"bred vectors"*. Em todos os experimentos, que serão descritos a seguir, utilizamos 4 entradas, sendo cada entrada o número de estrelas de cada cor (azul, verde amarelo e vermelho) em cada volta.

7.1 Sistema de Lorenz

Em princípio, trabalhamos com duas classes de saída:

- Classe 0: A trajetória se manterá na atual região.
- Classe 1: A trajetória mudará de região.

Utilizando as 275 amostras de testes, chegamos a 91,63% de acertos com o ANFIS. A matriz de confusão da Tabela 7.1 mostra esse resultado. Nesta matriz, as linhas representam a saída calculada e as colunas representam a saída desejada, desta forma, a diagonal principal representa os acertos do sistema.

Utilizando uma Rede Neural Artificial a taxa de acertos foi de 90,9% com os mesmos dados.

Tabela 7.1 - Matriz de confusão (linhas representam a saída calculada e colunas representam a saída desejada).

Classes	**0**	**1**
0	**139**	15
1	8	**113**

Em seguida, trabalhamos com as seguintes classes de saída:

- Classe 0: A trajetória se manterá na atual região.
- Classe x: A trajetória mudará de região e se manterá na nova região por x voltas ($x = 1, 2, 3, ...$).

Aumentando o número de classes para 9 e utilizando as 275 amostras de testes, chegamos a 81,45% de acertos utilizando o ANFIS, a matriz de confusão (Tabela 7.2) mostra esse resultado. Nesta matriz, as linhas representam a saída calculada e as colunas representam a saída desejada, desta forma, a diagonal principal representa os acertos do sistema.

Utilizando uma Rede Neural Artificial a taxa de acertos foi de 80% com os mesmos dados. Ou seja, o classificador neuro-fuzzy apresentou desempenho similar ao classificador neural.

Tabela 7.2 - Matriz de confusão (linhas representam a saída calculada e colunas representam a saída desejada).

Classes	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
0	**140**	7	2	3	0	0	0	0	1	1
1	5	**41**	10	0	1	0	0	0	0	1
2	0	0	**27**	2	0	0	0	0	0	0
3	0	0	2	**6**	3	0	0	0	0	0
4	0	0	1	2	**5**	1	0	0	0	2
5	0	0	0	0	3	**3**	0	0	0	0
6	0	0	0	0	0	2	**0**	0	0	0
7	0	0	0	0	0	1	0	**0**	1	0
8	0	0	0	0	0	0	0	0	**0**	0
9	0	0	0	0	0	0	0	0	0	**2**

Os resultados da Tabela 7.2 mostram se a trajetória vai permanecer na mesma região e, caso mude, quantas voltas vai permanecer na nova região. Porém, em algumas aplicações pode ser necessária apenas uma estimativa de quantas voltas a trajetória vai permanecer na nova região, para isso trabalhamos com 4 classes de saída:

- Classe 0: A trajetória se manterá na atual região.
- Classe 1: A trajetória mudará de região e se manterá na nova região de 1 a 3 voltas.
- Classe 2: A trajetória mudará de região e se manterá na nova região de 4 a 6 voltas.
- Classe 3: A trajetória mudará de região e se manterá na nova região mais de 6 voltas.

Utilizando as 275 amostras de testes, chegamos a 89,81% de acertos com o ANFIS, a matriz de confusão da Tabela 7.3 mostra esse resultado. Nesta matriz, as linhas representam a saída calculada e as colunas representam a saída desejada, desta forma, a diagonal principal representa os acertos do sistema.

Utilizando uma Rede Neural Artificial a taxa de acertos foi de 87,63% com os mesmos dados.

Tabela 7.3 - Matriz de confusão (linhas representam a saída calculada e colunas representam a saída desejada).

Classes	**1**	**2**	**3**	**4**
1	**143**	7	1	3
2	6	**87**	4	1
3	0	3	**14**	2
4	0	0	1	**3**

Nos dois casos anteriores, a saída foi modelada considerando se a trajetória vai permanecer na mesma região ou quantas voltas vai permanecer na nova região, caso mude. Porém, pode ser interessante saber também quantas voltas vai permanecer na região atual antes de mudar. Para esta análise, trabalhamos com 6 classes de saída, considerando também, caso não mude a região, quantas voltar vai permanecer na região atual:

- Classe 1: A trajetória se mantém na região atual por mais de 6 voltas.
- Classe 2: A trajetória se mantém na região atual de 4 a 6 voltas.
- Classe 3: A trajetória se mantém na região atual de 1 a 3 voltas.
- Classe 4: A trajetória mudará de região e se manterá na nova região de 1 a 3 voltas.
- Classe 5: A trajetória mudará de região e se manterá na nova região de 4 a 6 voltas.
- Classe 6: A trajetória mudará de região e se manterá na nova região por mais de 6 voltas.

Utilizando as 275 amostras de testes, chegamos a 80,36% de acertos, utilizando o ANFIS. A matriz de confusão da Tabela 7.4 mostra esse resultado. Nesta matriz

as linhas representam a saída calculada e as colunas representam a saída desejada. Desta forma, a diagonal principal representa os acertos do sistema.

Utilizando uma Rede Neural Artificial a taxa de acertos foi de 75,63% com os mesmos dados. Neste exemplo, o classificador neuro-difuso foi claramente superior.

Tabela 7.4 - Matriz de confusão (linhas representam a saída calculada e colunas representam a saída desejada).

Classes	**1**	**2**	**3**	**4**	**5**	**6**
1	**1**	6	0	0	0	1
2	1	**11**	10	0	0	0
3	2	6	**107**	4	0	4
4	1	0	5	**87**	4	2
5	1	0	1	2	**12**	3
6	0	0	0	0	1	**3**

Analisando as Tabelas de 7.1 a 7.4 é possível notar que a maior parte dos padrões que foram classificados de maneira errada estão próximos à diagonal principal (que seriam classificados como certos), poucos casos tiveram erros mais grosseiros, por exemplo, a saída desejada era Classe 1 e a saída calculada foi Classe 5. Isso, de certa forma mostra o potencial do modelo. Além disso, nos resultados apresentados nas Tabelas 7.1, 7.2 e 7.3 as taxas de acerto foram superiores a 80%, e somente na Tabela 7.4 tivemos uma taxa de acerto abaixo dos 80%. Comparando os resultados obtidos através o ANFIS com outro método de reconhecimento de padrões, no nosso caso Redes Neurais artificiais, o ANFIS apresentou resultados um pouco melhores, porém não muito discrepantes.

7.2 Modelo de três-ondas

No modelo três-ondas, conforme mencionado, há também 2 regimes distintos: um deles representado pela linha curva e um segundo regime, que é a linha reta que divide a trajetória curva (ver a reta na Figura 2.2). Análise de previsibilidade é diferente do caso anterior e está relacionada a um intervalo onde a trajetória vai emergir de um regime para o outro. De acordo com o local onde a atual trajetória se transfere para outro regime, é possível inferir o tipo de trajetória na próxima volta. Nos experimentos do modelo três-ondas, foi utilizada uma amostra com 170 padrões (voltas). A amostra foi dividida em três partes: treinamento (100 padrões), validação (30 padrões) e teste (40 padrões). Assim como no modelo de Lorenz, cada padrão

possui 4 entradas, sendo cada entrada o número de estrelas de uma determinada cor em cada volta. A saída desejada é definida de acordo com o número de pontos que a trajetória atual vai permanecer na reta (Figura 2.2). Os testes com o modelo três-ondas também foram realizados com o ANFIS utilizando o software MATLAB versão R2010a. Para fins de comparação, foi utilizado outro método de reconhecimento de padrões, as Redes Neurais artificiais, neste caso, foi utilizado o simulador WEKA.

No primeiro experimento foram definidas apenas duas classes:

- Classe 0: Na próxima volta, a trajetória sobre o regime reta vai durar até 1200 passos de tempo.
- Classe 1: Na próxima volta, a trajetória na reta vai durar mais de 1200 passos de tempo.

Utilizando as 40 amostras de testes, chegamos a 87,5% de acertos, a seguinte matriz de confusão (Tabela 7.5) mostra esse resultado. Nesta matriz, as linhas representam a saída calculada e as colunas representam a saída desejada. Desta forma, a diagonal principal representa os acertos do sistema.

Utilizando uma Rede Neural Artificial a taxa de acertos foi de 92,5% com os mesmos dados.

Tabela 7.5 - Matriz de confusão (linhas representam a saída calculada e colunas representam a saída desejada).

Classes	**0**	**1**
0	**15**	2
1	3	**20**

No experimento seguinte, foram definidas cinco classes:

- Classe 0: Na próxima volta a trajetória na reta vai durar até 1200 passos de tempo.
- Classe 1: Na próxima volta a trajetória na reta vai durar de 1201 até 1600 passos de tempo.
- Classe 2: Na próxima volta a trajetória na reta vai durar de 1601 até 2000 passos de tempo.

- Classe 3: Na próxima volta a trajetória na reta vai durar de 2001 até 2400 passos de tempo.

- Classe 4: Na próxima volta a trajetória na reta vai durar mais de 2400 passos de tempo.

Utilizando as 40 amostras de testes, chegamos a 77,5% de acertos, a seguinte matriz de confusão (Tabela 7.6) mostra esse resultado. Nesta matriz as linhas representam a saída calculada e as colunas representam a saída desejada. Desta forma, a diagonal principal representa os acertos do sistema. Utilizando uma RNA a taxa de acertos foi de 75% com os mesmos dados.

Tabela 7.6 - Matriz de confusão (linhas representam a saída calculada e colunas representam a saída desejada).

Classes	**0**	**1**	**2**	**3**	**4**
0	**16**	0	0	0	1
1	2	**2**	0	0	0
2	0	1	**3**	1	0
3	0	0	0	**4**	1
4	0	0	0	3	**6**

O próximo experimento consistiu em criar um número maior de classes e depois agrupar os resultados em duas ou cinco classes para comparar com os resultados anteriores, neste caso, foram criadas quinze classes:

- Classe 0: Na próxima volta a trajetória na reta vai durar de 0 até 200 passos de tempo.

- Classe 1: Na próxima volta a trajetória na reta vai durar de 201 até 400 passos de tempo.

- Classe 2: Na próxima volta a trajetória na reta vai durar de 401 até 600 passos de tempo.

- Classe 3: Na próxima volta a trajetória na reta vai durar de 601 até 800 passos de tempo.

- Classe 4: Na próxima volta a trajetória na reta vai durar de 801 até 1000 passos de tempo.

- Classe 5: Na próxima volta a trajetória na reta vai durar de 1001 até 1200 passos de tempo.
- Classe 6: Na próxima volta a trajetória na reta vai durar de 1201 até 1400 passos de tempo.
- Classe 7: Na próxima volta a trajetória na reta vai durar de 1401 até 1600 passos de tempo.
- Classe 8: Na próxima volta a trajetória na reta vai durar de 1601 até 1800 passos de tempo.
- Classe 9: Na próxima volta a trajetória na reta vai durar de 1801 até 2000 passos de tempo.
- Classe 10: Na próxima volta a trajetória na reta vai durar de 2001 até 2200 passos de tempo.
- Classe 11: Na próxima volta a trajetória na reta vai durar de 2201 até 2400 passos de tempo.
- Classe 12: Na próxima volta a trajetória na reta vai durar de 2401 até 2600 passos de tempo.
- Classe 13: Na próxima volta a trajetória na reta vai durar de 2601 até 2800 passos de tempo.
- Classe 14: Na próxima volta a trajetória na reta vai durar mais de 2800 passos de tempo.

A Tabela 7.7 mostra os resultados desse experimento considerando todas as quinze classes.

A tabela 7.7 ilustra os resultados com 15 classes usando o ANFIS, neste caso, a taxa de acertos foi de 51,21%. Utilizando a RNA a taxa de acertos foi de 50% com os mesmos dados. Porém, ao agruparmos as classes de 0 a 5 como uma classe, e as classe de 6 a 14 como outra classe, temos um experimento similar ao primeiro feito com duas classes, a Tabela 7.8 ilustra este resultado agrupado.

Através da Tabela 7.8 é possível notar que a taxa de acertos foi de 92% após o agrupamento das classes, com as RNAs a taxa de acertos foi de 95% após o agrupamento. De maneira similar, agrupamos as classes da Tabela 7.7 em 5 classes:

Tabela 7.7 - Matriz de confusão (linhas representam a saída calculada e colunas representam a saída desejada).

Classes	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**
0	**4**	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	2	**2**	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	**3**	0	0	0	0	0	0	0	0	0	1	0	0
3	0	0	0	**0**	1	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	**0**	1	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	**0**	2	1	0	0	0	0	0	0	0
6	0	0	0	0	0	0	**2**	1	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	**1**	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	1	**0**	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	**3**	1	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	**3**	1	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	**0**	1	0	0
12	0	0	0	0	0	0	0	0	0	0	1	2	**3**	0	0
13	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0
14	0	0	0	0	0	0	0	0	0	0	0	1	1	1	**0**

Tabela 7.8 - Matriz de confusão (linhas representam a saída calculada e colunas representam a saída desejada).

Classes	**0**	**1**
0	**17**	3
1	0	**20**

As classes de 0 a 5 se tornaram classe 0.

As classes 6 e 7 se tornaram classe 1.

As classes 8 e 9 se tornaram classe 2.

As classes 10 e 11 se tornaram classe 3.

As classes de 12 a 14 se tornaram classe 4.

Com este agrupamento temos um experimento similar ao segundo experimento com cinco classes, a Tabela 7.9 ilustra este resultado.

Após o agrupamento em cinco classes, a taxa de acertos foi de 80% com o ANFIS (Tabela 7.9) e 80% com a rede neural. Nos dois casos analisados, os resultados após o agrupamento foram melhores do que os resultados com as classes (duas ou

Tabela 7.9 - Matriz de confusão (linhas representam a saída calculada e colunas representam a saída desejada).

Classes	**0**	**1**	**2**	**3**	**4**
0	**13**	3	0	0	0
1	0	**4**	0	0	1
2	0	1	**3**	1	0
3	0	0	0	**6**	1
4	0	0	0	1	**6**

cinco) pré-definidas. Apesar da dificuldade na análise com um grande número de classes (51,21% de acertos com 15 classes), a estratégia de aumentar a granularidade (número de classes) para então colapsar em um número menor de classes indica que o sistema é bastante eficiente. Os resultados acima foram obtidos utilizando o ANFIS, porém com a utilização de RNAs os resultados foram similares.

7.3 Geração Automatizada de Regras Interpretáveis

Apesar dos bons resultados em termos de acurácia, nos experimentos anteriores não foram geradas regras interpretáveis. No caso das Redes Neurais Artificiais, o conhecimento do sistema está nos pesos que são ajustados após o treinamento, porém esse conhecimento é difícil de interpretar, fazendo com que a Rede Neural funcione como uma "Caixa Preta", os dados são apresentados e a rede os classifica sem que o usuário saiba exatamente como esses dados foram classificados. No caso do ANFIS, são geradas regras, porém essas não são facilmente interpretáveis, pois o ANFIS gera um modelo difuso do tipo Takagi-Sugeno, o qual também funciona como um classificador, mas com regras de difícil interpretação pelo usuário.

Para gerar um classificador com regras interpretáveis, utilizamos o Software GUAJE. O GUAJE permite criar um sistema fuzzy do tipo Mamdani, que trabalha com regras mais facilmente interpretáveis, essas regras são geradas automaticamente com o GUAJE através de seus algoritmos.

Em todos os testes com o GUAJE, foram utilizados três conjuntos difusos para cada entrada, nomeados como Baixo (*Low*), Médio (*Average*) e Alto (*High*), de acordo com o valor da entrada. A Figura 7.1 ilustra um exemplo destes conjuntos difusos para a entrada "número de estrelas vermelhas". Também foi utilizada a configuração default do GUAJE para a geração das regras em todos os testes. Os experimentos com o GUAJE foram utilizados com os mesmos dados dos demais experimentos e também foram criadas as mesmas classes para fins de comparação.

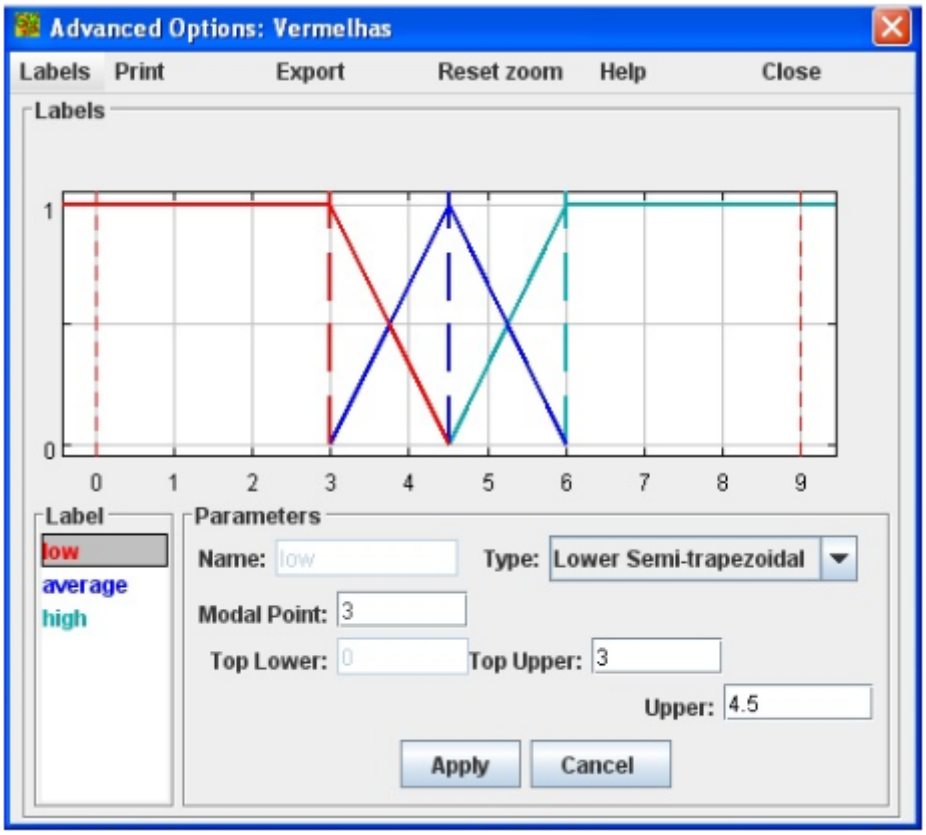

Figura 7.1 - Exemplo dos três conjuntos difusos da entrada número de estrelas vermelhas

7.4 Resultados do GUAJE para o modelo de Lorenz

Utilizando os mesmo dados e classes discutidos na Seção 7.1, em princípio, trabalhamos com as seguintes classes de saída:

- Classe 0: A trajetória se manterá na atual região.
- Classe 1: A trajetória mudará de região.

Considerando as duas classes acima o GUAJE gerou as 16 regras de acordo com a Figura 7.2.

Aplicando as regras acima em um sistema difuso do tipo Mamdani e utilizando as 275 amostras de teste, chegamos a 85,45% de acertos. A matriz de confusão da Tabela 7.10 mostra esse resultado.

Em seguida, utilizamos as seguintes classes de saída:

Rules

Rule	Type	Active	If Vermelhas	AND Amarelas	AND Azuis	AND Verdes	THEN Saida
1	I	yes	low		low	low	1.0
2	I	yes	low		average	low	0.0
3	I	yes	average	low	low	low	1.0
4	I	yes	average	average	low	low	0.0
5	I	yes	average		average	low	0.0
6	I	yes	high			low	0.0
7	I	yes	low			average	1.0
8	I	yes	average			average	0.0
9	I	yes				high	1.0
10	I	yes	low			low	1.0
11	I	yes	average	low	low	low	0.0
12	I	yes	average	low	average	low	0.0
13	I	yes	average	average		low	0.0
14	I	yes	high			low	0.0
15	I	yes				average	1.0
16	I	yes				high	1.0

Figura 7.2 - Regras Geradas pelo GUAJE

Tabela 7.10 - Matriz de confusão (linhas representam a saída calculada e colunas representam a saída desejada).

Classes	**0**	**1**
0	**153**	1
1	39	**82**

- Classe 0: A trajetória se manterá na atual região.
- Classe x: A trajetória mudará de região e se manterá na nova região por x voltas.

Considerando as classes acima o GUAJE gerou as 28 regras de acordo com a Figura 7.3.

Aplicando as regras acima em um sistema difuso do tipo Mamdani e utilizando as 275 amostras de teste, chegamos a 60,36% de acertos, a seguinte matriz de confusão (Tabela 7.11) mostra esse resultado.

No próximo experimento, utilizamos as seguintes classes de saída:

- Classe 0: A trajetória se manterá na atual região.
- Classe 1: A trajetória mudará de região e se manterá na nova região de 1 a 3 voltas.

Rules

Rule	Type	Active	If Vermelhas	AND Amarelas	AND Verdes	AND Azuis	THEN Saida
1	I	yes	average	low	low	low	Classe 0
2	I	yes	average	average	low	low	Classe 0
3	I	yes	average	low	average	low	Classe 0
4	I	yes	average	average	average	low	Classe 0
5	I	yes	average	low	low	average	Classe 2
6	I	yes	high	low	low	low	Classe 0
7	I	yes	high	average	low	low	Classe 0
8	I	yes	low	low	low	average	Classe 4
9	I	yes	average	low	average	average	Classe 3
10	I	yes	high	low	average	low	Classe 0
11	I	yes	high	average	average	low	Classe 0
12	I	yes	average	average	low	average	Classe 1
13	I	yes	low	low	low	low	Classe 2
14	I	yes	average	average	average	average	Classe 1
15	I	yes	low	low	average	average	Classe 4
16	I	yes	low	low	average	low	Classe 1
17	I	yes	low	average	low	low	Classe 1
18	I	yes	low	average	low	average	Classe 2
19	I	yes	low	average	average	low	Classe 1
20	I	yes	low	average	average	average	Classe 2
21	I	yes	average	low	low	high	Classe 7
22	I	yes	low	low	low	high	Classe 8
23	I	yes	high	average	low	average	Classe 0
24	I	yes	high	average	average	average	Classe 0
25	I	yes	high	low	low	average	Classe 0
26	I	yes	high	low	average	average	Classe 0
27	I	yes	low	low	average	high	Classe 6
28	I	yes	average	low	average	high	Classe 6

Figura 7.3 - Regras Geradas pelo GUAJE

- Classe 2: A trajetória mudará de região e se manterá na nova região de 4 a 6 voltas.

- Classe 3: A trajetória mudará de região e se manterá na nova região mais de 6 voltas.

Considerando as classes acima o GUAJE gerou as 25 regras de acordo com a Figura 7.4.

Aplicando as regras acima em um sistema difuso do tipo Mamdani e utilizando as 275 amostras de teste, chegamos a 77,09% de acertos. A seguinte matriz de confusão (Tabela 7.12) mostra esse resultado.

No próximo teste, utilizamos as seguintes classes de saída:

- Classe 0: A trajetória se mantém na região atual por mais de 2 voltas.

- Classe 1: A trajetória se mantém na região atual por exatamente 2 voltas.

Tabela 7.11 - Matriz de confusão (linhas representam a saída calculada e colunas representam a saída desejada).

Classes	0	1	2	3	4	5	6	7	8	9
0	**96**	42	3	0	0	0	0	0	0	0
1	19	**26**	16	0	0	0	0	0	0	0
2	0	0	**39**	0	0	0	0	0	0	0
3	0	0	7	**2**	2	0	0	0	0	0
4	0	0	6	3	**2**	0	0	0	0	0
5	0	0	2	0	4	**0**	0	0	0	0
6	0	0	0	1	1	0	**0**	0	0	0
7	0	0	0	0	0	1	0	**1**	0	0
8	0	0	0	0	0	0	0	0	**0**	0
9	0	0	0	0	0	1	0	1	0	**0**

Rules

Rule	Type	Active	If Vermelhas	AND Amarelas	AND Verdes	AND Azuis	THEN Saida
1	I	yes	high	low	low	low	1.0
2	I	yes	high	low	low	high	3.0
3	I	yes	high	high	low	low	1.0
4	I	yes	high	high	low	high	2.0
5	I	yes	high	low	high	low	0.0
6	I	yes	high	high	high	high	1.0
7	I	yes	high	low	high	high	3.0
8	I	yes	high	high	high	low	0.0
9	I	yes	low	low	low	low	1.0
10	I	yes	low	low	average	low	0.0
11	I	yes	low	low	high	low	0.0
12	I	yes	low	average		low	0.0
13	I	yes	low	high		low	0.0
14	I	yes	low			average	1.0
15	I	yes	low			high	2.0
16	I	yes	average	low	low	low	0.0
17	I	yes	average	low	low	average	0.0
18	I	yes	average	low	average		0.0
19	I	yes	average	low	high		0.0
20	I	yes	average	average	low		0.0
21	I	yes	average	average	average		0.0
22	I	yes	average	high			0.0
23	I	yes	high	low			0.0
24	I	yes	high	average			0.0
25	I	yes	high	high			0.0

Figura 7.4 - Regras Geradas pelo GUAJE

- Classe 2: A trajetória se mantém na região atual por exatamente 1 volta.
- Classe 3: A trajetória mudará de região e se manterá na nova região por exatamente 1 volta.
- Classe 4: A trajetória mudará de região e se manterá na nova região por

Tabela 7.12 - Matriz de confusão (linhas representam a saída calculada e colunas representam a saída desejada).

Classes	**0**	**1**	**2**	**3**
0	**153**	1	0	0
1	47	**49**	2	0
2	0	9	**10**	0
3	0	0	4	**0**

exatamente 2 voltas.

- Classe 5: A trajetória mudará de região e se manterá na nova região por mais de 2 voltas.

Considerando as classes acima o GUAJE gerou as 29 regras de acordo com a Figura 7.5.

Rules

Rule	Type	Active	If Vermelhas	AND Amarelas	AND Verdes	AND Azuis	THEN Saida
1	I	yes	average	low	low	average	3.0
2	I	yes	average	average	average	low	3.0
3	I	yes	low	average	low	low	3.0
4	I	yes	low	low	average	low	3.0
5	I	yes	average	average	low	low	3.0
6	I	yes	high	average	low	low	3.0
7	I	yes	low	low	low	low	4.0
8	I	yes	low	low	low	average	4.0
9	I	yes	average	low	average	low	4.0
10	I	yes	low	low	low	high	6.0
11	I	yes	average	low	low	low	2.0
12	I	yes	high	low	low	low	2.0
13	I	yes	low	low	low	low	3.0
14	I	yes	low	average	low	low	2.0
15	I	yes	low	high	low	low	2.0
16	I	yes	low		average	low	3.0
17	I	yes	low		high	low	3.0
18	I	yes	average	low	low	low	3.0
19	I	yes	average	average	low	low	4.0
20	I	yes	average	high	low	low	4.0
21	I	yes	average		average	low	3.0
22	I	yes	average		high	low	3.0
23	I	yes	high	low	low	low	3.0
24	I	yes	high	low	average	low	3.0
25	I	yes	high	low	high	low	3.0
26	I	yes	high	average		low	4.0
27	I	yes	high	high		low	4.0
28	I	yes				average	5.0
29	I	yes				high	5.0

Figura 7.5 - Regras Geradas pelo GUAJE

Aplicando as regras acima em um sistema difuso do tipo Mamdani e utilizando as 275 amostras de teste, chegamos a 65,81% de acertos. A matriz de confusão da Tabela 7.13 mostra esse resultado.

Tabela 7.13 - Matriz de confusão (linhas representam a saída calculada e colunas representam a saída desejada).

Classes	**1**	**2**	**3**	**4**	**5**	**6**
1	**0**	0	8	0	0	0
2	0	**0**	22	0	0	0
3	0	0	**121**	2	0	0
4	0	0	47	**50**	2	0
5	0	0	0	9	**10**	0
6	0	0	0	0	4	**0**

A Tabela 7.14 apresenta os resultados obtidos para o sistema de Lorenz utilizando as três ferramentas ANFIS, RNAs e GUAJE. Os valores na Tabela7.14 representam a porcentagem de acertos utilizando o conjunto de testes em cada um dos experimentos realizados com o sistema de Lorenz.

Tabela 7.14 - Resultados obtidos para o sistema de Lorenz utilizando ANFIS, RNAs e GUAJE.

Acurácia	**2 Classes**	**10 Classes**	**4 Classes**	**6 Classes**
ANFIS	91,63%	81,45%	89,81%	80,36%
RNA	90,90%	80%	87,63%	75,63%
GUAJE	85,45%	60,36%	77,09%	65,81%

Analisando a Tabela 7.14, é possível notar que o GUAJE apresentou resultados um pouco inferiores em relação ao ANFIS e à rede neural, porém o GUAJE apresenta a vantagem de gerar regras mais facilmente interpretáveis de maneira automática, ou seja, com o GUAJE perdemos um pouco em acurácia e ganhamos em interpretabilidade.

7.5 Resultados do GUAJE para o modelo de três ondas

Utilizando os mesmo dados e classes discutidos na Seção 7.2, em princípio, trabalhamos com as seguintes classes de saída:

- Classe 0: Na próxima volta, a trajetória na reta vai durar até 1200 passos de tempo.
- Classe 1: Na próxima volta, a trajetória na reta vai durar mais de 1200 passos de tempo.

Considerando as duas classes acima o GUAJE gerou as 14 regras de acordo com a Figura 7.6.

Rules

Rule	Type	Active	If Vermelhas	AND Amarelas	AND Verdes	AND Azuis	THEN Saida
1	I	yes	low				1.0
2	I	yes	average	low	low	low	1.0
3	I	yes	average	low	low	average	1.0
4	I	yes	average	low	low	high	0.0
5	I	yes	average	average	low	low	1.0
6	I	yes	average	average	low	average	1.0
7	I	yes	average	average	low	high	0.0
8	I	yes	average	high	low		0.0
9	I	yes	average	low	average	low	1.0
10	I	yes	average	average	average	low	1.0
11	I	yes	average		average	average	1.0
12	I	yes	average		high		1.0
13	I	yes	high		low		0.0
14	I	yes	high		average		0.0

Figura 7.6 - Regras Geradas pelo GUAJE

Aplicando as regras acima em um sistema difuso do tipo Mamdani e utilizando as 40 amostras de teste, chegamos a 87,5% de acertos. A seguinte matriz de confusão (Tabela 7.15) mostra esse resultado.

Tabela 7.15 - Matriz de confusão (linhas representam a saída calculada e colunas representam a saída desejada).

Classes	**0**	**1**
0	**12**	5
1	0	**23**

Em seguida, utilizamos as seguintes cinco classes de saída:

- Classe 0: Na próxima volta a trajetória na reta vai durar até 1200 passos de tempo.

- Classe 1: Na próxima volta a trajetória na reta vai durar de 1201 até 1600 passos de tempo.

- Classe 2: Na próxima volta a trajetória na reta vai durar de 1601 até 2000 passos de tempo.

- Classe 3: Na próxima volta a trajetória na reta vai durar de 2001 até 2400 passos de tempo.

- Classe 4: Na próxima volta a trajetória na reta vai durar mais de 2400 passos de tempo.

Considerando as duas classes acima o GUAJE gerou as 12 regras de acordo com a Figura 7.7.

Rules

Rule	Type	Active	If Vermelhas	AND Amarelas	AND Verdes	AND Azuis	THEN Saida
1	I	yes	low	average	average	high	0.0
2	I	yes	average	average	low	low	0.0
3	I	yes	average	average	average	low	2.0
4	I	yes	average	average	high	low	4.0
5	I	yes	average	high	low	low	0.0
6	I	yes	low	average	high	low	4.0
7	I	yes	low	low	average	low	3.0
8	I	yes	low	high	low	low	0.0
9	I	yes	high	high	low	low	0.0
10	I	yes	low	low	high	low	4.0
11	I	yes	low	average	low	low	1.0
12	I	yes	low	average	average	low	2.0

Figura 7.7 - Regras geradas pelo GUAJE

Aplicando as regras acima em um sistema difuso do tipo Mamdani e utilizando as 40 amostras de teste, chegamos a 70% de acertos, conforme resultados apresentados na matriz de confusão (Tabela 7.16).

Similar à Seção 7.2,o próximo experimento consistiu em criar um número maior de classes e depois agrupar os resultados em duas ou cinco classes para comparar com os resultados anteriores, neste caso, também foram criadas quinze classes:

- Classe 0: Na próxima volta a trajetória na reta vai durar de 0 até 200 passos de tempo.

Tabela 7.16 - Matriz de confusão (linhas representam a saída calculada e colunas representam a saída desejada).

Classes	**0**	**1**	**2**	**3**	**4**
0	**13**	3	0	0	1
1	0	**4**	0	0	0
2	0	1	**1**	3	0
3	0	0	0	**4**	1
4	0	0	0	3	**6**

- Classe 1: Na próxima volta a trajetória na reta vai durar de 201 até 400 passos de tempo.
- Classe 2: Na próxima volta a trajetória na reta vai durar de 401 até 600 passos de tempo.
- Classe 3: Na próxima volta a trajetória na reta vai durar de 601 até 800 passos de tempo.
- Classe 4: Na próxima volta a trajetória na reta vai durar de 801 até 1000 passos de tempo.
- Classe 5: Na próxima volta a trajetória na reta vai durar de 1001 até 1200 passos de tempo.
- Classe 6: Na próxima volta a trajetória na reta vai durar de 1201 até 1400 passos de tempo.
- Classe 7: Na próxima volta a trajetória na reta vai durar de 1401 até 1600 passos de tempo.
- Classe 8: Na próxima volta a trajetória na reta vai durar de 1601 até 1800 passos de tempo.
- Classe 9: Na próxima volta a trajetória na reta vai durar de 1801 até 2000 passos de tempo.
- Classe 10: Na próxima volta a trajetória na reta vai durar de 2001 até 2200 passos de tempo.
- Classe 11: Na próxima volta a trajetória na reta vai durar de 2201 até 2400 passos de tempo.

- Classe 12: Na próxima volta a trajetória na reta vai durar de 2401 até 2600 passos de tempo.
- Classe 13: Na próxima volta a trajetória na reta vai durar de 2601 até 2800 passos de tempo.
- Classe 14: Na próxima volta a trajetória na reta vai durar mais de 2800 passos de tempo.

Considerando as classes acima o GUAJE gerou as 28 regras de acordo com a Figura 7.8.

Rules

Rule	Type	Active	If Vermelhas	AND Amarelas	AND Verdes	AND Azuis	THEN Saida
1	1	yes	average	average	low	low	Classe 3
2	1	yes	low	low	high	low	Classe 12
3	1	yes	average	average	average	low	Classe 7
4	1	yes	low	average	average	low	Classe 8
5	1	yes	low	average	low	low	Classe 5
6	1	yes	average	high	low	low	Classe 0
7	1	yes	average	average	low	average	Classe 3
8	1	yes	low	low	average	low	Classe 10
9	1	yes	low	high	low	low	Classe 1
10	1	yes	low	average	high	low	Classe 13
11	1	yes	average	average	average	average	Classe 5
12	1	yes	low	average	low	average	Classe 3
13	1	yes	average	high	low	average	Classe 2
14	1	yes	average	low	high	low	Classe 12
15	1	yes	average	high	average	low	Classe 4
16	1	yes	low	high	low	average	Classe 3
17	1	yes	average	average	high	low	Classe 12
18	1	yes	low	high	average	low	Classe 5
19	1	yes	high	high	low	low	Classe 0
20	1	yes	high	average	low	low	Classe 0

Figura 7.8 - Regras Geradas pelo GUAJE

A Tabela 7.17 mostra os resultados desse experimento considerando todas as quinze classes.

A Tabela 7.17 ilustra os resultados com 15 classes usando o ANFIS. Neste caso, a taxa de acertos foi de 50%. Porém, ao agruparmos as classes de 0 a 5 como uma única classe, e as classe de 6 a 14 como outra classe, temos um experimento similar ao primeiro feito com duas classes. A Tabela 7.18 ilustra este resultado agrupado.

Através da Tabela 7.18 é possível notar que a taxa de acertos foi de 90% após o agrupamento das classes. Da mesma foram, os dados da Tabela 7.5 foram agrupados

Tabela 7.17 - Matriz de confusão (linhas representam a saída calculada e colunas representam a saída desejada).

Classes	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**
0	**3**	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	**1**	1	1	0	0	0	0	0	0	0	0	0	0	0
2	0	0	**1**	0	0	0	0	0	0	0	0	1	0	0	0
3	0	0	0	**4**	1	0	0	0	0	0	0	0	0	0	0
4	0	0	0	1	**3**	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	**1**	0	0	0	0	0	0	0	0	0
6	0	0	0	0	2	0	**0**	0	0	0	0	0	0	0	0
7	0	0	0	0	0	1	0	**0**	0	0	0	0	0	0	0
8	0	0	0	0	0	0	1	0	**0**	0	0	0	0	0	0
9	0	0	0	0	0	0	0	1	0	**0**	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	**4**	2	0	0	0
11	0	0	0	0	0	0	0	0	0	0	1	**3**	0	0	0
12	0	0	0	0	0	0	0	0	0	0	2	2	**0**	0	0
13	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0
14	0	0	0	0	0	0	0	0	0	0	0	3	0	0	**0**

Tabela 7.18 - Matriz de confusão (linhas representam a saída calculada e colunas representam a saída desejada).

Classes	**0**	**1**
0	**17**	1
1	3	**19**

em cinco classes, a saber:

As classes de 0 a 5 se tornaram classe 0.

As classes 6 e 7 se tornaram classe 1.

As classes 8 e 9 se tornaram classe 2.

As classes 10 e 11 se tornaram classe 3.

As classes de 12 a 14 se tornaram classe 4.

Com este agrupamento temos, conforme a Tabela 7.19, um experimento similar ao segundo experimento com cinco classes.

Tabela 7.19 - Matriz de confusão (linhas representam a saída calculada e colunas representam a saída desejada).

Classes	**0**	**1**	**2**	**3**	**4**
0	**17**	0	1	0	0
1	3	**1**	0	0	0
2	0	0	**1**	0	0
3	0	0	4	**10**	0
4	0	0	0	0	**3**

Analisando a Tabela 7.19 é possível notar que a taxa de acertos foi de 80%. Nos dois casos analisados os resultados após o agrupamento foram melhores do que os resultados com as classes (duas ou cinco) pré-definidas. Apesar da dificuldade de trabalhar com um grande número de classes (50% de acertos com 15 classes), os resultados indicam que o sistema é bastante eficiente quando a granularidade (número de classes) é aumentada para depois os dados serem agrupados em um número menor de classes.

A Tabela 7.20 apresenta os resultados obtidos para o modelo três ondas utilizando as três ferramentas ANFIS, RNAs e GUAJE. Os valores na Tabela 7.20 representam a porcentagem de acertos utilizando o conjunto de testes em cada um dos experimentos realizados com o modelo três ondas.

Tabela 7.20 - Resultados obtidos para o modelo três ondas utilizando ANFIS, RNAs e GUAJE

Acurácia	**2 Classes (original)**	**5 Classes (original)**
ANFIS	87%	77,5%
RNA	92,5%	75%
GUAJE	87,5%	70%
Acurácia	**2 Classes (agrupado**	**5 Classes (agrupado)**
ANFIS	92%	80%
RNA	95%	80%
GUAJE	90%	80%

De maneira similar aos resultados do sistema de Lorenz, é possível notar que o GUAJE, no geral, apresentou resultados um pouco inferiores em relação ao ANFIS e à RNA (Tabela 7.20), porém o GUAJE apresenta a vantagem de gerar regras mais facilmente interpretáveis de maneira automática. Ou seja, como antes, com o GUAJE perdemos um pouco em acurácia e ganhamos em interpretabilidade.

8 CONCLUSÃO

O principal objetivo deste trabalho trata do desenvolvimento de técnicas de previsibilidade de sistemas dinâmicos. A maneira mais tradicional usada em centros operacionais de previsão de tempo é o uso da **previsão por conjunto** (ou *ensemble*)[1]: uma vez determinado um conjunto de trajetórias admissíveis é possível estimar zonas de alta previsibilidade (convergência de trajetórias) e regiões de baixa previsibilidade (local onde as trajetórias divergem uma das outras). A previsão por *ensemble* é uma técnica que permite determinar algumas propriedades estatísticas: a média do *ensemble* (esta é a informação de previsão que é divulgada para o público), o desvio padrão (o quanto as trajetórias estão agrupadas em torno da média), além do intervalo de confiança de previsão.

Os sistemas dinâmicos não lineares podem comportar-se de maneira caótica. É bem conhecida a dificuldade de se realizar uma previsão precisa sob uma dinâmica caótica, devida a alta sensibilidade destes sistemas às condições iniciais.

Diferentemente da técnica de *ensemble*, no presente estudo, a previsibilidade é formulada como um problema de classificação. Além de identificação de diferentes classes de dinâmica, tem-se como objetivo a criação de regras que possam simplificar a interpretabilidade para o sistema que estima o tipo de dinâmica (a "previsibilidade"). Os sistemas neuro-difusos cumprem a tarefa de ser classificadores com a capacidade de geração de regras.

Dois sistemas dinâmicos que apresentam comportamento caótico foram usados como exemplos neste estudo: o sistema de *Lorenz* (vinculado a dinâmica da atmosfera) e o Modelo Três Ondas (associado à física solar). A primeira etapa, modelar o problema de previsibilidade em um problema de classificação, foi realizada através da utilização do conceito de *"bred vectors"*, gerando pares de entrada-saída desejada de acordo com os valores dos *"bred vectors"*.

As técnicas utilizadas como classificadores foram dois sistemas neuro-difusos: do tipo Takagi-Sugeno (MATLAB/ANFIS) e do tipo Mamdani (GUAJE). Para avaliar a capacidade de estimação de classes dos sistemas neuro-difusos e corroborar os resultados, foi utilizada uma Rede Neural Artificial, do tipo MLP - *Multi-Layer Perceptron* (WEKA).

Os resultados foram muito promissores. No caso do Sistema de Lorenz, a acurácia

[1] Ver artigo na wikipedia: $http://en.wikipedia.org/wiki/Ensemble_forecasting$

ficou em torno de 80% a 90% de acertos com o ANFIS e com a RNA como classificadores, no caso do Modelo Três Ondas, a acurácia ficou em torno de 75% a 90% de acertos com estes classificadores (SANTOS et al., 2012a). Em ambos os sistemas dinâmicos, com a utilização do GUAJE, a acurácia ficou, em geral, um pouco menor. Porém, o GUAJE traz a vantagem de gerar regras automáticas mais facilmente interpretáveis. Deste modo, perdemos um pouco em acurácia e ganhamos bastante em previsibilidade (SANTOS et al., 2012b). Como um dos objetivos era a geração de regras automáticas, as regras utilizadas foram aquelas geradas com o GUAJE, porém é possível realizar modificações manualmente nestas regras.

a) É possível formular um problema de previsibilidade de sistemas dinâmicos em um problema de classificação.

b) Técnicas de inteligência artificial mostraram-se promissoras.

c) Sistemas *Neuro-difuso* mostraram-se bons classificadores e capazes de gerar regras automáticas de fácil interpretabilidade.

d) É possível realizar ajustes manuais nas regras automáticas, melhorando a previsão e mantendo a interpretabilidade.

Para sistemas dinâmicos de grande porte, como os modelos numéricos de previsão de tempo e clima, a quantidade de informações pode tornar proibitivo o uso de classificadores, ou gerar sistemas pouco precisos. A dimensionalidade do problema pode ser uma questão relevante e técnicas de redução de dimensão dos dados podem ser aplicadas, como conjuntos aproximativos (ANOCHI, 2010), *p-value* (RUIVO, 2013) e *wavelets* (KISI; SHIRI, 2011) e (OZGER et al., 2012).

REFERÊNCIAS BIBLIOGRÁFICAS

ALBUQUERQUE, A. C.; NETO, A. D.; MELO, J. D. Algoritmos genéticos e processamento paralelo aplicado ao treinamento e definição de redes neurais perceptrons de múltiplas camadas. In: XV CONGRESSO BRASILEIRO DE AUTOMÁTICA, Gramado, RS. **Anais...** Gramado: SBA, 2004. 18

ALONSO, J. M. Analyzing interpretability of fuzzy rule-based systems by means of fuzzy inference-grams. In: 1ST WORLD CONFERENCE ON SOFT COMPUTING, San Francisco, CA. **Proceedings...** San Francisco State University, 2011. p. 181.1–181.8. 30

ALONSO, J. M.; MAGDALENA, L. Guaje - A java environment for generating understandable and accurate models. In: XV SPANISH CONFERENCE FOR FUZZY LOGIC AND TECHNOLOGY, Huelva, Spain. **Proceedings...** Huelva: ESTYLF, 2010. p. 399–404. 30

_____. Generating understandable and accurate fuzzy rule-based systems in a java environment. In: LECTURE NOTES IN ARTIFICIAL INTELLIGENCE - 9TH INTERNATIONAL WORKSHOP ON FUZZY LOGIC AND APPLICATIONS, Trani, Italy. **Proceedings...** Trani: Springer Berlin Heidelberg, 2011. p. 212–219. 31

_____. Hilk++: an interpretability-guided fuzzy modeling methodology for learning readable and comprehensible fuzzy rule-based classifiers. **Soft Computing**, v. 15, n. 10, p. 1959–1980, 2011. 30

ALONSO, J. M.; MAGDALENA, L.; GUILLAUME, S. Hilk: A new methodologyfor designing highly interpretable linguistic knowledge bases usingthe fuzzy logic formalism. **International Journal of Intelligent Systems**, v. 23, n. 7, p. 761–794, 2008. 30

ALONSO, J. M.; PANCHO, D. P.; MAGDALENA, L. Enhancing the fuzzy modeling tool guaje with a new module for fingrams-based analysis of fuzzy rule bases. In: INTERNATIONAL CONFERENCE ON FUZZY SYSTEMS, WORLD CONGRESS ON COMPUTATIONAL INTELLIGENCE (WCCI), Brisbane, Australia. **Proceedings...** Brisbane: IEEE, 2012. p. 1082–1089. 30

ANOCHI, J. **Modelos baseados em redes neurais para o estudo de padrões climáticos sazonais a partir de dados tratados com a teoria dos**

conjuntos aproximativos. Dissertação de Mestrado — Instituto Nacional de Pesquisas Espaciais, São José dos Campos, 2010. 62

ANOCHI, J. A.; SAMBATTI, S. B. M.; LUZ, E. F. P.; VELHO, H. F. C. Previsão climática de precipitação usando rede neural. In: CONGRESSO BRASILEIRO DE METEOROLOGIA (CBMET), Gramado, RS. **Anais...** Gramado: SBMET, 2012. 18

BRAGA, A. P.; CARVALHO, A. C. P. L. F.; LUDERMIR, T. B. **Redes Neurais Artificiais**: Teoria e aplicações. Rio de Janeiro: Livros Técnicos e Científicos (LTC), 2000. 9, 10, 11, 12, 14, 15, 16, 17, 18, 37

CARVALHO, A. R.; CHAVES, A. A.; RAMOS, F. M. Metaheuristics for the feedforward artificial neural network (ANN) architecture optimization problem. **Neural Computing and Applications**, v. 20, p. 1273–1284, 2011. 14

CINTRA, R. S.; VELHO, H. F. C. Predictability fora chaotic solar plasma system. **Symposia:Advanced Modeling on Computational Fluid Dynamics**, 2008. 1, 8, 33, 36

CINTRA, R. S.; VELHO, H. F. C.; TODLING, R. Redes neurais artificiais na melhoria de desempenho de métodos de assimilação de dados: Filtro de kalman. **TEMA. Tendências em Matemática Aplicada e Computacional**, v. 11, p. 29–39, 2010. 5

ECKHARDTA B.; YAOA, D. Local lyapunov exponents in chaotic systems. **Physica D: Nonlinear Phenomena**, v. 65, n. 1-2, p. 100–108, 1993. 1, 8

EVANS, E. Rise undergraduates find that regime changes in lorenz s model are predictable. **Bull. Amer. Meteor. Soc.**, v. 85, p. 521–524, 2004. 1, 8, 33, 36

GODFREY, M. D.; HENDRY, D. F. The computer as von neumann planned it. **IEEE Annals of the History of Computing**, v. 15, n. 1, p. 11–21, 1993. 10

GREBOGI, C. Strange attractors that are not chaotic. **Physica D: Nonlinear Phenomena**, v. 13, p. 261–268, 1984. 3, 4

GUÉGAN, D.; LEROUX, J. **Predicting Chaos with Lyapunov Exponents: Zero Plays no Role in Forecasting Chaotic Systems, Chaotic Systems**. 2011. Disponível em: <http://www.intechopen.com/articles/show/title/rpredicting-chaos-with-lyapunov-exponents-zero-/rplays-no-role-in-forecasting-chaotic-systems>. Acesso em: 04 de março de 2011. 1, 8

HAMILL, T. M.; SNYDER, C. A comparison of probabilistic forecasts from bred, singular-vector, and perturbed observation ensembles. **Monthly Weather Review**, v. 128, p. 1835–1851, 2000. 1, 8

HAYKIN, S. **Neural Networks**: A comprehensive foundation. New Jersey: Prentice Hall, 1999. 12, 29, 37

JACINTHO, L. F. O. **Redes Neuro-Difusos: Um Estudo de Caso em Diagnóstico de Alzheimer**. Trabalho Final de Graduação — Universidade Federal do ABC, Snto André, 2010. 28, 29

JANG, J. S. R. Anfis: Adaptive-network-based fuzzy inference systems. **IEEE Transactions on Systems, Man, and Cybernetics**, v. 23, p. 714–723, 1993. 1, 28, 29

KISI, O.; SHIRI, J. Precipitation forecasting using wavelet-genetic programming and wavelet-neuro-fuzzy conjuction models. **Water Resources Management**, v. 25, p. 3135–3152, 2011. 62

KITANO, H. Empirical studies on the speed of convergence of neural network training using genetic algorithms. In: VIII NATIONAL CONFERENCE ON ARTIFICIAL INTELLIGENCE, Boston, Massachusetts. **Proceedings...** Boston, 1990. 18

KOVACS, Z. L. **Redes Neurais Artificiais**: Fundamentos e aplicações. São Paulo: Edição Acadêmica, 1996. 9

LORENZ, E. N. Deterministic non-periodic flow. **J. Atmos. Sci.**, v. 20, p. 130–141, 1963. 5

_____. The predictability of a flow which possesses many scales of motion. **Tellus**, v. 17, p. 321–333, 1965. 5

_____. A study of the predictability of a 28-variable atmospheric model. **Tellus**, v. 17, p. 321–333, 1965. 5

MAMDANI, E. H. Application of fuzzy algorithms for control of simple dynamic plant. **Proceedings of the Institution of Electrical Engineers**, v. 121, n. 12, p. 298–316, 1974. 23

MATHWORKS. **R2011b Documentation**. 2011. Disponível em: <http://www.mathworks.com/help/toolbox/fuzzy/fp49243.html>. Acesso em: 05 de março de 2011. 23, 24, 25

MENDONCA, A. M.; BONATTI, J. P. O sistema de previsão de tempo global por ensemble do CPTEC. In: XII CONGRESSO BRASILEIRO DE METEOROLOGIA, Foz do Iguaçu, PR. **Anais...** Foz do Iguaçu: SBMET, 2002. 1, 8

MONTANA, D. J.; DAVIS, L. Training feedforward neural networks using genetic algorithms. **Proceedings of the International Joint Conference on Artificial Intelligence**, p. 762–767, 1989. 18

MONTEIRO, L. H. A. **Sistemas Dinâmicos**. São Paulo: Livraria da Física, 2006. 3

MOTA, T. C. **Aplicação de Algoritmos Genéticos - Definição da Arquitetura e ao Treinamento de Redes Neurais MLP**. Trabalho Final de Graduação — Universidade Federal do Rio de Janeiro, Rio de Janeiro, 2007. 18

NEWMAN, C. E.; READ, P. L.; LEWIS, S. R. Breeding vectors and predictability in the oxford mars gcm. In: FIRST INTERNATIONAL WORKSHOP ON MARS ATMOSPHERE MODELLING AND OBSERVATIONS, Granada, Spain. **Proceedings...** Granada, 2003. p. 13–15. 1, 8, 33

OTT, E. **Chaos in dynamical systems**. Cambridge: Cambridge university press, 2002. 5, 8

OZGER, M.; MISHRA, A. K.; SINGH, V. P. Long lead time drought forecasting using a wavelet and fuzzy logic combination model: A case study in texas. **Journal of Hydrometeor**, v. 13, n. 1, p. 284–297, 2012. 62

PALMER, T. N. Singular vectors, metrics, and adaptive observations. **Journal of the Atmospheric Sciences**, v. 55, p. 633–653, 1998. 1, 8

PASINI, A.; PELINO, V. Can we estimate atmospheric predictability by performance of neural network forecasting? the toy case studies of unforced and forced lorenz models. In: INTERNATIONAL CONFERENCE ON COMPUTATIONAL INTELLIGENCE FOR MEASUREMENT SYSTEMS AND APPLICATIONS (CIMSA 2005), Giardini Naxos, Italy. **Proceedings...** Giardini Naxos: IEEE, 2005. 2

REZENDE, S. O. **Sistemas Inteligentes**: Fundamentos e aplicações. São Paulo: Editora Manole, 2003. 27

RODRIGUES, L. M.; DIMURO, G. P. Utilizando lógica fuzzy para avaliar a qualidade de uma compra via internet. In: WEIT 2011 - WORKSHOP-ESCOLA DE INFORMáTICA TEóRICA, Pelotas, RS. **Anais...** Pelotas, 2011. p. 60–66. 19

RUIVO, H. M. **Metodologias de mineração de dados em análise climática**. Tese de doutorado — Instituto Nacional de Pesquisas Espaciais, São José dos Campos, 2013. 62

SAMBATTI, S. B. M.; ANOCHI, J. A.; LUZ, E. F. P.; SHIGUEMORI, E. H.; CARVALHO, A. R.; VELHO, H. F. C. Automatic configuration for neural network applied to atmospheric temperature profile identification. In: INTERNATIONAL CONFERENCE ON ENGINEERING OPTIMIZATION (ENGOPT), Rio de Janeiro, Brazil. **Proceedings...** Rio de Janeiro, 2012. 14

SANDRI, S. A.; CORREA, C. Lógica nebulosa. In: V ESCOLA DE REDES NEURAIS, São José dos Campos, SP. **Anais...** ITA: Conselho Nacional de Redes Neurais, 1999. p. c73–c90. 19, 20, 21, 27

SANTOS, P. L. B.; SANDRI, S. A.; VELHO, H. F. C. Previsibilidade em sistemas caóticos utilizando o sistema neuro-difuso ANFIS. In: II CONGRESSO BRASILEIRO DE SISTEMAS FUZZY (CBSF-2012), Natal, RN. **Anais...** Natal, 2012. p. 113–126. 62

SANTOS, P. L. B.; VELHO, H. F. C.; SANDRI, S. A. Chaotic systems predictability using neuro-fuzzy systems and neural networks with bred vectors. In: 2ND WORLD CONFERENCE ON SOFT COMPUTING, Baku, Azerbaijan. **Proceedings...** Baku, 2012. p. 1–8. 62

SUGENO, M.; KANG, G. T. Structure identification of fuzzy model. **Fuzzy Sets and Systems**, v. 28, p. 329–346, 1986. 22

TAKAGI, T.; SUGENO, M. Fuzzy identification of systems and its applications to modeling and control. **IEEE Transactions on Systems, Man, and Cybernetics**, v. 15, n. 1, p. 116–132, 1985. 22

TANSCHEIT, R. **Sistemas Fuzzy**. 2013. Disponível em: <http://www2.ica.ele.puc-rio.br/Downloads/41/LN-Sistemas%20Fuzzy.pdf>. Acesso em: 15 de outubro de 2013. 19, 21

TEIXEIRA, M. C.; PIETROBOM, H. C.; aO, E. A. Novos resultados sobre a estabilidade e controle de sistemas não-lineares utilizando modelos fuzzy e lmi. **Controle Automação**, v. 11, n. 1, p. 37–48, 2000. 23

TOTH, Z.; KALNAY, E. Ensemble forecasting at ncep and the breeding method. **Monthly Weather Review**, v. 126, p. 3292–3302, 1997. 1, 8, 33

VITOR, T. S. **Projeto de sistemas de controle com modelagem Takagi-Sugeno e implementação de controladores Fuzzy com retroação de estados**. Dissertação de Mestrado — Universidade Federal de Itajubá, Itajubá, 2011. 23

WEKA, S. oficial do. **WEKA - Waikato Environment for Knowledge Analysis**. 2012. Disponível em: <http://www.cs.waikato.ac.nz/~ml/weka/>. Acesso em: 25 de julho de 2012. 37

ZADEH, L. A. Fuzzy sets. **Information and Control**, v. 8, p. 338–353, 1965. 19

APÊNDICE A - TRABALHO COMPLETO EM CONGRESSO

Chaotic Systems Predictability Using Bred Vectors with Neural Networks and Neuro-Fuzzy Systems

Pettras Leonardo Bueno dos Santos, Haroldo Fraga de Campos Velho, Rosangela Cintra and Sandra Sandri

***Abstract*—The predictability of the behavior of chaotic systems is of great importance because many real-world phenomena have some type of chaotic regime. In chaotic systems, small changes in the initial conditions can lead to very different results from the original system trajectory. The prediction of chaotic systems behavior is usually very difficult, particularly in practical applications in which initial conditions are obtained by measurement instruments, very often subject to acquisition errors. Here we use "bred vectors" methodology to generate pairs of input/output required for training Neural Networks and Neuro-Fuzzy systems. We apply the approach to predict the regime for the strange attractors of Lorenz and the nonlinear coupled three-waves problem from solar physics.**

***Index Terms*—Chaotic systems, bred vectors, Lorenz attractor, three-waves problem, neural networks, neuro-fuzzy systems.**

I. Introduction

THE ability to predict the future state of a system, given its present state, is a non-trivial problem. It is also of capital importance; a correct prediction is the main factor in the prevention (or minimization of impact) of major natural disasters.

Due to their own nature, prediction in the context of chaotic systems is particularly challenging. In chaotic systems, small changes in the initial conditions can lead to very different results from the original trajectory of the system. The problem is aggravated in practical applications in which initial conditions are obtained by measuring instruments, very often subject to errors in precision, as in many applications in geophysical and astronomical sciences. Earth climate, for instance, is a prototypical chaotic system [12] of fundamental importance. Its behavior is hard to predict and is nowadays obtained from the aggregation of an ensemble of results, derived from running of a computational model with different sets of initial conditions.

In recent years, the breeding vector technique has been used with success to predict the behavior of several aspects in various chaotic systems. The breeding method was developed as a method to generate initial perturbations for ensemble forecasting in numerical weather prediction at the National Centers for Environmental Prediction (NCEP) [18], being nowadays a well-established and computationally inexpensive method for generating perturbations for ensemble integrations [4].

The breeding method involves simply running the nonlinear model twice at each time step, one with the original data (control run) and the other with a small perturbation added to it. After a fixed number of time steps, the obtained results are subtracted and the difference is rescaled so that it has the same size as the original perturbation. The rescaled difference, called a bred vector, is added to the control run and the process repeated.

Bred vectors have been used to predict several chaotic systems [5] [14] [4]. In particular, the authors of [5] (respec. [4]) derived rules for predicting behavior of the Lorenz attractor (respec. three-wave problem), upon observation of the bred-vector growth. The accuracy of the prediction using these rules have been extremely good in both experiments, showing that not only bred vectors provide a good tool for the prediction of behavior of chaotic systems but that the knowledge produced to do so is interpretable. On the other hand, observation is a painstakingly method.

In this work, we investigate the use of neuro-fuzzy system ANFIS [16] to evaluate the predictability of chaotic systems. This task is formulate as a classification problem, where classes of dynamics are identified. The ANFIS results are compared to those obtained through the use of a standard neural network [8]. We have performed experiments on Lorenz strange attractor [12] and the nonlinear coupled three-waves model [4] [3]. Neural networks have already been used with bred vectors to predict the behavior of Lorenz attractor [14] but here we explore different types of experiments.

This paper is organized as follows. In Section 2 we briefly present chaotic systems and bred vectors, as well as the two systems used in our applications: the Lorenz attractor and the three-waves problem. In Section 3 we briefly address Neural Networks, Fuzzy Systems and Neural Fuzzy Systems. Sections 4 and 5 respectively bring the experiments and the conclusion.

II. Chaotic Systems and Bred Vectors

Chaotic systems are extremely sensitive to initial conditions: a slight deviation from a trajectory in the state space can lead to dramatic changes in future behavior [7]. The prediction of the future state of a system knowing its initial conditions is a fundamental problem with obvious applications in geophysical flows. The predictability of weather and climate forecasts

Pettras Leonardo Bueno dos Santos, Haroldo Fraga de Campos Velho, Rosangela Cintra and Sandra Sandri are with the Brazilian National Institute for Space Research (INPE), São José dos Campos, SP, 12227-010 Brazil e-mail: pettrasleonardo@yahoo.com.br, haroldo@lac.inpe.br, rosangela.cintra@lac.inpe.br, sandra.sandri@inpe.br

for instance is determined by the projection of uncertainties in both initial conditions and model formulation onto flow-dependent instabilities of the chaotic climate attractor. Since it is essential to be able to estimate the impact of such uncertainties on forecast accuracy, no weather or climate prediction can be considered complete without a forecast of the associated flow-dependent predictability [4].

The breeding method [18] was developed as a method to generate initial perturbations for ensemble forecasting in numerical weather prediction. The method consists in running the nonlinear model twice, one with the original data (control) and the other with a small perturbation added to it. The control solution is subtracted from the perturbed solution after a fixed amount of time steps and the rescaled difference (a bred vector) is then added to the control run and the process repeated[1]. The growth rate of the bred vectors is a measure of the local instability of flow [4].

Bred vectors are a nonlinear generalization of leading Lyapunov exponents, that rates the differences of two initially close trajectories of chaotic dynamical systems. Studies on the stability properties of evolving flows can be found in [2] for Lyapunov vectors and in [10] for bred vectors.

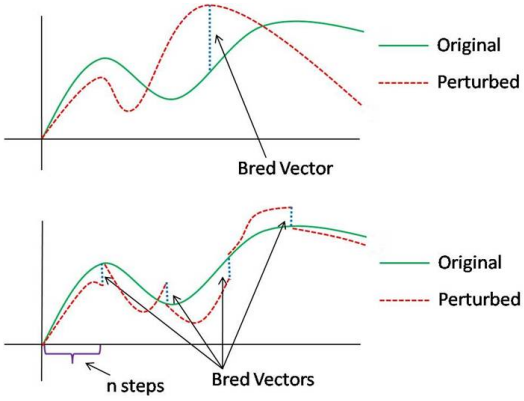

Fig. 1. Bred vectors growth.

The Bred Vectors Algorithm is given by [11]:

1) Initialization step (executed only once).
   Calculate the initial perturbation $A = \delta f(x,t)$, using an arbitrary norm.
2) Add the perturbation calculated in the previous step to the basic solution, integrate the perturbed condition with the nonlinear model for a fixed number of time steps, and subtract the original unperturbed solution from the perturbed nonlinear integration

$$\delta A = \delta f(x, t+\Delta t) = f(x, t+\Delta t) - f(x,t) \quad (1)$$

3) Measure the size $A + \delta A$ of the evolved perturbation $\overline{\delta f(x,t+\Delta t)}$, and divide the perturbation by the measured amplification factor so that its size remains equal to A:

$$\delta f(x, t+\Delta t) = \overline{\delta f(x, t+\Delta t)} \sim A/(A+\delta A) \quad (2)$$

Steps 2 and 3 are repeated for the next time interval and so on.

Bred vectors have been used with success to predict the behavior of chaotic systems such as the Lorenz strange attractor [5] and the three-waves systems [4].

[1]The difference between the control and perturbed solutions are rescaled so the difference has the same size as the original perturbation.

A. Lorenz attractor

A strange attractor is a kind of chaotic dynamic system, whose trajectory is confined in a finite region and is such that no fragment of the trajectory is ever repeated. The Lorenz strange attractor has been very widely used as a prototype of chaotic behavior [12]. It has two regimes, which could represent two possibilities for any given season of the year (e.g."warm winter" and "cold winter"). Although apparently simple, in such a system it is hard to identify when a regime change will happen and how long it will last. The Lorenz Model equations are given as:

$$dx/dt = -\sigma x - y \quad (3)$$
$$dy/dt = -\rho x - y - xz \quad (4)$$
$$dz/dt = xy - \beta z \quad (5)$$

With $\sigma = 10, \rho = 28, \beta = 8/3$ as parameters, the resulting system is a strange attractor [12] (see Figure 2).

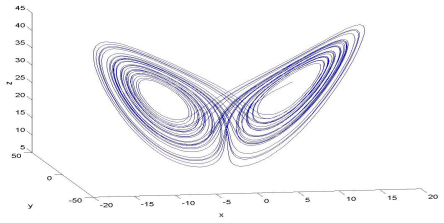

Fig. 2. Solutions of the Lorenz model equations showing two chaotic regimes.

In [4], the breeding was performed on the Lorenz model integrated with time steps $\Delta t = 0.01$, and a second run started from an initial perturbation $\delta x_0 = (\delta x_0, \delta y_0, \delta z_0)$ added to the control at time t_0. The difference δx between the perturbed and the control run was taken at every 8 times steps. The growth rate of the perturbation was measured per time step as [5]

$$g = \frac{1}{n} * (|\delta x|/|\delta x_0|).$$

Figure 3 illustrates the attractor built with under these conditions, using 4 colors to indicate bred vector growth intervals. The presence of a red star indicates that the bred vector growth in the previous 8 steps was greater than 0.064, and the blue stars indicate a negative growth rate. Figure 4 depicts the same system, considering only one axis: the abscissa (time)

separates the two regimes and each inflection point indicates the beginning of a new orbit of the system (similar graphics can be drawn for the other axes).

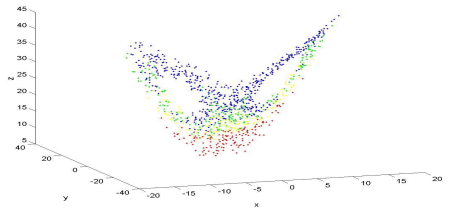

Fig. 3. Lorenz attractor colored with the bred vector classes.

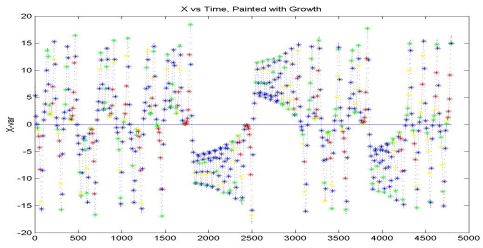

Fig. 4. $X(t)$ for Lorenz attractor colored with the bred vector classes.

From the observation of the system depicted in Figure 3 and 4, the following rules have been proposed in [5]:

Rule Ev.1:

When the growth rate exceeds 0.064 over a period of eight steps, as indicated by the presence of one (or more) red stars, the current regime will end after it completes the current orbit.

Rule Ev.2:

The length of the new regime is proportional to the number of red stars. For example, the presence of five or more stars in the old regime, indicating sustained strong growth, implies that the new regime will last four orbits or more.

The Lorenz attractor was then executed for 40,000 time steps (with 187 changes of regime) using these rules, obtaining 91.4% prediction accuracy (see [5] for more details).

B. Nonlinear coupled three-waves model

Nonlinear three-wave coupling is of general interest in many branches of physics [4]. It is for instance responsible for the generation and modulation of plasma waves in the planetary magnetosphere and solar wind [3].

In the simplest model for describing the temporal dynamics of resonant nonlinear coupling of three waves, one assumes the waves to be monochromatic, with the electric fields written in the form: $E_\alpha(x,t) = \frac{1}{2}A_\alpha(x,t)\exp\{i(k_\alpha x - \omega t)\}$, where $\alpha = 1,2,3$ and the time scale of the nonlinear interactions is much longer than the periods of the linear (uncoupled) waves.

In order for three-wave interactions to occur, the wave frequencies ω_α and wave vectors k_α must satisfy the resonant conditions

$$\omega_3 \cong \omega_1 - \omega_2, k_3 = k_1 - k_2 \tag{6}$$

Under these circumstances, the nonlinear temporal dynamics of the system can be governed by the following set of three first-order autonomous differential equations written in terms of the complex slowly varying wave amplitude [13]:

$$dA_1/d\tau = v_1 A_1 + A_2 A_3 \tag{7}$$

$$dA_2/d\tau = i\delta A_2 + v_2 A_2 A_1 A_3^* \tag{8}$$

$$dA_3/d\tau = v_3 A_3 A_1 A_2^* \tag{9}$$

where the variable $\tau = \chi t$, with χ is a characteristic frequency: $\delta = (\omega_1 - \omega_2 - \omega_3)/\chi$ is the normalized linear frequency mismatch and $v_\alpha = v_\alpha/\chi$ give the linear wave behaviors on the long time scale. In the experiments, following [4], wave A_1 is assumed to be linearly unstable ($v_1 > 0$) and the other two waves, A_2 and A_3, are linearly damped ($v_2 = v_3 \equiv -v < 0$) and thus $\chi = v_1$ [13] [1]. The system admits both periodic and chaotic waves. For the chaotic dynamics, a strange attractor is found – see Figure 5. As the Lorenz system dynamics, the coupled three-waves system has also two *seasons* in the strange attractor. However, the seasons (or regimes) have no symmetry. One regime is identified as a line formed by a curve on XY plane followed by another curve on the YZ plane (Figure 5). The other regime is characterized by the straight line in the intersection between the XY and YZ planes.

In [4], the breeding method was applied in the three-wave model with $\Delta t = 0.001$, and initial perturbation δx_0 added to the control at time t_0. The bred vector was calculated at every 8 time steps (using the same bred vector classification as in the Lorenz model. Figure 5 illustrates the attractor built with under these conditions, and Figure 6 depicts the system, considering only one axis (similar graphics can be drawn for the other axes). We can see that here the regimes alternate steadily (contrary to what happens with the Lorenz attractor); here the question is at which time step the "straight line" regime will change to the "curve line" regime.

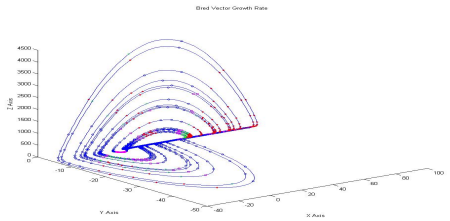

Fig. 5. Three-wave model attractor colored with the bred vector classes.

From the observation of the system depicted in Figures 5 and 6, the following rules have been proposed by for each wave:

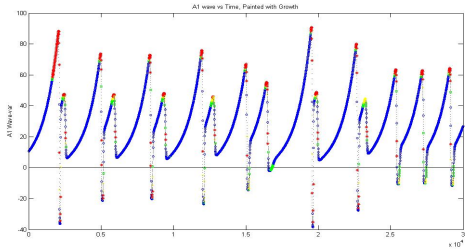

Fig. 6. $A1(t)$ for the three-wave model model.

Rule Ci.1:
The presence of a red star shows bred vector growth in the previous 8 steps was greater than 0.35. Appearing a red star implies the regime is going to change, for instance, if the trajectory is on the straight line, the dynamics is going to drop on a curve line; or, if the dynamics follows into a curve line, the trajectory will be to follow on the straight line.

Rule Ci.2:
Furthermore, the presence of two or three red stars for regime-2 indicates that the next regime will be on the straight line, with a long duration trajectory on this regime. Appearing more than three red stars into regime-2, this implies into a changing to the regime-2 with long duration (following on YZ and XY planes).

The 3-wave system was then executed for 20,000 time steps (with 23 changes of regime) using these rules, with several thresholds, obtaining 100% prediction accuracy with the best threshold for Rule 1 and 95.6 % for Rule 2 (see [4] for more details).

III. Neural Networks, Fuzzy Systems and Neuro-Fuzzy Systems

ANNs (Artificial Neural Networks) [8] aims at emulating the learning behavior of the human brain. The network is composed of nodes (with some calculation abilities) and weighted edges between nodes. Different ANN models have different constraints on the which nodes are connected. Usually, a back-propagation method is used to adjust the weights according to the difference between the desired and the calculated output. In the experiments reported here, we have used a simple neural network with a single hidden layer from the WEKA platform [17].

Fuzzy Systems [6] aim at emulating some of the human capacity of reasoning with vague information. Membership to a fuzzy set is measured by a number between 0 and 1 in the real scale, instead of simply 0 or 1 as in its classical counterpart. Most of the systems created using fuzzy sets theory are based on rules of thumb of the type “If condition then conclusion”, where the variables in both the condition and conclusion parts are associated to fuzzy sets.

The term “neuro-fuzzy” emerged in the mid-1980s and corresponds a mixture of artificial neural networks with fuzzy rule-based systems. A neuro-fuzzy system [9] is used to derive a fuzzy rule based system, whose defining parameters (fuzzy terms and rules) are learnt through training performed in a neural network manner upon the presentation of a set of pairs (input, desired output).

Artificial neural networks are very appropriate to deal with problems related to pattern recognition, but ill-suited to explain how the answers are obtained, functioning as a sort of ”black box”. On the other hand, fuzzy rule based systems, even working with inaccurate information, is appropriate to explain how results are obtained, but needs an expert to create the inference rules. Neuro-fuzzy systems, using neural networks learning apparatus to generate the inference rules of a fuzzy system, aims at maximizing the advantages of each model while minimizing their inconveniences.

Neuro-fuzzy systems are classified by the the type of fuzzy system they produce. The most noteworthy neuro-fuzzy systems are those derived from Sugeno and Mamdani fuzzy systems. Neuro-fuzzy systems can be thought of as a series of successive processing layers, but, contrary to the case of neural-networks, each layer is composed by its own specific kind of nodes and represents a particular stage in the processing of a fuzzy system. The initial layers of Sugeno and Mamdani neuro-fuzzy systems coincide and diverge greatly in the last ones. In the following we present the coinciding layers.

1) In the first layer, the current value of each input fuzzy variable is compared with the fuzzy terms associated with that variable, resulting in a compatibility degree for each term.
2) In the second layer, the compatibility degrees from the different input variables are combined, resulting in the overall compatibility degree of the potential rules.

In Sugeno neuro-networks systems, each potential rule is associated with a (usually linear) function of the values of the input variable: the rules have thus fuzzy terms in the left-side and a set of function parameters on the right-hand side. In Mamdami neuro-networks systems, the right-hand side of rules are fuzzy sets of the output variables, a characteristic that make them interpretable, contrary to what happens with Sugeno systems.

here the experiments we made with ANFIS (Adaptive-Network-Based Fuzzy Inference System) [16]. It uses the backpropagation algorithm to calculate the parameters of the fuzzy terms on the rule left-hand side the LMS algorithm (Least Square Means) for calculating the parameters on the right-hand side of the rule.

IV. Experiments

Our experiments view the prediction problem as a classification task. Here we have addressed the prediction of two chaotic systems: the Lorenz attractor and the three-wave problem. For one type of experiment with the Lorenz attractor (Experiment 1) and for the three-waves system, two kinds of training were made, one with only two classes (dichotomic) and another with a larger number of classes (multi-classes).

The testing was performed for these two kinds of training plus an extra one, in which we condensed the results of the multi-classes experiment to the two classes used in the dichotomic experiment. In all experiments, the size of the bred vectors obtained at each time step t, denoted as $s(t)$ were classified by colors: red ($s(t) > 0.064$), yellow ($0.032 < s(t) \leq .0064$), green ($0 < s(t) \leq 0.032$) and blue ($s(t) \leq 0$).

For both problems addressed here, the same ANN and ANFIS configuration were used. The neural network (ANN) and the ANFIS experiments were produced using platforms WEKA and MATLAB, respectively. Training with ANFIS was performed considering 3 triangular fuzzy terms for each input variable and constant output, with 300 epochs. A Multilayer Perceptron ANN with 3 layers have been used. The ANNs used the following configuration: learning rate = .3, momentum = .2, epochs = 500, 4 neurons in the input layer, n neurons on the output layer, where n is the number of classes, and (4+n)/2 neurons in the hidden layer.

A. Three-waves system

Here we describe our best results for the three-waves system and the configuration of the predictors (ANN and ANFIS) used to obtain those results. In the three-waves experiment we have used 170 samples, divided as: training (100), validation (30) and test (40). The input is is number of bred vectors in each class (color) found in preceding straight line regime.

In the two-classes experiment, the classes description are given as

- A: the straight line trajectory will last up to 1200 time steps
- B: the straight line trajectory will last more than 1200 time steps

In the multi-classes experiment, we have 15 classes, numbered from 0 to 14. Each class is associated to the interval bounding the number of time steps in which the straight line trajectory is predicted to last:

- 0: [0, 200]
- 1: [201, 400]
- 2: [401, 600]
- 3: [601, 800]
- 4: [801, 1000]
- 5: [1001, 1200]
- 6: [1201, 1400]
- 7: [1401, 1600]
- 8: [1601, 1800]
- 9: [1801, 2000]
- 10: [2001, 2200]
- 11: [2201, 2400]
- 12: [2401, 2600]
- 13: [2601, 2800]
- 14: $\geq$ 2801

The confusion matrices obtained by the use of ANFIS and the ANN on the test set for the multi-class experiments are given in Table I. The lines correspond to the real class and the column to system prediction. Table II bring the confusion matrices obtained by the use of ANFIS and the ANN on the two-classes experiments. Table III bring the confusion matrices, with classes A and B obtained by merging classes 0 to 5 and 6 to 14, respectively, from the multi-classes experiment.

a) ANFIS

Classes	0	1	2	3	4	5	6	7	8	9
0	14	2	1	0	0	0	0	0	0	0
1	2	1	0	0	0	0	0	0	0	0
2	0	0	1	0	0	0	0	0	0	0
3	0	0	0	1	0	0	0	0	0	0
4	0	0	0	0	2	2	0	0	0	0
5	0	0	0	0	0	3	1	0	0	0
6	0	0	1	0	0	0	0	0	0	0
7	0	0	0	0	0	1	2	3	0	0
8	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	1	1	1	0

b) ANN

Classes	0	1	2	3	4	5	6	7	8	9
0	16	0	0	0	0	0	0	0	1	0
1	1	1	1	0	0	0	0	0	0	0
2	0	0	1	0	0	0	0	0	0	0
3	0	0	1	0	0	0	0	0	0	0
4	0	0	0	0	4	0	0	0	0	0
5	0	0	0	0	3	0	0	1	0	0
6	0	0	0	0	0	0	0	0	1	0
7	0	0	0	0	1	1	0	2	2	0
8	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	1	2	0

TABLE I
CONFUSION MATRICES FOR MULTI-CLASSES EXPERIMENTS FOR THE THREE-WAVES PROBLEM.

ANFIS	A	B
A	15	2
B	3	20

ANN	A	B
A	15	2
B	1	22

TABLE II
CONFUSION MATRICES FOR TWO-CLASSES EXPERIMENTS FOR THE THREE-WAVES PROBLEM.

ANFIS	A	B
A	13	1
B	0	26

ANN	A	B
A	16	1
B	0	23

TABLE III
CONFUSION MATRICES FOR TWO-CLASSES EXPERIMENTS FOR THE THREE-WAVES PROBLEM WITH CLASSES MERGED FROM THE MULTI-CLASSES EXPERIMENTS.

Accuracy	ANFIS	ANN
15-classes	47.5%	50%
2-classes (original)	87.5%	92.5%
2-classes (merged)	97.5%	97.5%

TABLE IV
ACCURACY FOR THE LORENZ ATTRACTOR EXPERIMENTS.

Table IV brings the accuracy results for the 3-waves ex-

periments. We can see that ANN and ANFIS have approximate overall performances with a slight advantage for ANN. Moreover, they both perform poorly with a large number of classes but very well in the dichotomic cases. Finally, both had improved results when the training was done with a large number of classes, which were then merged into only two.

B. Lorenz attractor

Here we describe our best results for the Lorenz attractor and the configuration of the predictors (ANN and ANFIS) used to obtain those results. In the Lorenz attractor experiment we have used 1275 samples, divided as: training (800), validation (200) and test (275). The input is is number of bred vectors in each class (color) found in each orbit.

We have performed 3 types of experiments using the Lorenz attractor. In Experiment T1, we are only interested in the prediction of when the regime will change. In Experiment T2 we are interested in knowing whether the regime will go unchanged in the next orbit and, if it is supposed to change, in which orbit it will effectively change. Experiment T3 is a mix of Experiments 1 and 2. Experiment 1 is similar to what can be found in the literature (see **?? ??**). Experiments 2 and 3 intend to verify whether extra information may be useful in the prediction process. This is relevant in certain applications, in which this kind of prediction is an indicator of intensity of a phenomenon, as in climate prediction for instance.

1) Experiment T1: In the two-classes experiment, the classes description are given as

- A: the current regime will not change until 5 orbits
- B: the current regime will change after 5 orbits

In the multi-classes experiment, we have 10 classes, numbered from 0 to 9. Each class is associated to the number of orbits that are predicted to occur before regime change; class 9 means that the number of orbits before regime change is larger than 8.

The confusion matrices obtained by the use of ANFIS and the ANN on the test set for the multi-class experiments are given in Table V. The lines correspond to the real class and the column to system prediction. Table VI bring the confusion matrices obtained by the use of ANFIS and the ANN on the two-classes experiments. Table VII bring the confusion matrices, with class A corresponding to class 0 (i.e. the regime will change in the next orbit), and class B obtained by merging classes 1 to 9, from the multi-classes experiment. Classes A and B thus mean that "the regime will change in the next orbit", and "the trajectory will stay in the current regime for more than one orbit".

Table VIII brings the accuracy results for the Lorenz attractor experiments. We can see that ANN and ANFIS have approximate overall performances with a slight advantage for ANN. Moreover, they both perform reasonably well with a large number of classes but much better results have been obtained in the dichotomic cases. Finally, ANN obtained improved results when the training was done with a large number of classes, which were then merged into only two, contrary to ANFIS, which had its performance decreased.

a) ANFIS

Classes	0	1	2	3	4	5	6	7	8	9
0	104	14	4	0	0	0	0	0	0	0
1	9	46	7	2	0	0	0	0	0	0
2	0	13	17	2	0	3	0	0	0	0
3	0	2	7	5	7	3	0	0	0	0
4	0	0	0	3	4	3	2	0	0	0
5	0	0	1	1	1	3	0	0	0	0
6	0	0	0	0	3	1	0	0	0	0
7	0	0	0	0	1	1	0	0	0	0
8	0	0	0	0	0	2	0	0	0	0
9	0	1	0	0	1	1	1	0	0	0

b) ANN

Classes	0	1	2	3	4	5	6	7	8	9
0	115	4	1	0	2	0	0	0	0	0
1	5	47	11	1	0	0	0	0	0	0
2	0	9	21	2	3	0	0	0	0	0
3	0	1	8	10	5	0	0	0	0	0
4	0	0	1	4	6	1	0	0	0	0
5	0	0	1	2	3	0	0	0	0	0
6	0	0	0	1	3	0	0	0	0	0
7	0	0	0	1	1	0	0	0	0	0
8	0	0	0	0	2	0	0	0	0	0
9	1	0	0	0	2	0	0	0	0	1

TABLE V
CONFUSION MATRICES FOR MULTI-CLASSES EXPERIMENTS FOR THE LORENZ ATTRACTOR: A) ANFIS B) ANN.

ANFIS	A	B
A	111	11
B	9	144

ANN	A	B
A	106	16
B	5	148

TABLE VI
CONFUSION MATRICES FOR TWO-CLASSES EXPERIMENTS FOR THE LORENZ ATTRACTOR: A) ANFIS B) ANN.

ANFIS	A	B
A	104	18
B	9	144

ANN	A	B
A	115	7
B	6	147

TABLE VII
CONFUSION MATRICES FOR TWO-CLASSES EXPERIMENTS FOR THE LORENZ ATTRACTOR, WITH CLASSES MERGED FROM THE MULTI-CLASSES EXPERIMENTS: A) ANFIS B) ANN.

Accuracy	ANFIS	ANN
10-classes	72.72%	65.09%
2-classes (original)	92.72 %	92.36 %
2-classes (merged)	90.18%	95.27%

TABLE VIII
ACCURACY FOR THE LORENZ ATTRACTOR EXPERIMENTS.

2) Experiments T2 and T3: Experiments T2 and T3 are fragments of a series of experiments reported in [15].

Experiment T2 is multi-classes with 13 classes numbered from 0 to 12, described as:

- 0: The trajectory will stay in the current regime.
- k: The regime will change in the next orbit and will then

Classes	0	1	2	3	4	5	6	7	8	9	10	11	12
0	140	7	2	3	0	0	0	0	1	0	0	0	0
1	5	41	10	0	1	0	0	0	0	1	0	0	0
2	0	0	27	2	0	0	0	0	0	0	0	0	0
3	0	0	2	6	3	0	0	0	0	0	0	0	0
4	0	0	1	2	5	1	0	0	0	0	1	0	0
5	0	0	0	0	3	3	0	0	0	0	0	0	0
6	0	0	0	0	0	2	0	0	0	0	0	0	0
7	0	0	0	0	0	1	0	0	1	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	1	0	0
11	0	0	0	0	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	0	1

TABLE IX
CONFUSION MATRIX FOR EXPERIMENT T2 FOR THE LORENZ ATTRACTOR USING ANFIS WITH 13 CLASSES

Classes	1	2	3	4	5	6
1	44	8	1	1	0	0
2	8	13	12	2	0	0
3	4	7	48	3	0	2
4	1	1	4	41	9	2
5	0	0	0	0	27	2
6	0	0	0	0	3	32

TABLE X
CONFUSION MATRIX FOR EXPERIMENT T3 FOR THE LORENZ ATTRACTOR USING ANFIS WITH 6 CLASSES

stay in the new regime for k orbits.

The confusion matrix obtained by the use of ANFIS is given in Table IX.

Experiment T3 is multi-classes with 6 classes numbered from 1 to 6 described as:

- 1: The trajectory will stay in the current regime for more than two orbits.
- 2: The trajectory will stay in the current regime for two more orbits.
- 3: The trajectory will stay in the current regime for one more orbit.
- 4: The regime will change in the next orbit and will then stay in the new regime for 1 more orbit.
- 5: The regime will change in the next orbit and will then stay in the new regime for 2 more orbits.
- 6: The regime will change in the next orbit and will then stay in the new regime for more than 2 orbits.

The confusion matrix obtained by the use of ANFIS is given in Table X. The accuracy in Experiments T2 and T3 are given in XI.

Accuracy	ANFIS	ANN
Experiment 2	81.45%	80.00%
Experiment 3	74.54%	70.90%

TABLE XI
ACCURACY FOR THE LORENZ ATTRACTOR EXPERIMENTS T2 AND T3.

We see that for both learning systems the results can be considered very good, considering that the system is predicting long-term events. The accuracy in Experiment T3 is lower than both Experiments T1 and T2. Only new experiments will tell whether it is better to split the classes from Experiments T1 and T2, train separately and make a post-processing to obtain the final classification (special care would have to be taken for the class that characterizes immediate regime change).

V. CONCLUSIONS AND FUTURE WORK

We have investigated the ability of ANFIS neuro-fuzzy system to evaluate the predictability of chaotic systems. This task is formulated as a classification problem, where classes of dynamics are identified. We have performed experiments on two systems, the well-known Lorenz attractor and the coupled three-waves model. We have compared our results with a standard artificial neural network (multilayer perceptron). For the three-waves problem and for Experiment T1 for the Lorenz attractor, we trained the learning systems using: i) two classes and ii) a larger number of classes (10 for the Lorenz attractor and 15 for the three-waves systems). We have also used a post-classification strategy to obtain two classes from the multi-classes experiments. We obtained very good results, specially in the dichotomic case, both when training was performed directly with two classes as well as when the two classes were obtained from merging classes in the multi-classes experiments, with a slight advantage for the ANN over ANFIS. In Experiments T2 and T3 for the Lorenz attractor we have explored the possibility of using the data for making predictions in problems in which other factors than the main prediction, such as intensity, play a role.

The bred vector methodology has already been applied to analyze the Lorenz system [11] [14], and neural network was employed to investigate the classes of the dynamics for this system [14]. Here, the breeding and a neuro-fuzzy system are used to identify the system behavior. In the present work, the approach with bred vector and a neuro-fuzzy system has been applied for the study of predictability to the nonlinear coupled waves from the solar physics for the first time.

The results obtained so far show that neuro-fuzzy systems of the Sugeno type (ANFIS) are useful for the prediction of chaotic systems. We are currently investigating the use of other systems (neuro-fuzzy and otherwise) that produce fuzzy systems by learning, in particular those based in the Mamdani paradigm [6]. The ultimate goal is to use the derived fuzzy systems as a basis for the automatic production of interpretable rules, such as those created by observation for the Lorenz attractor [5] and the three-waves system [4].

We also intend to investigate the use of asymmetrical means of quantifying the accuracy of predictions. In some contexts, it is more harmful that an event occurs before expected. For others, more damage is produced when an event occurs ahead of what is expected. For instance, a crop may be lost when the rain comes either too early or too late but at different costs.

Finally, we intend in the future to quantify the confidence on a particular prediction and also to explore the tradeoff between imprecise but trustful predictions versus precise but less trustful ones.

ACKNOWLEDGMENT

The authors thank FAPESP and CNPq for research support.

REFERENCES

[1] S.R. Lopes and A.-L. Chian, A coherent nonlinear theory of auroral Langmuir-Alfvén-whistler (LAW) events in the planetary magnetosphere. Astron. Astrophys, vol. 365, pp. 669 676, 1996.
[2] K.T. Alligood, T.D. Sauer and J.A. Yorke, Chaos: an introduction to dynamical systems. Springer-Verlag, 1996.
[3] A.C.-L. Chian, S.R. Lopes and M.V. Alves, Nonlinear excitation of Langmuir and Alfvén waves by auroral whistler waves in the planetary magnetosphere. Astron. Astrophys., vol. 288, pp. 981–984, 1994.
[4] R. S. Cintra and H. F. Campos Velho, Predictability for a Chaotic Solar Plasma System. Symposia: Advanced Modeling on Computational Fluid Dynamics. 2008.
[5] E. Evans, E., N.K. Bathi, J. Kinney, L. Pann. M. Peña, S-C Yang, E. Kalnay and J. Hansen, Rise Undergraduates Find That Regime Changes in Lorenz's Model are Predictable. Bull. Amer. Meteor. Soc., v. 85, 521524. 2004.
[6] F. Gomide, W. Pedrycz, An introduction to Fuzzy Sets, MIT Press, 1998.
[7] D. Guégan and J. Leroux, Predicting Chaos with Lyapunov Exponents: Zero Plays no Role in Forecasting Chaotic Systems, Chaotic Systems. Ed. Esteban Tlelo-Cuautle. InTech. http://www.intechopen.com/articles/show/title/predicting-chaos-with-lyapunov-exponents-zero-plays-no-role-in-forecasting-chaotic-systems Access: 04 mar. 2011.
[8] S. Haykin, Neural Networks: A Comprehensive Foundation. New Jersey: Prentice Hall. 1999.
[9] C.-T. Lin and C. S. G. Lee. Neural Fuzzy Systems: A Neuro-Fuzzy Synergism to Intelligent Systems. Upper Saddle River, NJ: Prentice Hall, 1996.
[10] E. Kalnay, Atmospheric modeling, data assimilation and predictability. Cambridge University Press.
[11] E. Kalnay, M. Peña, S.-C. Yang and M. Cai, Breeding and predictability in coupled Lorenz models. In Proceedings of the ECMWF, 2002.
[12] Lorenz, E. N., 1963: Deterministic non-periodic flow. J. Atmos. Sci., 20, 130141, 1963.
[13] C. Meunier, M.N. Bussac and G. Laval., Intermittency at the onset of stochasticity in nonlinear resonant coupling processes. Physica, vol. D4, pp. 236243, 1982.
[14] A. Pasini, V. Pelino, Can We Estimate Atmospheric Predictability by Performance of Neural Network Forecasting? The Toy Case Studies of Unforced and Forced Lorenz Models, Proc. CIMSA'2005, 2005.
[15] Previsibilidade em sistemas caticos utilizando o sistema Neuro-Difuso ANFIS P.L.B. dos Santos, S. Sandri and H.F. de Campos Velho, submitted (in Portuguese).
[16] J. Shing and R. Jang, Anfis: Adaptive-network-based fuzzy inference systems. IEEE Trans. on Systems. Man, and Cybernetics, v. 23, p. 714723, 1993.
[17] WEKA Waikato Environment for Knowledge Analysis. http://www.cs.waikato.ac.nz/ ml/weka/ Acess: 25 jul. 2012.
[18] Z. Toth and E. Kalnay, Ensemble forecasting at NCEP and the breeding method. Monthly Weather Review, v. 126, 32923302. 1997.

APÊNDICE B - TRABALHO COMPLETO EM CONGRESSO

Previsibilidade em sistemas caóticos utilizando o sistema *Neuro-Difuso* ANFIS

Pettras Leonardo Bueno dos Santos[1]
Sandra Aparecida Sandri[1]
Haroldo Fraga de Campos Velho[1]

[1] Instituto Nacional de Pesquisas Espaciais (INPE)
Laboratório Associado de Computação Aplicada (LAC)
Avenida dos Astronautas, 1758
12227-010 São José dos Campos, SP
pettrasleonardo@yahoo.com.br
{sandri, haroldo}@lac.inpe.br

Abstract. Os sistemas caóticos apresentam grande sensibilidade às condições iniciais. Pequenas alterações nestas condições iniciais podem levar a resultados muito diferentes da trajetória original do sistema. Esta característica faz com que seja muito difícil prever o comportamento destes sistemas, principalmente porque em várias aplicações práticas, as condições iniciais são obtidas com instrumentos de medida, os quais estão sujeitos a erros de precisão. A previsibilidade do comportamento de sistemas caóticos é uma área de grande importância porque muitos fenômenos do mundo real apresentam algum tipo de comportamento caótico. Para tentar realizar algum tipo de previsão em sistemas caóticos, algumas técnicas têm sido utilizadas, alguns exemplos são: "*bred vectors*", "*singular vectors*", "*Perturbed Observation*" e Coeficientes Locais de *Lyapunov*. Neste trabalho, utilizamos a técnica de "*bred vectors*" para gerar pares de entrada/saída desejada para o sistema ANFIS. A partir do treinamento com esses pares o ANFIS gera um sistema *Fuzzy* do tipo *Takagi-Sugeno*, tal sistema pode ser utilizado com um conjunto de testes para verificar a eficácia do modelo. Neste trabalho, o sistema *Takagi-Sugeno* gerado pelo ANIFS é utilizado para predizer se a trajetória do sistema de *Lorenz* vai mudar de região ou não, e caso mude, após quantas a trajetória voltará à região atual.

Keywords: *Lógica Fuzzy, Sistemas Caóticos, Neuro-Fuzzy, Bred Vectors*.

1 Introdução

Os sistemas caóticos possuem grande sensibilidade às condições iniciais, pequenas alterações nas condições iniciais podem levar a uma trajetória bem diferente da trajetória do sistema original. Porém, muitas aplicações do mundo real que são modeladas através de sistemas caóticos estão sujeitas a pequenos erros de precisão em instrumentos de medidas, o que torna a tarefa de realizar algum tipo de previsão com esses sistemas uma tarefa complicada. Muitos trabalhos têm sido desenvolvidos com o objetivo de conseguir algum tipo de previsibilidade em sistemas caóticos. Por exemplo, em [1], [2], [3] e [4] os autores utilizam a técnica conhecida como "*bred*

vectors" para prever o comportamento de sistemas caóticos. Já em [5] e [6] são utilizados o Expoentes Locais de Lyapunov – LLE (do inglês, *Local Lyapunov Exponents*). Em [7] os autores explicam o sistema de previsão de tempo global por "*ensemble"* do Centro de Previsão de Tempo e Estudos Climáticos (CPTEC) – INPE. Em (PALMER, 1998) é utilizada a técnica "*singular vectors*". Em [8] os autores fazem um estudo comparativo entre "*bred vectors*", "*singular vectors*" e PO ("*Perturbed Observation*").

Neste trabalho, o sistema caótico utilizado como objeto de estudo foi o atrator de *Lorenz*, o qual será explicado na seção seguinte. Ao gerar um gráfico através das equações de *Lorenz* (Equações 1, 2 e 3), é possível notar que o atrator gera órbitas em duas regiões (Figura 1), o objetivo deste trabalho é verificar qual será o comportamento futuro do sistema dada a órbita e a região atuais, da seguinte forma: se o sistema vai permanecer na mesma região, quantas órbitas ainda serão geradas nesta região, ou se o sistema for mudar de região, quantas órbitas vai durar até voltar à região atual.

O modelo utilizado neste trabalho é o sistema *Neuro-Difuso* ANFIS (*Adaptive-Network-Based Fuzzy Inference System*) [9], o qual utiliza pares de entrada / saída desejada como treinamento e ao final deste treinamento gera um sistema de inferência difuso baseado no modelo Takagi-Sugeno [10, 11].

2 Atrator de *Lorenz*

Edward Lorenz desenvolveu em seus trabalhos [12, 13, e 14] um modelo matemático simplificado para descrever movimentos atmosféricos. A partir destes trabalhos percebeu que: pequenas variações nos valores iniciais das variáveis de seu modelo resultavam em valores muito divergentes. Em sistemas dinâmicos complexos, estes resultados "instáveis" dizem respeito à evolução temporal como função de seus parâmetros e variáveis. Lorenz em sua pesquisa de sistemas dinâmicos usou três equações para representar graficamente o comportamento dinâmico através de computadores, descreveu um sistema relativamente simples, e verificou que a partir de estados iniciais ligeiramente diferentes, o sistema de equações diferenciais resultava em soluções completamente diferentes entre si. O sistema de Lorenz consiste de três equações diferenciais ordinárias de primeira ordem, acopladas [15].

$$\frac{dx}{dt} = -\sigma x - y \qquad (1)$$

$$\frac{dy}{dt} = -\rho x - y - xz \qquad (2)$$

$$\frac{dz}{dt} = xy - \beta z \qquad (3)$$

Nas equações acima, utilizando os parâmetros $\sigma = 10$, $\rho = 28$ e $\beta = 8/3$, suas soluções numéricas levam a um atrator imerso em um espaço tridimensional com

coordenadas (x, y, z) [16]. Neste trabalho, sempre que nos referirmos ao Sistema de Lorenz, estamos considerando estes valores de parâmetros. A Figura 1 ilustra o Atrator Lorenz.

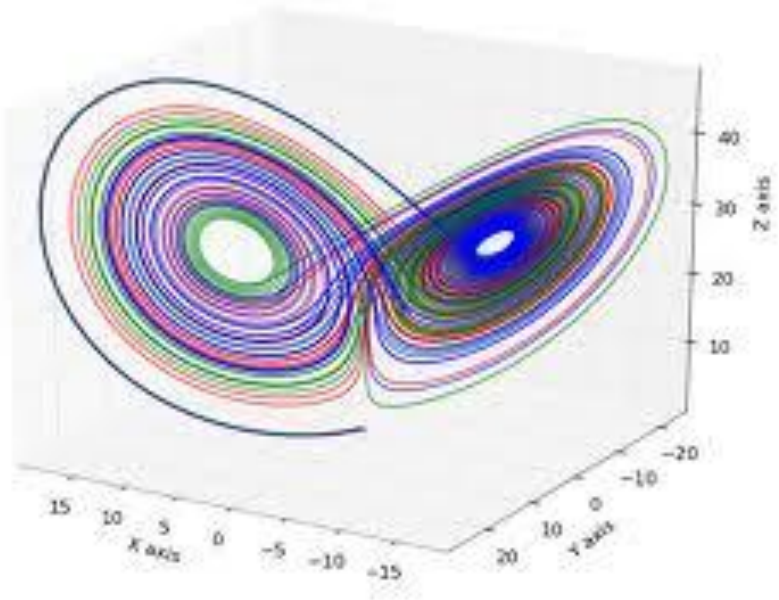

Figura 1 – Sistema Atrator de Lorenz

3 *Bred Vectors*

Nos trabalhos desenvolvidos em [1] e [2] os autores utilizaram uma técnica proposta por [3] conhecida como "*bred vectors*" ou "*breeding method*" para realizar a previsão de comportamento em sistemas caóticos, principalmente no Modelo de Lorenz. O método "*bred vectors*" consiste nos seguintes passos [3]:

a) Adicionar uma perturbação pequena e arbitrária para a análise do sistema (estado inicial), sendo t0 o tempo inicial.

b) Integrar ambos os modelos, o que sofreu a perturbação e o que não sofreu para um curto período (t1 – t0).

c) Subtrair os resultados obtidos entre os modelos.

d) Reduzir o campo de diferença de modo que tenha a mesma norma (por exemplo, amplitude RMS ou Energia Cinética de Rotação) como perturbação inicial.

e) Adicionar esta perturbação à próxima análise (tempo t1).

f) Repetir (b) – (e) nos tempos subseqüentes.

Nota-se que, uma vez que a perturbação inicial é introduzida no passo (a), o desenvolvimento do campo de perturbação é determinado dinamicamente pelo fluxo de evolução do sistema [3]. Através das sucessivas aplicações e comparações com os

resultados esperados (sem perturbação) é possível inferir alguns padrões e até mesmo algumas regras sobre o comportamento do sistema. A Figura 2 exemplifica a utilização dos *"bred vectors*".

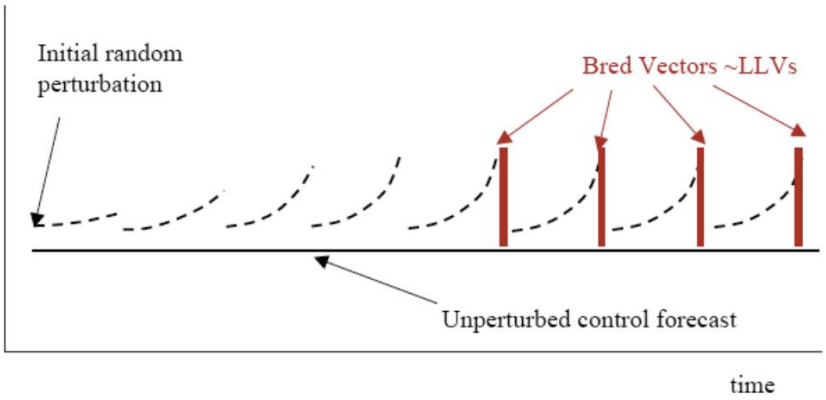

Figura 2 – *Bred Vectors*

A técnica dos "*Bred Vectors*" para previsibilidade em sistemas caóticos tem sido bastante utilizada atualmente, principalmente em pesquisas relacionadas à atmosfera, clima ou tempo, por exemplo, em [1], [2], [3] e [4].

Através da Figura 1 é possível notar que o Sistema de *Lorenz* forma duas regiões de atração, as quais os autores de [2] chamaram de Regime Quente ("*Warm Regime"*) e Regime Frio (*"Cold Regime"*). No referido trabalho os autores compararam os resultados com e sem perturbação e chegaram a duas regras sobre o comportamento do Sistema de *Lorenz* (a Figura 3 mostra uma forma de visualizar estas regras):

Regra 1: Quando a taxa de crescimento for superior a 0,064 sobre um período de 8 passos, conforme indicado presença de uma ou mais estrelas vermelhos (Figura 3) o atual regime terminará após completar a órbita atual.

Regra 2: O comprimento do novo regime é proporcional ao número de estrelas vermelhas. Por exemplo, a presença de cinco ou mais estrelas no antigo regime, indicando um forte crescimento sustentado, implica que o novo regime vai durar quatro órbitas ou mais.

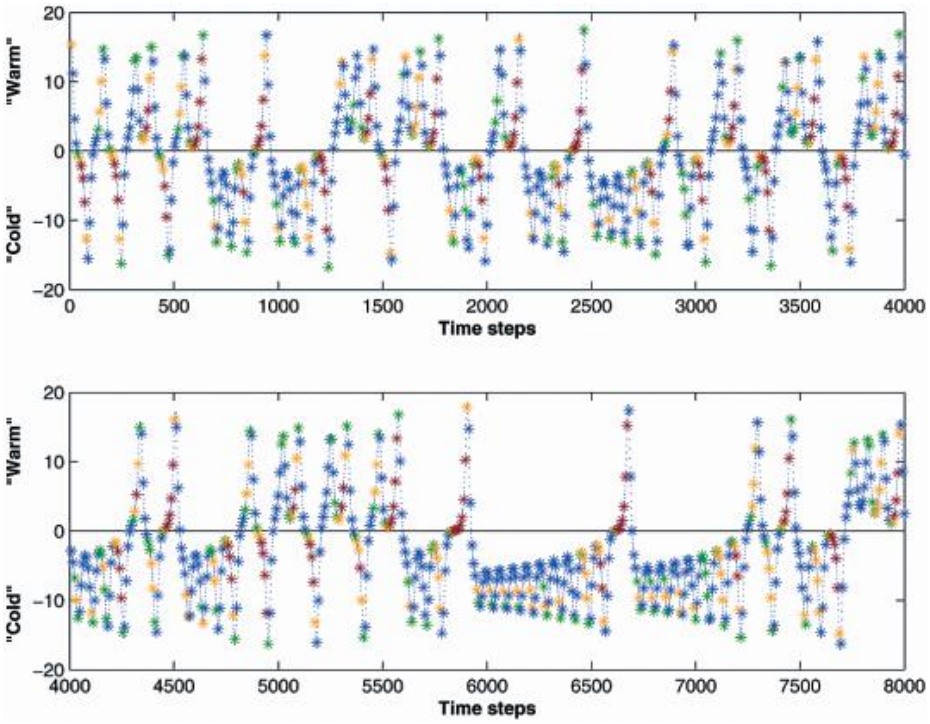

Figura 3 – Série temporal x(t) em função do número de passos

Na Figura 3, cada estrela colorida é uma amostra do tamanho do *bred vector* após 8 passos (pontos), ou seja, a distância do ponto com perturbação em relação ao ponto da trajetória original. Cada cor significa uma seguinte faixa de valores:

- Azul: *bred vector* menor que 0. Neste caso significa que a trajetória perturbada está se aproximando da trajetória original.
- Verde: *bred vector* maior que 0 e menor ou igual 0,032.
- Amarelo: *bred vector* maior que 0,032 e menor ou igual 0,064.
- Verde: *bred vector* maior que 0,064.

Cada volta que gera um pico no gráfico representa uma órbita no atrator (Figura 1). Os autores chegaram às regras citadas através de observação e contando manualmente o número de estrelas em cada volta, assim conseguiram identificar padrões e concluíram as regras.

O objetivo deste trabalho é utilizar uma ferramenta que automatize esta tarefa e consiga reconhecer padrões apresentados na Figura 4. Para, isso utilizamos o sistema *Neuro-Difuso*, ANFIS, que trabalha com pares de entrada / saída desejada e através de um processo de treinamento utilizando estes pares gera um sistema difuso do tipo *Takagi-Sugeno*, tal sistema pode ser utilizado para calcular uma saída, dado uma determinada entrada, em geral, uma entrada que não foi utilizada na etapa de treinamento, pois o objetivo é conseguir uma generalização que será utilizada como classificação. Neste trabalho utilizamos como entradas o número de estrelas de cada

cor em cada uma das órbitas da Figura 4, e como saída a informação se o a trajetória vai mudar de região e, caso mude, o número de órbitas que vai durar a nova região.

4 Sistemas *Neuro-Difusos* e ANFIS

O termo *Neuro-Difuso (Neuro-Fuzzy)* surgiu em meados da década de 1980 e significa uma mescla de RNAs (Redes Neurais Artificiais) com Sistemas de Inferência Difusos, resultando em um sistema inteligente híbrido que potencializa as características destes dois importantes paradigmas [17]. A rigor, qualquer sistema que misture os paradigmas de sistemas difusos e sistemas conexionistas poderia ser chamado de "*Neuro-Difuso*", como, por exemplo, a utilização de um controlador nebuloso para alterar dinamicamente a taxa de aprendizado de uma rede neural. No entanto, o termo é utilizado para um tipo específico de sistema que de certa forma engloba os dois paradigmas. Nestes sistemas, os termos e regras de um sistema nebuloso são aprendidos mediante a apresentação de pares (entrada, saída desejada) [18].

As Redes Neurais Artificiais consistem em uma ótima ferramenta para trabalhar em problemas relacionados a reconhecimento de padrões, porém apresentam certa dificuldade para explicar como as respostas são obtidas, funcionando como uma espécie de "caixa-preta". Já os Sistemas de Inferência Difusos, mesmo trabalhando com informações imprecisas, apresentam grande facilidade para explicar como os resultados são obtidos, porém necessitam fortemente de um especialista "externo" para criar as regras de inferência. Os sistemas *Neuro-Difusos* utilizam o aprendizado das Redes Neurais Artificiais para gerar as regras de inferência de um Sistema Difuso, com o objetivo de potencializar as vantagens de cada modelo e minimizar suas desvantagens.

O modelo ANFIS (do inglês, *Adaptive-Network-Based Fuzzy Inference System*) é uma Rede Neural Artificial, proposta por [9], que consiste em implementar um Sistema de Inferência Difuso através da própria RNA, de forma que os algoritmos de aprendizado possam ser utilizados para ajustar o Sistema de Inferência Difuso [19]. A Figura 4 representa um exemplo de sistema ANFIS.

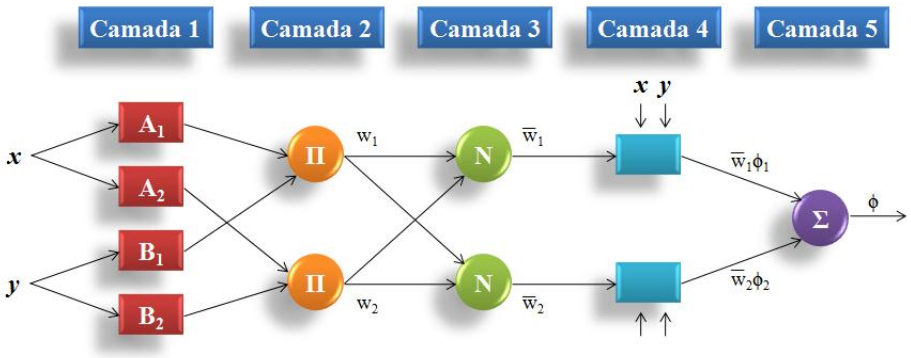

Figura 4 – Exemplo de Sistema ANFIS. Adaptada de [19].

O modelo apresentado na Figura 4 é formado por cinco camadas, as quais serão descritas a seguir:

- Camada 1 – Sendo x e y os valores de entrada para o sistema, os nós desta camada (A_1, A_2, B_1 e B_2) representam o grau de pertinência destes nós em relação aos valores de entrada, ou seja, esta camada realiza a fase de fuzzificação do sistema. Qualquer função contínua e diferenciável no intervalo [0,1] pode ser utilizada como função de ativação nesta camada, por exemplo, a função sino (equação 4), na qual os valores a, b e c são parâmetros utilizados para mudar o comportamento da função.

$$\mu_A(x) = \frac{1}{1+[(\frac{x-c}{a})^2]^b} \tag{4}$$

- Camada 2 – As saídas dos nós desta camada representam a "força de disparo de uma regra". A equação 5 é utilizada como função de ativação nesta camada, porém qualquer operador de norma T que represente o operador AND lógico pode ser utilizado como função de ativação nesta camada [9].

$$w_i = \top(\mu_{A_i}(x),\ \mu_{B_i}(x)), \quad i = 1,2 \tag{5}$$

- Camada 3 – Cada nó desta camada funciona como um operador de normalização do nível de disparo das regras, definido, de acordo com [9], como a Equação 6.

$$\overline{w_i} = \frac{w_i}{w_1 + w_2}, \quad i = 1,2 \tag{6}$$

- Camada 4 – Esta camada calcula o produto entre as saídas da camada 3 (níveis de disparo normalizados) e os valores de entrada x e y, ponderados pelos valores (p, q e r), os quais [9] chamou de parâmetros conseqüentes, de acordo com a equação 7.

$$\overline{w_i} z_i = \overline{w_i}(p_i x + q_i y + r_i), \quad i = 1,2 \tag{7}$$

- Camada 5 – Esta camada realiza a fase de *defuzzificação* do sistema. É formada por apenas um nó que realiza o somatório das saídas da camada 4, de acordo com a equação 8.

$$\sum \overline{w_i} z_i = \frac{\sum w_i z_i}{\sum w_i}, \quad i = 1,2. \tag{8}$$

O sistema ANFIS, apresentado na Figura 4, utiliza o algoritmo *backpropagation* para calcular os parâmetros antecedentes das regras (Equação 4) e o algoritmo LMS

(do inglês, *Least Mean Square*)[1] para calcular os parâmetros conseqüentes (Equação 7). Cada iteração do algoritmo é dividida em duas partes [19]:

i) Os dados de entrada são propagados e os parâmetros conseqüentes são calculados de acordo com o LMS, enquanto os parâmetros antecedentes são fixados durante todo o ciclo através do conjunto de treinamento.

ii) As taxas de erro são retropropagadas e o *backpropagation* é utilizado para atualizar os parâmetros antecedentes, enquanto os parâmetros conseqüentes fixados.

5 Resultados

Note que na Figura 4 foram gerados 8000 pontos na trajetória, através destes 8000 pontos existem em torno de 100 órbitas (cada órbita é uma volta que gera um pico no gráfico). Cada uma destas órbitas foi utilizada como um par de entradas / saída desejada no nosso sistema, e cada um destes pares é um padrão a ser utilizando para treinamento, validação ou teste. Com o objetivo de conseguir uma boa generalização, geramos em torno de 100 mil pontos na trajetória e assim conseguimos uma amostra com 1275 padrões (órbitas). Dividimos a amostra em três partes: treinamento (800 padrões), validação (200 padrões) e teste (275 padrões). Cada padrão possui 4 entradas, sendo cada entrada o número de estrelas de uma determinada cor em cada órbita. A saída desejada é definida de acordo com o número de voltas que vai permanecer na região atual ou, caso for mudar de região, de acordo com o número de voltas que vai permanecer na nova região. Os testes com o ANFIS foram realizados com o *software* MATLAB versão R2010a, o qual possui uma implementação do ANFIS que trabalha com arquivos de treinamento, validação e testes com pares de entradas / saídas desejadas e ao final do treinamento gera um arquivo com extensão ".fis", que é um sistema difuso do tipo *Takagi-Sugeno* utilizado no MATLAB. Para fins de comparação com outro método de reconhecimento de padrões, utilizamos Redes Neurais artificiais, neste caso, utilizamos o simulador WEKA [20], trabalhando sempre com uma camada oculta com o número de neurônios nesta camada calculado pela seguinte fórmula: (número de variáveis de entrada + número de classes)/2. O algoritmo de treinamento utilizado foi o backpropagation padrão com taxa de aprendizado 0,3 e termo de momentum 0,2, além disso, foram utilizadas sempre 5000 épocas para o treinamento da rede. Mais informações sobre as Redes Neurais Artificiais podem ser encontrados em [21, 22].

Em princípio, trabalhamos com as seguintes classes de saída:

- Classe 0: A trajetória se manterá na atual região.
- Classe x: A trajetória mudará de região e se manterá na nova região por x voltas.
-

[1] O LMS (*Least Mean Square*) é um algoritmo baseado na utilização de valores instantâneos para a função de custo [22]

Utilizando as 275 amostras de testes, chegamos a 81,45% de acertos, a seguinte matriz de confusão (Tabela 1) mostra esse resultado, nesta matriz as linhas representam a saída calculada e as colunas representam a saída desejada, desta forma, a diagonal principal (destacada em amarelo) representa os acertos do sistema. Utilizando uma Rede Neural Artificial a taxa de acertos foi de 80% com os mesmos dados.

Classes	0	1	2	3	4	5	6	7	8	9	10	11	12
0	140	7	2	3	0	0	0	0	1	0	0	0	0
1	5	41	10	0	1	0	0	0	0	1	0	0	0
2	0	0	27	2	0	0	0	0	0	0	0	0	0
3	0	0	2	6	3	0	0	0	0	0	0	0	0
4	0	0	1	2	5	1	0	0	0	0	1	0	0
5	0	0	0	0	3	3	0	0	0	0	0	0	0
6	0	0	0	0	0	2	0	0	0	0	0	0	0
7	0	0	0	0	0	1	0	0	1	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	1	0	0
11	0	0	0	0	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	0	1

Tabela 1 – Matriz de confusão (linhas representam a saída calculada e colunas representam a saída desejada).

Os resultados da Tabela 1 mostram se a trajetória vai permanecer na mesma região e, caso mude, quantas voltas vai permanecer na nova região. Porém, em algumas aplicações pode ser necessário apenas uma estimativa de quantas voltas a trajetória vai permanecer na nova região, para isso trabalhamos com 4 classes de saída:

- Classe 1: A trajetória se manterá na atual região.
- Classe 2: A trajetória mudará de região e se manterá na nova região de 1 a 3 voltas.
- Classe 4: A trajetória mudará de região e se manterá na nova região de 4 a 6 voltas.

- Classe 4: A trajetória mudará de região e se manterá na nova região mais de 6 voltas.

Utilizando as 275 amostras de testes, chegamos a 89,81% de acertos, a seguinte matriz de confusão (Tabela 2) mostra esse resultado, nesta matriz as linhas representam a saída calculada e as colunas representam a saída desejada, desta forma, a diagonal principal (destacada em amarelo) representa os acertos do sistema. Utilizando uma Rede Neural Artificial a taxa de acertos foi de 87,63% com os mesmos dados.

Classes	1	2	3	4
1	143	7	1	3
2	6	87	4	1
3	0	3	14	2
4	0	0	1	3

Tabela 2 – Matriz de confusão (linhas representam a saída calculada e colunas representam a saída desejada).

Nos dois casos anteriores, a saída foi modelada considerando se a trajetória vai permanecer na mesma região ou quantas voltas vai permanecer na nova região, caso mude. Porém, pode ser interessante saber também quantas voltas vai permanecer na região atual antes de mudar. Para esta análise, utilizamos duas abordagens.

Na primeira abordagem trabalhamos com 6 classes de saída, considerando também, caso não mude a região, quantas voltar vai permanecer na região atual:

- Classe 1: A trajetória se mantém na região atual por mais de 6 voltas.
- Classe 2: A trajetória se mantém na região atual de 4 a 6 voltas.
- Classe 3: A trajetória se mantém na região atual de 1 a 3 voltas.
- Classe 4: A trajetória mudará de região e se manterá na nova região de 1 a 3 voltas.
- Classe 5: A trajetória mudará de região e se manterá na nova região de 4 a 6 voltas.
- Classe 6: A trajetória mudará de região e se manterá na nova região por mais de 6 voltas.

Utilizando as 275 amostras de testes, chegamos a 80,36% de acertos, a seguinte matriz de confusão (Tabela 3) mostra esse resultado, nesta matriz as linhas representam a saída calculada e as colunas representam a saída desejada, desta forma, a diagonal principal (destacada em amarelo) representa os acertos do sistema. . Utilizando uma Rede Neural Artificial a taxa de acertos foi de 75,63% com os mesmos dados.

Classes	1	2	3	4	5	6
1	1	6	0	0	0	1
2	1	11	10	0	0	0
3	2	6	107	4	0	4
4	1	0	5	87	4	2
5	1	0	1	2	12	3
6	0	0	0	0	1	3

Tabela 3 – Matriz de confusão (linhas representam a saída calculada e colunas representam a saída desejada).

Na segunda abordagem também trabalhamos com 6 classes de saída, considerando, caso não mude a região, quantas voltar vai permanecer na região atual:

- Classe 1: A trajetória se mantém na região atual por mais de 2 voltas.
- Classe 2: A trajetória se mantém na região atual por exatamente 2 voltas.
- Classe 3: A trajetória se mantém na região atual por exatamente 1 volta.
- Classe 4: A trajetória mudará de região e se manterá na nova região por exatamente 1 volta.
- Classe 5: A trajetória mudará de região e se manterá na nova região por exatamente 2 voltas.
- Classe 6: A trajetória mudará de região e se manterá na nova região por mais de 2 voltas.
-

Utilizando as 275 amostras de testes, chegamos a 74,54% de acertos, a seguinte matriz de confusão (Tabela 3) mostra esse resultado, nesta matriz as linhas representam a saída calculada e as colunas representam a saída desejada, desta forma, a diagonal principal (destacada em amarelo) representa os acertos do sistema. . Utilizando uma Rede Neural Artificial a taxa de acertos foi de 70,9% com os mesmos dados.

Classes	1	2	3	4	5	6
1	44	8	1	1	0	0
2	8	13	12	2	0	0
3	4	7	48	3	0	2
4	1	1	4	41	9	2
5	0	0	0	0	27	2
6	0	0	0	0	3	32

Tabela 4 – Matriz de confusão (linhas representam a saída calculada e colunas representam a saída desejada).

Analisando as Tabelas de 1 a 4 é possível notar que a maior parte dos padrões que foram classificados de maneira errada estão próximos da diagonal principal (que seriam classificados como certos), poucos casos tiveram erros mais grosseiros, por exemplo, a saída desejada era Classe 1 e a saída calculada foi Classe 5. Isso, de certa forma mostra o potencial do modelo, inclusive podendo ser aumentada a porcentagem de acertos com alguns ajustes em trabalhos futuros. Além disso, nos resultados apresentados nas Tabelas 1,2 e 3 a taxa de acerto foi acima de 80%, somente na Tabela 4 tivemos uma taxa de acerto abaixo dos 80%. Comparando os resultados obtidos através o ANFIS com outro método de reconhecimento de padrões, no nosso caso Redes Neurais artificiais, o ANFIS apresentou resultados um pouco melhores.

5 Conclusão

Neste trabalho utilizamos uma ferramenta de reconhecimento de padrões para prever o comportamento de um sistema caótico. O objeto de estudo utilizado foi o modelo de Lorenz e a ferramenta escolhida foi o sistema Neuro-Difuso ANFIS. Os resultados são bastante animadores, pois conseguimos taxas de acerto acima de 80% prevendo se a trajetória vai ou não mudar de região no modelo de *Lorenz*, nosso sistema também faz uma estimativa sobre quanto tempo (voltas) a trajetória vai demorar a mudar de região ou quanto tempo vai durar a nova região, caso mude. Além disso, comparamos os resultados obtidos através do ANFIS com resultados obtidos através de Redes Neurais Artificiais, em todos os nossos testes o ANFIS apresentou resultados um

pouco melhores que as Redes Neurais. Esses resultados mostram um bom potencial desta metodologia para ser aplicada neste tipo de problema. Pretendemos em trabalho futuros utilizar outros sistemas caóticos como objeto de estudo, a fim de corroborar a utilização do ANFIS como ferramenta para a previsibilidade em sistemas caóticos.

Referências

[1] CINTRA, R. S.; CAMPOS VELHO, H. F. Predictability for a Chaotic Solar Plasma System. Symposia: *Advanced Modeling on Computational Fluid Dynamics*. 2008.

[2] EVANS, E; *et al*. Rise Undergraduates Find That Regime Changes in Lorenz's Model are Predictable. *Bull. Amer. Meteor. Soc*., v. 85**,** 521–524. 2004.

[3] TOTH, Z.; KALNAY, E. Ensemble forecasting at NCEP and the breeding method. *Monthly Weather Review*, v. 126**,** 3292–3302. 1997.

[4] NEWMAN, C.E.; READ, P.L.; LEWIS, S.R. Breeding vectors and predictability in the Oxford Mars GCM. *First International Workshop on Mars atmosphere modelling and observations*, 13-15. Granada, Spain. 2003.

[5] GUÉGAN, D.; LEROUX, J. Predicting Chaos with Lyapunov Exponents: Zero Plays no Role in Forecasting Chaotic Systems, Chaotic Systems. *Ed. Esteban Tlelo-Cuautle. InTech*. Disponível em: < http://www.intechopen.com/articles/show/title/predicting-chaos-with-lyapunov-exponents-zero-plays-no-role-in-forecasting-chaotic-systems >. Acesso em: 04 mar. 2011.

[6] ECKHARDTA, B.; YAOA, D. Local Lyapunov exponents in chaotic systems. *Physica D: Nonlinear Phenomena*, v. 65 (1-2), pp. 100-108. 1993.

[7] MENDONÇA, A. M.; BONATTI J. P. O Sistema de Previsão de Tempo Global por Ensemble do CPTEC. *Centro de Previsão de Tempo e Estudos Climáticos (CPTEC) – INPE. XII Congresso Brasileiro de Meteorologia*, Foz de Iguaçu-PR, 2002.

[8] HAMILL, T. M.; CHRIS SNYDER, R. E. M. A Comparison of Probabilistic Forecasts from Bred, Singular-Vector, and Perturbed Observation Ensembles. *Monthly Weather Review*, v. 128, 1835–1851. 2000.

[9] JANG, J. S. R. Anfis: Adaptive-network-based fuzzy inference systems. IEEE Trans. on Systems. *Man, and Cybernetics*, v. 23, p. 714–723, 1993.

[11] SUGENO, M.; KANG, G. T. Structure identification of fuzzy model. *Fuzzy Sets and Systems*. 28:329–346. 1986.

[10] TAKAGI, T.; SUGENO, M. Fuzzy identification of systems and its applications to modeling and control. *IEEE Transactions on Systems, Man, and Cybernetics*, 15(1):116–132. 1985.

[12] LORENZ, E. N. Deterministic non-periodic flow. *J. Atmos. Sci*., v. 20, p. 130-141, 1963.

[13] LORENZ, E. N. A study of the predictability of a 28-variable atmospheric model. *Tellus,* v. 17, p. 321-333, 1965.

[14] LORENZ, E. N. The predictability of a flow which possesses many scales of motion. *Tellus,* v. 21, p. 289-307,1969.

[15] CINTRA R. S.; CAMPOS VELHO, H. F; Todling, R. Redes Neurais Artificiais na Melhoria de Desempenho de Métodos de Assimilação de Dados: Filtro de Kalman. *TEMA. Tendências em Matemática Aplicada e Computacional*, v. 11, p. 29-39, 2010.

[16] CAMPANHARO, S. L. O.; Macau, E. E. N.; Ramos, F. M. Detectando a presença de caos em uma série temporal. Disponível em: <http://www.sbmac.org.br/eventos/cnmac/cd_xxviii_cnmac/resumos%20estendidos/andriana_campanharo_ST21.pdf >. Acesso em: 04 mar. 2012.

[17] REZENDE, S. O. Sistemas inteligentes: fundamentos e aplicações. *Ed. Manole*, São Paulo, 2003.

[18] SANDRI, S. A.; CORREA, C. Lógica Nebulosa. *V Escola de Redes Neurais, Promoção: Conselho Nacional de Redes Neurais*. ITA, São José dos Campos – SP. 1999. pp. c073-c090.

[19] JACINTHO, L. F. O. Redes Neuro-Difusos: Um Estudo de Caso em Diagnóstico de Alzheimer. *Monografia (Bacharel em Ciência da Computação)* – Universidade Federal do ABC (UFABC), Santo André, 2010.

[20] WEKA – Waikato Environment for Knowledge Analysis. Site oficial do WEKA, onde se encontram disponíveis para *download* os arquivos de instalação e ampla documentação. Disponível em: < http://www.cs.waikato.ac.nz/~ml/weka/ > Acesso em: 25 jul. 2012.

[21] HAYKIN, S. Neural Networks: A Comprehensive Foundation. *New Jersey: Prentice Hall*. 1999.

[22] BRAGA, A. P.; CARVALHO A. C. P. L. F.; LUDERMIR T. B**.** Redes Neurais Artificiais: Teoria e Aplicações. *Rio de Janeiro: Livros Técnicos e Científicos* (LTC), 2000.