UNIVERSIDADE DE BRASÍLIA
FACULDADE DE TECNOLOGIA
DEPARTAMENTO DE ENGENHARIA MECÂNICA

SISTEMA DE CONTROLE DE SISTEMAS CAÓTICOS SOBRE ÓRBITAS PERIÓDICAS

DANNIEL KENNETH BORGES COSTA

ORIENTADOR: TITO DIAS JUNIOR

DISSERTAÇÃO DE MESTRADO EM CIÊNCIAS MECÂNICAS

PUBLICAÇÃO: ENM.DM – 147A / 2010

BRASÍLIA/DF: MARÇO – 2010

UNIVERSIDADE DE BRASÍLIA

FACULDADE DE TECNOLOGIA

DEPARTAMENTO DE ENGENHARIA MECÂNICA

SISTEMA DE CONTROLE DE SISTEMAS CAÓTICOS SOBRE ÓRBITAS PERIÓDICAS

DANNIEL KENNETH BORGES COSTA

DISSERTAÇÃO SUBMETIDA AO DEPARTAMENTO DE ENGENHARIA MECÂNICA DA FACULDADE DE TECNOLOGIA DA UNIVERSIDADE DE BRASÍLIA COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS MECÂNICAS.

APROVADA POR:

_____

Prof. Dr. Tito Dias Junior, (ENM-UnB)
(Orientador)

_____

Prof. Dr. Guilherme Caribé Carvalho
(Examinador Interno)

_____

Prof. Dr. Emanuel Rocha Woiski
(Examinador Externo)

BRASÍLIA/DF, 23 DE MARÇO DE 2010

FICHA CATALOGRÁFICA

C837 Costa, Danniel Kenneth Borges

Sistema de controle de sistemas caóticos sobre órbitas periódicas/ Danniel Kenneth Borges Costa ; Tito Dias Junior (Orientador). – Brasília, 2009.

85 f. : il. ; 30 cm.

Dissertação de Mestrado – Universidade de Brasília, Faculdade de Tecnologia, Departamento de Engenharia Mecânica, 2009.

1. Engenharia de controle. 2. Redes neurais artificiais. 3. Algoritmos genéticos. 4. Órbitas caóticas. 5.Sistema de controle do caos. I. Dias Junior, Tito. II. Título.

CDU: 621.3

REFERÊNCIA BIBLIOGRÁFICA

COSTA, Danniel K. Borges. ***Sistema de Controle de Sistemas Caóticos Sobre Órbitas Periódicas.*** *2009. 85 f. Dissertação de Mestrado em Ciências Mecânicas - Faculdade de Tecnologia, Universidade de Brasília.*

CESSÃO DE DIREITOS

AUTOR: DANNIEL KENNETH BORGES COSTA

SISTEMA DE CONTROLE DE SISTEMAS CAÓTICOS SOBRE ÓRBITAS PERIÓDICAS: concepção, projeto e simulação de um sistema de controle para um sistema não-linear caótico (sistema de Lorenz) utilizando redes neurais artificiais e algoritmos genéticos.

GRAU: Mestre ANO: 2009

Danniel Kenneth Borges Costa
Rua Capelinha n° 50 - Centro.
75860-000 Quirinópolis – GO - Brasil.

AGRADECIMENTOS

Agradeço primeiramente a Deus, que sempre me deu forças nos momentos de maior dificuldade durante estes dois anos de trabalho. Agradeço também aos meus pais que sempre acreditaram em mim, mesmo quando tudo parecia dar errado.

Agradeço a Lívia, minha companheira, que secou muitas lágrimas nos dias de desespero, prazos curtos e múltiplos afazeres.

Agradeço ao meu orientador Tito, que mesmo com 777 outras coisas para fazer não desistiu de me orientar.

Danniel Kenneth Borges Costa

RESUMO

Este trabalho foi motivado pela falta, na teoria existente de controle de sistemas caóticos, do emprego de redes neurais artificiais em conjunto com algoritmos genéticos no desenvolvimento de um sistema de controle adaptativo eficiente e que esteja em concordância com aspectos de sistemas caóticos como o uso de apenas pequenas perturbações, da ordem de 10^{-2}, para efetuar o controle do sistema. Desta forma o objetivo deste trabalho é construir um software em linguagem de programação MATLAB, utilizando as teorias de redes neurais artificiais e algoritmos genéticos, capaz de controlar sistemas caóticos sobre órbitas periódicas. Como modelo de sistema caótico foi usado o sistema do atrator de Lorenz. Tal controle pode ser realizado através de um processo no qual uma pequena perturbação é aplicada a um sistema caótico a fim de gerar um comportamento desejado, caótico, periódico ou estacionário, no mesmo.
Foram discutidas algumas questões de grande importância relacionadas ao controle do caos, como o problema de direcionamento, isto é, como trazer uma trajetória para uma vizinhança próxima de uma posição desejada no atrator caótico, bem como testada a flexibilidade do sistema de controle desenvolvido sob diversas condições. Em suma, este trabalho vem contribuir para a área de estudo de sistemas de controle de sistemas caóticos abordando conteúdos que, juntos, fornecem um poderoso método de controle evolutivo-adaptativo. Como resultado foi obtido um sistema de controle robusto a ruídos, a diferentes condições iniciais e a períodos de controle desligado.

Palavras-chave: Engenharia de controle. Redes neurais artificiais. Algoritmos genéticos. Órbitas caóticas. Sistema de controle do caos.

ABSTRACT

This essay was motivated by the lack, in the existing theories of chaotic systems control, of use of artificial neural networks in conjunction with genetic algorithms in the implementation of an adaptive control system that is efficient and in accordance with aspects of chaotic systems such as the use of only small perturbations, around 10^{-2}, to effect the control system. Thus the objective of this work is to build a software, on MATLAB programming language, using the theories of artificial neural networks and genetic algorithms, able to control chaotic systems on periodic orbits. As a model of chaotic system was used Lorenz`s attractor system. Such control can be accomplished through a process in which a small perturbation is applied to a chaotic system to generate a desired behavior, chaotic, periodic or stationary in it. Some important issues related to control of chaos are discussed, as the problem of targeting, how to bring a path to a neighborhood near a desired position in the chaotic attractor, and tested the flexibility of the control system implemented under various conditions. In short, this work contributes to the study of control systems of chaotic systems approaching subjects that together provide a powerful method of evolutionary and adaptive control. The result was a control system robust to noise, different initial conditions and periods where the control perturbations are off.

Keywords: Control Engineering. Artificial neural network. Genetic algorithms. Chaotic orbits. Chaos controlling system's.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SÍMBOLOS E ABREVIATURAS

Símbolos Latinos

X	Variável proporcional à intensidade da convecção no sistema de Lorenz	
Y	Variável proporcional à diferença de temperatura entre as correntes de fluido ascendente e descendente no sistema de Lorenz	
Z	Variável proporcional à distorção do perfil de temperatura vertical, relativo a um perfil linear, no sistema de Lorenz	
t	Tempo	[s]
r	Número de Rayleigh e parâmetro de controle	
b	Relação entre grandezas	
s	Relação entre grandezas	

Símbolos Gregos

σ	Relação entre grandezas	

Grupos Adimensionais

w	Ponderação aplicada sobre uma entrada

Subscritos

i	Posicionamento linha em uma matriz
j	Posicionamento coluna em uma matriz
c	Valor crítico
$*$	Defasagem de um período

Sobrescritos

$\bullet$	Derivada temporal primeira
$\bar{}$	Valor médio

Siglas

AG	Algoritmo genético
MCP	McCulloch Pitts
RNA	Rede neural artificial
OGY	Referência a Ott, Grebory e York
EDO	Equação diferencial ordinária
CI	Condições iniciais
OPI	Órbita periódica instável
RPO	Realimentação proporcional ocasional

1- INTRODUÇÃO

1.1 - CONTEXTUALIZAÇÃO

Um sistema determinístico é considerado caótico sempre que sua evolução depende sensivelmente de suas condições iniciais. Esta propriedade implica que duas trajetórias emergindo de duas condições iniciais ligeiramente diferentes afastam-se exponencialmente no decorrer do tempo. Os requisitos necessários para um sistema determinístico ser caótico são que tal sistema deve ser não linear, e ser no mínimo tridimensional.

O fato de alguns modelos de sistemas dinâmicos que mostram estes requisitos possuírem uma dependência crítica em relação a suas condições iniciais é conhecido desde o fim do século XIX. Entretanto, somente nos últimos 30 anos, observações experimentais mostraram que, na verdade, sistemas caóticos são comuns na natureza. Eles podem ser encontrados, por exemplo, na química, na ótica não linear (lasers), na eletrônica, em dinâmica dos fluidos, e em muitas outras áreas de estudo. Muitos fenômenos naturais também podem ser caracterizados como caóticos tais como aqueles encontrados na meteorologia, sistema solar, coração e cérebro de organismos (Boccaletti *et al*, 2000).

O grande volume de propostas teóricas para o controle de sistemas caóticos tem estimulado diferentes aplicações nos sistemas experimentais cujo comportamento natural mostrou caos.

A estabilização de um sistema dinâmico em direção a um estado estacionário ou de periodicidade controlada permite várias aplicações tecnológicas possíveis, motivando então o interesse pela demonstração experimental da confiabilidade das diferentes técnicas teóricas, em situações práticas onde freqüentemente um modelo matemático para o sistema é desconhecido, ou características dinâmicas muito detalhadas são na prática impossíveis de ser extraídas das medidas.

Diferentes métodos foram desenvolvidos a partir dos anos 60 para solucionar os problemas no desenvolvimento de sistemas de controle. Tais métodos foram impulsionados pela crescente disponibilidade de computadores digitais, os quais apresentavam capacidades de processamento cada vez maiores.

Um desses métodos é utilizar Redes Neurais Artificiais (RNAs) e apresenta algumas características peculiares, como ser capaz de atingir bons resultados mesmo trabalhando com sistemas não-lineares e, por ser um tipo de sistema adaptativo, ter a capacidade de adequar-se ao sistema no qual está sendo aplicado, através de um treinamento iterativo.

Outro método similar às RNAs, no sentido de também se tratar de um sistema adaptativo, que pode ser utilizado em controle de sistemas dinâmicos são os Algoritmos Genéticos (AGs), os quais utilizam os princípios da evolução darwiniana para, de modo iterativo, procurar soluções para uma função objetivo em um espaço de busca desconhecido.

Juntos, RNAs e AGs, podem ser utilizados para o desenvolvimento de sistemas de controle ainda pouco estudados. Desse modo, esse trabalho teve como motivação a obtenção de como tal sistema de controle responde quando aplicado a um sistema instável e não-linear.

1.2 - DEFINIÇÃO DO PROBLEMA

Devido à dependência crítica às condições iniciais, e ao fato que, em geral, condições iniciais experimentais nunca são bem conhecidas, sistemas caóticos são intrinsecamente imprevisíveis. Certamente, a trajetória prevista a partir de condições iniciais exatas e a trajetória real vinda das condições iniciais medidas experimentalmente disponíveis divergem exponencialmente ao longo do tempo, então o erro da previsão, a distância entre as trajetórias prevista e real, cresce exponencialmente no tempo, até fazer a trajetória real do sistema completamente diferente da prevista, em períodos longos (Boccalleti *et al*, 2000).

Por muitos anos esta propriedade tornou o caos algo indesejável e a maioria dos pesquisadores considerou tal característica como algo a ser fortemente evitado. Junto à sensibilidade crítica às condições iniciais, sistemas caóticos exibem duas outras propriedades importantes. Primeiramente existe um número infinito de órbitas instáveis imersas no arranjo caótico. Em outras palavras, a estrutura de um atrator caótico é uma coleção de um número infinito de órbitas periódicas, instáveis individualmente. Além disso, a dinâmica em um atrator caótico é ergódica, o que implica que durante sua evolução no tempo o sistema visita ergodicamente pequenas vizinhanças de cada ponto em cada uma das órbitas periódicas instáveis intrínsecas ao atrator caótico.

Uma conseqüência relevante destas propriedades é que uma dinâmica caótica pode ser vista camuflando alguns comportamentos periódicos em um dado tempo, e erraticamente saltando de uma órbita periódica para outra. A idéia de controle do caos é aplicada quando uma trajetória aproxima-se ergodicamente de uma órbita periódica desejada, imersa no atrator, e são aplicadas pequenas perturbações para estabilizar tal órbita. Se for aplicada tal perturbação estabilizante, a trajetória move-se para a vizinhança da órbita periódica desejada, tornando possível sua estabilização. Este fato sugere que a sensibilidade crítica de um sistema caótico às mudanças, perturbações, em suas condições iniciais pode ser desejável em situações de experiências práticas. Certamente, se é verdade que uma perturbação, da ordem de 10^{-2}, nas condições iniciais pode resultar em uma resposta duas ordens de grandeza maior no decorrer do tempo, também é verdade que uma escolha acertada de uma perturbação no sistema dinâmico pode, em princípio, direcionar a trajetória para qualquer lugar que se queira no atrator, e produzir uma série de estados dinâmicos desejados e estáveis.

Um ponto importante é que, devido ao caos, está-se apto a produzir um número infinito de comportamentos dinâmicos desejados, tanto periódicos como não periódicos, usando o mesmo sistema caótico, com a única ajuda de pequenas perturbações escolhidas adequadamente. É válido enfatizar que este não é o caso para dinâmicas não caóticas, onde as perturbações aplicadas para produzir um comportamento desejado devem, geralmente, ser da mesma ordem de magnitude que as variáveis dinâmicas não perturbadas.

No início dos anos 90, as idéias para controlar o caos foram apontadas (Ott *et al*, 1990). A idéia principal consiste em esperar por uma passagem natural da trajetória caótica próximo à órbita periódica instável desejada, e então aplicar uma pequena perturbação ao sistema dinâmico, criteriosamente escolhida após um período de aprendizado, a fim de estabilizar tal dinâmica periódica. Através da construção de dinâmicas apropriadas, metas, compatíveis com o atrator caótico, um operador pode aplicar pequenas perturbações para produzir qualquer tipo de dinâmica desejada, mesmo não periódica, com aplicação prática no processo de codificação de sinais (Boccalleti *et al*, 2000).

Em algumas situações práticas, entretanto, pode ser desejável aplicar perturbações na variável de estado acessível ao operador. Isto sugere o desenvolvimento de algumas abordagens alternativas. Uma delas consiste em desenvolver uma linha de realimentação apropriada através da qual uma variável de estado é diretamente

perturbada para controlar, por exemplo, uma órbita periódica. Este segundo método requer a disponibilidade de uma variável de estado para observação experimental e para as perturbações. Nesse caso uma linha de realimentação negativa pode ser projetada de modo a ser proporcional à diferença entre o valor atual da variável de estado e o valor atrasado de um intervalo de tempo ΔT. A idéia é que, quando ΔT coincide com o período de uma órbita periódica instável do sistema não perturbado, a realimentação negativa leva a zero a diferença entre as dinâmicas atual e atrasada, então a órbita periódica é estabilizada. Além do mais, assim que o controle se torna efetivo, esta diferença vai efetivamente para zero, então a perturbação da realimentação desaparece. É importante lembrar que, como antes, um tempo preliminar de aprendizado é necessário (Boccalleti *et al*, 2000)

Um dos maiores problemas no processo anterior é que só é possível ligar o controle quando o sistema está suficientemente próximo do comportamento desejado. Isto é garantido pela ergodicidade do caos sem a interferência das condições iniciais escolhidas para a evolução caótica, mas pode acontecer da pequena vizinhança de um dado ponto atrator, alvo, ser visitada apenas raramente. Então a dinâmica não perturbada pode levar um longo tempo para se aproximar do alvo, resultando em um tempo de espera inaceitável para o operador aplicar o processo de controle do caos.

Métodos eficientes de direcionamento podem reduzir o tempo de espera em ordens de magnitude. Eles podem ser vistos como tarefas preliminares para o controle do caos, independente do algoritmo de controle particular que é aplicado.

A sensibilidade crítica às condições iniciais de um sistema caótico pode ser explorada não apenas para produzir um grande número de comportamentos periódicos possíveis, mas qualquer comportamento desejado compatível com a evolução natural do sistema. Este é o ponto a ser explorado neste trabalho, utilizar as características peculiares de sistemas caóticos em prol de melhorar a eficiência de sistemas de controle adaptativos, neste trabalho utilizando-se redes neurais artificiais e algoritmos genéticos.

1.3 - OBJETIVOS

Os objetivos deste projeto são estudar o controle de caos utilizando redes neurais artificiais e algoritmos genéticos, desenvolver um software capaz de controlar sistemas caóticos sobre órbitas desejadas e testar sua robustez a adversidades como ruído, variações nas condições iniciais e controle intermitente.

2 - REVISÃO BIBLIOGRÁFICA

Muitos artigos são direcionados à demonstração da aplicabilidade de noções e técnicas padrão de engenharia de controle para o controle do caos. A propósito, em muitos casos mesmo uma simples realimentação proporcional pode alcançar a meta de controle desejada. Por exemplo, o também chamado método de malha aberta e fechada (Jackson & Grosu, 1995) emprega uma combinação de realimentação proporcional com a chamada "ação direta de Hubler" que permite, em alguns casos, estabilização da trajetória desejada (Aquirre & Torres,2000).

Controle por pulsos proporcionais foi estudado por Casas & Grebori, 1997 e Chau, 1997. Realimentação proporcional no espaço estendido, ou seja, realimentação dinâmica, foi examinada por Magnitskii & Sidorov, 1998 e Zhao *et al*, 1998.

O potencial da realimentação dinâmica pode ser mais bem explorado usando uma visão do observador que permita o uso sistemático de realimentação. Uma análise de técnicas não lineares de observação pode ser encontrada em Nijmeijer & Mareels, 1997 e algumas particularidades podem ser econtradas em Grassi & Mascolo; Morgui & Solak, 1997.

Controle linear com ganho elevado e vista do observador para não linearidades de Lipschitz foi estudado por Liao, 1998.

Modelos de sistemas caóticos freqüentemente não satisfazem uma condição global de Lipschitz. Embora trajetórias de sistemas caóticos possuam fronteiras, este não é necessariamente o caso quando o sistema é influenciado pelo controle. Por outro lado a solução pode escapar em um tempo finito e não faz sentido discutir estabilidade e convergência. A possibilidade de escapar em sistemas não lineares controlados é freqüentemente ignorada em artigos para aplicação.

Alguns métodos são baseados em redução contínua de alguma função objetivo. O valor atual desta função pode refletir a distância entre o estado atual e o ponto da trajetória alvo. Para sistemas em tempo contínuo o valor da meta não depende diretamente do controle e abaixando o valor da velocidade pode ser colocado como uma meta imediata de controle ao invés de reduzir a função. Esta é a idéia básica do método de gradiente de velocidade (GV) (Fradkov, 1979), onde uma modificação no controle ocorre durante o gradiente (no controle) da velocidade da função alvo. Esta abordagem foi primeiramente usada para controlar sistemas caóticos (Fradkov, 1994). Exposição sistemática e referências posteriores podem ser encontradas em Fradkov & Pogromsky,

1998. Khovanov *et al*, 2000 propôs um algoritmo de controle de otimização energética do tipo GV para um oscilador periodicamente dirigido, movendo-o de um atrator caótico para um ciclo limite estável existente.

Uma direção produtiva é usar os métodos de domínio da freqüência para controle não linear (Basso *et al*, 1999). Em particular, métodos de aproximação de balanço harmônico para obtenção e predição de modos caóticos são usados em conjunto com teoria absoluta de estabilidade. Um método interessante dentro deste contexto emprega um filtro seletivo que atinge todos os sinais dentro de um intervalo estreito de freqüências (Meucci *et al*, 1998). Se tal filtro é incluído no laço de realimentação de um sistema caótico e a base de freqüência do filtro coincide com a freqüência de uma das soluções instáveis existentes é de se esperar que o sistema esteja em um movimento periódico do que caótico.

Em suma a maioria das abordagens de controle não linear pode ser agrupada em duas grandes classes: Abordagens de Lyapunov e abordagens de compensação.

É importante observar que genericamente, o controle baseado na alteração de alguns parâmetros de sistemas, como no esquema de OGY a ser visto no capítulo 3, é equivalente ao controle pela mudança de uma força adicional. Chen & Liu, 2002a, b mostraram para alguns sistemas caóticos que a equivalência entre realimentação linear e realimentação paramétrica pode ser estabelecida por uma transformação não linear adequada.

A análise de artigos publicados mostram que o bom uso do ferramental moderno de controle linear e não linear freqüentemente não leva em conta todos aspectos do movimento caótico. Isso significa usualmente que o princípio da pequena perturbação, pelo menos duas ordens de grandeza menor que a variável disponível para aplicação do controle, é violado. Freqüentemente, apenas modelos de baixa dimensão são considerados e pressupostos irreais são impostos. O uso apropriado da teoria de controle moderno para lidar com problemas realistas do controle do caos ainda é uma promessa.

Tendo em vista isso o presente trabalho tem como objetivo estudar o controle de caos utilizando redes neurais artificiais e algoritmos genéticos respeitando o uso de somente pequenas perturbações no intuito de levar uma trajetória caótica para um comportamento periódico.

3 - REVISÃO TEÓRICA

3.1 - REDES NEURAIS ARTIFICIAIS

3.1.1 Histórico

Os estudos de redes neurais artificiais datam da década de 40, mas têm evoluído rapidamente a partir de meados de 1980. Warrem McCulloch e Walter Pitts (1943 apud Braga; Ludemir; Carvalho, 2000) desenvolveram um modelo artificial de um neurônio. McCulloch era psiquiatra e neuroanatomista. Ele dedicou 20 anos em estudos visando representar um evento no sistema nervoso. Pitts era um matemático e optou por juntar-se a McCulloch em 1942. Em 1943, publicaram o trabalho "A logical calculus of the ideas immanent in nervous activity", no qual é feita uma análise de redes lógicas de nós, também chamados de nós McCulloch Pitts (MCP), e novas abordagens sobre elementos de decisão de limiar lineares, máquinas de estados finitos e modelos lógicos de formas de comportamento e memória.

O estudo de RNAs também aborda os métodos de aprendizado para que os nós consigam executar uma determinada função, porém o trabalho de McCulloch e Pitts tem como foco a descrição de um modelo artificial de um neurônio e a apresentação de suas capacidades computacionais.

Um dos primeiros trabalhos ligados diretamente ao aprendizado é de Donald Hebb (1949 apud Braga; Ludemir; Carvalho, 2000), o qual mostra como a variação dos pesos de entrada dos nós leva à plasticidade da aprendizagem de redes neurais. A teoria proposta por Hebb baseia-se no reforço das ligações sinápticas entre nós excitados. Essa teoria foi representada matematicamente e hoje é utilizada em vários algoritmos de aprendizado.

Outro importante nome no estudo de RNAs é Frank Rosemblatt. Esse demonstrou (1958 apud Braga; Ludemir; Carvalho, 2000), com seu modelo (o perceptron), que as RNAs com nós MCP, se acrescidas de sinapses ajustáveis, poderiam ser treinadas para classificar certos tipos de padrões. Rosemblatt conseguiu descrever uma topologia de RNA, estruturas de ligação entre os nós, e ainda propôs um algoritmo para treinar a rede para executar determinados tipos de funções.

O perceptron que Rosemblatt descreveu possui três camadas. A primeira, chamada de retina, recebe as entradas do exterior e possui conexões fixas; a segunda, conhecida como peso, recebe impulsos da primeira através de conexões cuja eficiência

de transmissão é ajustável e envia saídas para a terceira camada, denominada resposta.

O perceptron de Rosemblatt tem o comportamento de um classificador de padrões, o qual divide o espaço de entrada em regiões distintas para cada uma das classes existentes. O perceptron só consegue classificar padrões linearmente separáveis. Inicialmente verifica-se que a saída da rede é aleatória, porém ao ajustar gradualmente os pesos (segunda camada), o perceptron passa por um processo de treinamento que o faz fornecer saídas concordantes com os dados oferecidos no conjunto de treinamento.

Apesar de Minsky e Papert comprovarem (1969 apud Braga; Ludemir; Carvalho, 2000) as limitações das RNAs de classificar somente padrões linearmente separáveis, tal problema foi superado em 1982, quando os avanços tecnológicos na microinformática tornaram viáveis projetos de RNAs com múltiplas camadas. Parte do mérito dessa retomada se deve a John Hopfield o qual publicou um artigo que destacou as propriedades associativas das RNAs, mostrando a relação entre redes recorrentes auto-associativas e sistemas físicos.

A analogia com o cérebro humano utilizada no estudo de RNAs faz necessário um estudo mais detalhado das similaridades existentes entre os sistemas natural e artificial. A célula fundamental do cérebro é o neurônio. Cada neurônio comunica-se com milhares de outros de forma contínua e paralela. A capacidade humana de exercer funções motoras, de pensar, sentir emoções ou mudanças no ambiente é possibilitada pelo cérebro.

A rede cerebral é capaz de identificar padrões e usá-los de forma a descobrir possíveis relações existentes entre eles, manipular conhecimentos adquiridos por experiência e interpretar situações. Mesmo que o funcionamento do cérebro seja, ainda, não completamente desvendado, sua estrutura funcional, rede de nós, é conhecida e é a base para o estudo das RNAs.

As RNAs têm por objetivo reproduzir as funções das redes biológicas, visando obter seu comportamento básico e sua dinâmica. Mesmo diferindo bastante fisicamente das redes naturais, as redes artificiais possuem similaridades com as redes naturais que nos ajudam a entender o sistema nervoso. Ambas baseiam-se em unidades paralelas e distribuídas de computação. Essas unidades comunicam-se por conexões sinápticas. RNAs possuem modularização das conexões, detectores de características e redundância. Tais características tornam possível às RNAs reproduzir várias funções humanas, fazendo crer que o futuro da neurocomputação está diretamente ligado ao desenvolvimento de modelos com apelo biológico.

A idéia do funcionamento do processo de condução e da estrutura do neurônio biológico é relativamente simples. Divididos em três partes, corpo celular, axônio e dendritos, cada qual com funções específicas e complementares entre si, os neurônios funcionam da seguinte maneira: os impulsos vindos de outros neurônios são recebidos pelos dendritos e conduzidos ao corpo celular, onde tais impulsos são interpretados e retransmitidos, através de outro pulso nervoso, a outros neurônios, partindo do axônio da célula em que se encontra até os dendritos das próximas células, neurônios, passando pela sinapse.

As sinapses são pontos de união funcional dos neurônios para formar as redes neurais. Sua função é controlar a transmissão de informações, impulsos, entre os nós da rede. Por possuir um efeito variável, as sinapses fornecem aos neurônios o poder de adaptação. Os pulsos vindos dos nós pré-sinapticos chegam ao corpo celular do nó pós-sinaptico e são comparados com os demais pulsos recebidos pelo mesmo. Se é atingido um valor maior ou igual a um limiar pré-definido em um intervalo de tempo, o nó "dispara", transmitindo para os nós seguintes, pós-sinapticos, um impulso.

Esse é o princípio de funcionamento de grande parte das funções realizadas pelo cérebro. Operando em paralelo, os 10^{11} (dez elevado à décima primeira potência) nós do cérebro humano são capazes de realizar tarefas incrivelmente complexas.

3.1.2 Definição

Redes neurais artificiais (RNAs) são modelos matemáticos que se assemelham às estruturas neurais biológicas e que tem capacidade computacional adquirida por meio de aprendizado e generalização (Braga; Ludemir; Carvalho, 2000).

As RNAs são sistemas paralelos distribuídos compostos por unidades de processamento simples. Estas unidades, denominadas nós, neurônios ou nodos, são organizados em redes e são capazes de avaliar determinadas funções matemáticas lineares e não-lineares.

Os nós são dispostos em uma ou mais camadas interligadas por um grande número de conexões, quase sempre unidirecionais. Essas conexões geralmente estão associadas a pesos, os quais armazenam o conhecimento representado no modelo e têm o papel de ponderar a entrada recebida por cada neurônio da rede.

3.1.3 - Funcionamento

A arquitetura das RNAs e o modo como elas representam internamente classes de problemas possibilitam um desempenho superior ao de modelos que utilizam derivadas em seu método de solução. Nas RNAs, o procedimento seguido para solucionar problemas é composto inicialmente por uma fase de aprendizagem, na qual vários exemplos são apresentados à rede.

Assim a rede, através de treinamento adequado, é capaz de perceber e capturar características que a permitem representar os dados recebidos, informações. Ponto que evidencia a capacidade de aprendizado utilizando exemplos juntamente com o poder de generalizar a informação aprendida.

As RNAs conseguem abstrair informações não explícitas por meio dos exemplos e avaliam funções multivariáveis, com a vantagem de gerar um menor custo computacional que cresce linearmente com o número de variáveis, em relação a outros métodos que crescem com ordens maiores. A aplicabilidade de RNAs na solução de problemas complexos é viabilizada, entre outras coisas, por suas capacidades de processamento temporal e de auto-organização.

Inspirados no conhecimento científico do neurônio biológico, McCulloch e Pitts propuseram o modelo chamado MCP. Esse modelo é formado por "n" terminais de entrada "$x_1, x_2, ..., x_n$", representando os dendritos, e um terminal de saída y, representando o axônio.

As entradas "$x_1, x_2, ..., x_n$" possuem pesos "$w_1, w_2, ..., w_n$", os quais podem ser tanto positivos quanto negativos, contribuição excitatória ou inibitória, respectivamente e tem por função mensurar a importância da contribuição de uma determinada conexão para um neurônio específico, emulando o comportamento das sinapses biológicas. A contribuição de uma sinapse particular "i" no nó é calculada pelo produto "$x_i w_i$".

Assim como no sistema biológico, onde um neurônio dispara quando a soma dos impulsos que ele recebe excede seu limiar de excitação, no modelo computacional o corpo celular do neurônio é abstraído por um somador dos valores de "$x_i w_i$" recebidos pelo nó e constata se o mesmo deve ou não disparar, y = 1 ou y = 0, respectivamente, comparando a soma obtida ao limiar de ativação do neurônio, geralmente expresso por uma "função de ativação". A figura 1 mostra um diagrama representativo de um neurônio MCP, suas entradas, pesos e saída.

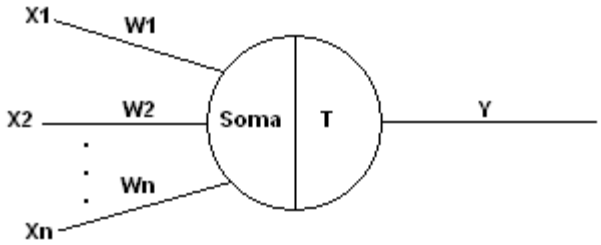

Figura 1: Neurônio MCP, onde T é o limiar de ativação do neurônio.

A arquitetura de uma RNA deve ser definida de forma cautelosa, pois é ela que restringe o tipo de problema que pode ser tratado por esta RNA. Por exemplo, redes com uma camada única de nós MCP só conseguem resolver problemas linearmente separáveis, redes cíclicas adéquam-se melhor à solução de problemas que envolvem processamento temporal.

A topologia da rede, seu número de camadas, o tipo de conexão existente entre seus nós e o número de nós em cada camada são parâmetros da definição da arquitetura da rede. RNAs podem ser, em relação ao número de camadas, de camada única ou de múltiplas camadas; quanto ao tipo de conexão entre os nodos podem ser acíclicas (*feedforward*) ou cíclicas (*feedback*); em relação à sua conectividade podem ser fracamente (parcialmente) ou completamente conectadas.

A capacidade de aprendizado das redes neurais artificiais é de fundamental importância para o correto funcionamento das mesmas. As RNAs aprendem utilizando exemplos e são capazes de fazer interpolações e extrapolações do que aprenderam. Um algoritmo de aprendizado é um conjunto de procedimentos com função de adaptar os parâmetros de uma RNA, treinando-a para um determinado fim, como controlar um sistema caótico.

A etapa de aprendizagem é um processo iterativo que tem por objetivo ajustar os parâmetros da RNA, ou seja, os pesos "$w_1, w_2, ..., w_n$" das conexões que guardam o conhecimento que a rede obteve de seu ambiente de operação.

Os métodos para treinamento de redes são classificados em aprendizado supervisionado e aprendizado não supervisionado.

O aprendizado supervisionado é o método mais comum no treinamento de redes neurais. Nesse método, a entrada e a saída desejadas para a RNA são fornecidas por uma entidade exterior, chamada de supervisor ou professor. Nesse tipo de aprendizado busca-se manipular os parâmetros da rede de forma a obter uma relação entre pares de

entradas e saídas fornecidos. A saída da RNA é constantemente calculada e comparada com o resultado desejado, recebendo do supervisor dados de erro corrente, o que possibilita modificar os pesos das conexões a fim de tornar tal erro cada vez menor. Um bom critério de desempenho é a soma dos erros quadráticos de todas as saídas, que pode ser interpretada como uma função a ser minimizada pelo processo de treinamento. Um ponto fraco na abordagem de aprendizado supervisionado é a sua forte dependência de um supervisor, sem o qual a RNA não é capaz de aprender estratégias que solucionem situações não consideradas nos exemplos do treinamento da RNA. A figura 2, mostra um diagrama do treinamento supervisionado.

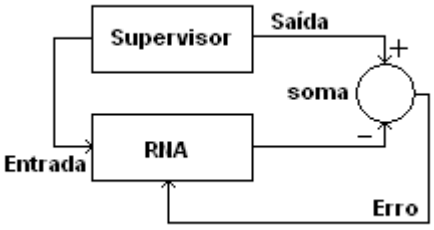

Figura 2: Aprendizado supervisionado.

No aprendizado não-supervisionado, diferentemente do aprendizado supervisionado, não existe uma entidade supervisora responsável por acompanhar o processo de aprendizado. Pelo contrário, esses algoritmos utilizam apenas os padrões de entrada que estão disponíveis para a RNA com o objetivo de alcançar uma relação harmônica com as regularidades estatísticas da entrada de dados e, quando bem sucedida, desenvolve a habilidade de representá-las internamente a fim de codificar características da entrada e criar novas classes ou grupos de forma automática, ou seja, identificar, através de exemplos, padrões na entrada de dados, e através destes expandir sua capacidade de solução para situações não apresentadas à rede anteriormente. Se não se verifica padrões nas informações fornecidas na entrada, este tipo de aprendizado torna-se impossível, pois não se conseguiria obter quaisquer características ou padrões dos dados de entrada.

3.2 - ALGORITMOS GENÉTICOS

3.2.1 - Introdução

Algoritmos genéticos (AGs) é um método de busca baseados nos mecanismos da evolução Darwiniana: seleção, recombinação genética, *crossover* ou cruzamento, e

mutação. Eles utilizam a idéia de sobrevivência do mais apto, por algum critério, através de formações vetoriais bem estruturadas, strings, as quais promovem trocas de informação através de operadores matemáticos.

Em cada geração, um novo conjunto de "indivíduos", vetores, é criado usando partes dos indivíduos mais aptos da geração anterior. Ocasionalmente, ocorrem mutações que geram novos indivíduos, sem ligação com as gerações anteriores, a fim de testar possíveis soluções fora da região inicial de busca. Geração após geração, os procedimentos tendem a levar a uma solução ótima, explorando as informações obtidas anteriormente para escolher novos pontos de análise com os quais se espera atingir resultados melhores que os correntes.

3.2.2 - Origem

Algoritmos genéticos (AGs) foram desenvolvidos por John Holand e seus colaboradores na Universidade de Michigan nos anos 70 (Goldberg, 1989). As metas de sua pesquisa desdobraram-se em abstrair e explicar rigorosamente os processos adaptativos dos sistemas naturais e projetar softwares de sistemas artificiais que conservassem os mecanismos dos sistemas naturais. Essa abordagem tem conduzido a importantes descobertas em ambos os sistemas, naturais e artificiais.

O tema central de pesquisa em algoritmos genéticos tem sido a robustez e o equilíbrio entre eficiência e eficácia necessárias para solução de diferentes classes de problemas. AGs são usados para representar sistemas adaptativos, ou seja, capazes de aprender com exemplos e adaptar-se de forma a extrapolar suas conclusões sobre o sistema analisado para situações não abordadas durante o treinamento. Esta é a vantagem peculiar nos sistemas adaptativos, e faz com que essa área de estudo continue ganhando força historicamente. São várias as implicações da robustez para sistemas artificiais, por exemplo, os gastos com alterações de projeto podem ser minimizados. Se níveis maiores de adaptação podem ser alcançados, sistemas já existentes podem desempenhar suas funções por mais tempo e com melhor desempenho.

3.2.3 - Funcionamento

Algoritmos genéticos fornecem um robusto modo de busca. Pesquisas consolidam a validade da técnica na otimização de funções e aplicações de controle, o que tem estendido amplamente suas aplicações nas mais diferentes áreas de estudo

(Goldberg, 1989).

O funcionamento dos algoritmos genéticos é surpreendentemente simples, envolvendo nada mais complexo que a cópia de seqüências de caracteres, cromossomos, e a troca de partes das mesmas, genes.

Em qualquer campo de investigações o algoritmo genético parte de uma população inicial aleatória, a qual é composta por indivíduos representados por seqüências, vetores, de elementos codificados segundo uma determinada base como binária, octal, decimal, hexadecimal e alfabética.

O primeiro passo é mensurar o quanto essa população gerada se aproxima do desempenho desejado chamado de aptidão ou *fitness*, geralmente determinado por uma função objetivo. Pode-se fazer uma analogia com uma caixa preta, para a qual fornecemos os dados de entrada e obtemos a saída, sem, no entanto, saber o que se passa no interior da caixa. Por exemplo, se queremos maximizar uma função desconhecida o algoritmo genético vai buscar na população inicial os indivíduos que proporcionam valores mais elevados para a função objetivo (Koza, 2003).

Os indivíduos usados pelo AG são submetidos basicamente a três operações: seleção, cruzamento e mutação.

Seleção é o processo no qual indivíduos isolados são copiados de acordo com seus valores de aptidão, ou seja, seu grau de adequação à função objetivo em relação aos demais indivíduos, o que indica que tal indivíduo pode contribuir para se obter um resultado melhor em uma geração subseqüente, e por isso deve ser mantido para futuros testes e processos. Esse operador simula artificialmente a seleção natural Darwiniana, a sobrevivência dos indivíduos mais adaptados dentro de uma população, que pode ser desenvolvida de várias maneiras, porém a mais simples delas é criar uma roleta ponderada, na qual cada indivíduo da população ocupa uma fatia de tamanho proporcional ao seu valor de aptidão.

Para efetuar o processo de seleção basta girar a roleta ponderada tantas vezes quanto for o número de indivíduos. Dessa maneira quanto maior a aptidão do indivíduo maior será a probabilidade de ele participar do processo de cruzamento, e conseqüentemente, de contribuir para a próxima geração. Uma vez que um indivíduo tenha sido selecionado para a operação de reprodução, uma cópia exata do mesmo é feita e colocada no que é chamada de “bacia de cruzamento”, os indivíduos selecionados na reprodução serão submetidos aos posteriores operadores genéticos, cruzamento e mutação (Linden, 2006).

O processo de cruzamento, *crossover* ou recombinação genética é realizado em dois passos. Primeiramente divide-se, aleatoriamente, a população existente em pares. Para cada par escolhe-se, também aleatoriamente, em que posição do indivíduo, cromossomo, ocorrerá o cruzamento. Assim dois novos indivíduos são criados a partir da troca de partes dos indivíduos pais.

O fato é que a reprodução combinada à troca de informação do cruzamento fornece ao algoritmo genético muito de seu poder de busca. Parte da explicação para isso está na idéia de justaposição de diferente meios de busca e de otimização.

No caso dos algoritmos genéticos, uma mistura de direcionamento e acaso gera uma maneira eficiente de construir novas soluções a partir das melhores soluções preliminares obtidas nas gerações anteriores. Algoritmos genéticos exploram uma quantidade enorme de informação reproduzindo o que é denominado esquemas ou noções de alta qualidade, segundo suas performances e, cruzando essas noções com muitas outras, também de alto desempenho, de outros indivíduos, caminha-se na direção de soluções cada vez melhores (Linden, 2006). Para que seja possível um maior entendimento do funcionamento dos algoritmos genéticos, o apêndice B traz um aprofundamento na teoria de esquemas. Então, a ação do cruzamento com prévia seleção de indivíduos na reprodução produz novas idéias, indivíduos filhos, construídas a partir de esquemas, "blocos", de alto desempenho das gerações anteriores. A figura 3 mostra um diagrama representativo do cruzamento.

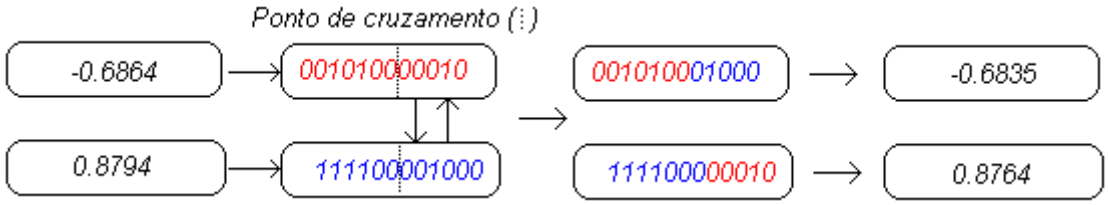

Figura 3: Representação de um processo de cruzamento.

Mutação é um importante operador dos algoritmos genéticos. Ela é necessária porque mesmo que a seleção e o cruzamento efetivamente busquem e recombinem uma vasta quantidade de esquemas, proporcionando a sobrevivência dos esquemas de melhor aptidão, ocasionalmente, eles podem tornar-se viesados e perder informações genéticas potencialmente úteis, ou seja, se o esquema que contém a solução ótima não estiver presente na população inicial, ocorrerá convergência prematura, ou local. Em sistemas genéticos artificiais, o operador de mutação tem como função proteger o sistema contra

tais perdas irrecuperáveis, introduzindo aleatoriamente e esporadicamente, durante a evolução, novos esquemas que sobreviverão, ou não, segundo a sua aptidão (Goldberg, 1989). Em um algoritmo genético simples, mutação caracteriza-se como uma alteração randômica ocasional do valor de uma posição, gene, da string, cromossomo ou indivíduo. A figura 4 mostra um diagrama de um processo de mutação.

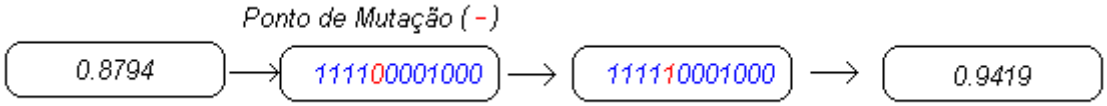

Figura 4: Representação de um processo de mutação.

A dependência entre indivíduos com altos valores de aptidão e as coincidências, similaridades, encontradas entre suas strings, representações, são pontos importantes a serem tratados no estudo de AGs. As informações subjacentes às seqüências determinam sua adequação à função objetivo. Isso leva à análise de um conjunto de "sub strings" com pontos em comum que as coloca em um patamar de adequação superior ou inferior ao das demais sub strings.

Na figura 5, está a representação de um esquema com dois indivíduos.

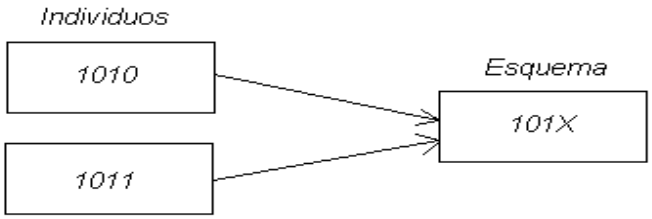

Figura 5: Representação de esquema

O conjunto de seqüências que pode ser representado por essa similaridade é chamado de esquema. Dessa forma analisam-se não mais indivíduos isolados, visto que similaridades importantes entre indivíduos com alto valor de aptidão podem ajudar a guiar o processo de busca. Um esquema contém um indivíduo isolado se uma das variações possíveis das posições livres, genes, do esquema faz com que o mesmo reproduza exatamente a string que representa o indivíduo em questão.

Se as strings forem consideradas separadamente teremos apenas "n" informações, onde "n" é o número de indivíduos da população vigente. Entretanto quando são consideradas as strings, seus valores de aptidão e as similaridades entre as strings na população, admite-se uma grande quantidade de novas informações para

ajudar a direcionar a busca (Linden, 2006).

O efeito da reprodução em um esquema em particular é determinado segundo a idéia de que quanto maior a aptidão dos indivíduos, maior a probabilidade dos mesmos serem selecionados. Em geral, atribui-se um número crescente de amostras para os melhores indivíduos observados. Entretanto a reprodução por si só não cria novos pontos no espaço de esquemas, apenas preserva os melhores.

Mutação, normalmente em taxas baixas de ocorrência, não destrói um esquema em particular, visto que a mutação, quando ocorre, muda apenas um gene do indivíduo, e um esquema é formado por um conjunto de indivíduos com alguns genes em comum. Isso leva a uma conclusão: altos valores de aptidão em conjunto com a definição curta de esquemas, mais gerais, contendo várias strings, são propagados geração após geração aumentando rapidamente o número de amostras dos melhores indivíduos observados.

As razões para a crescente difusão do uso de algoritmos genéticos são sua simplicidade computacional contrastante com sua poderosa técnica de busca e o fato de não ser limitada por suposições restritivas sobre o espaço de busca, envolvendo continuidade, existência de derivadas e o conhecimento prévio da forma matemática da função objetivo (Linden, 2006).

Algumas diferenças básicas em relação aos métodos convencionais de busca e otimização que fazem dos AGs algo tão simples e funcional, são:

1 – Trabalham com a codificação de um conjunto de parâmetros e não com os próprios parâmetros;

2 – Buscam soluções em uma população de pontos e não em pontos isolados. Com isso o AG “escala” vários picos simultaneamente, em paralelo. Conseqüentemente, a probabilidade de considerar um extremo local como um extremo global é reduzida;

3 – Usam informações da função objetivo, e não de derivadas ou outros dados auxiliares;

4 – Usam regras probabilísticas de transição. Regras determinísticas são passíveis de localizar falsos extremos, picos, de funções.

Avaliadas juntas, essas quatro características contribuem para a robustez do algoritmo genético, o que proporciona uma vantagem considerável em relação a outras técnicas de busca e otimização comumente usadas, como o método de hill-climbing e o método de Newton-Raphson.

Nesse trabalho os AGs serão utilizados para selecionar os neurônios da rede neural artificial entre ciclos de execução do programa principal, proporcionando ao

método uma maior capacidade de adaptação. Espera-se que o par RNA e AG formem um sistema de resolução de problemas robusto e confiável. Detalhes do desenvolvimento e resultados estão dispostos em capítulos posteriores.

3.3 - O MÉTODO OGY DE CONTROLAR O CAOS

O caos ocorre em uma grande variedade de processos naturais e também pode ocorrer porque se deseja desenvolver um experimento que apresente um comportamento caótico. A idéia de Ott, Grebogi e Yorke (OGY) é que o caos pode ser desejável desde que este possa ser controlado usando uma pequena perturbação sobre algum parâmetro acessível ou sobre alguma variável dinâmica do sistema (FIEDLER-FERRARA; PRADO,1994).

O ponto principal para obter o controle de caos é observar que um arranjo caótico, onde a trajetória do processo caótico vive, tem imergido em si um grande número de órbitas periódicas. Somado a isso, devido à ergodicidade, a trajetória acessa a vizinhança de cada uma dessas órbitas periódicas. Algumas dessas órbitas periódicas podem corresponder a um desempenho desejado de um sistema de acordo com algum critério. O segundo ponto importante é a percepção que o caos, por apresentar dependência sensível a pequenas alterações no estado atual, causa imprevisibilidade do estado do sistema em tempos longos. Isso implica que o comportamento do sistema que apresenta comportamento caótico pode ser alterado pela utilização de pequenas perturbações. Então, o acesso das trajetórias dos sistemas caóticos a várias órbitas periódicas diferentes, combinado com sua sensibilidade a pequenas perturbações, permite o controle e manipulação do processo caótico. Especificamente, a abordagem OGY é da seguinte forma: Primeiro determinam-se algumas das órbitas periódicas instáveis que estão amarradas ao arranjo caótico. Então, examinam-se a localização e a estabilidade destas órbitas e escolhe-se uma que produz o desempenho desejado do sistema. Finalmente, aplica-se um pequeno controle para estabilizar esta órbita periódica desejada. Tudo isso pode ser feito a partir dos dados do sistema, usando análise não-linear de séries temporais para a observação, compreensão e controle do sistema. Isto é particularmente importante uma vez que sistemas caóticos são bastante complicados e o conhecimento detalhado das equações do processo é freqüentemente ausente.

A idéia básica do controle do caos pode ser entendida considerando o seguinte mapa logístico unidimensional:

$$X_{n+1} = f(X_n, r) = rX_n(1 - X_n), \qquad \text{Eq.(1)}$$

Onde X é restrito ao intervalo unitário [0,1], e r é um parâmetro de controle. É sabido que este mapa desenvolve o caos via bifurcação por duplicação de período. Para $0 < r < 1$, o estado assintótico do mapa, ou atrator do mapa, é $X = 0$; para $1 < r < 3$, o atrator é um ponto fixo diferente de zero $X_f = 1 - \frac{1}{r}$; para $3 < r < 1 + \sqrt{6}$, este ponto fixo é instável e o atrator é uma órbita periódica de período 2. Como r é incrementado depois, uma seqüência de bifurcações com duplicação do período ocorre, na qual sucessivas órbitas de período dobrado tornam-se estáveis. A cascata de duplicação de período acumula em $r = r_\infty \approx 3.57$, depois do qual o caos pode acontecer.

Considere o caso $r = 3.8$ para o qual o sistema é aparentemente caótico. Uma importante característica do atrator caótico é que nele existe um número infinito de órbitas periódicas instáveis imersas. Por exemplo, existe um ponto fixo $X_f \approx 0.7368$ e uma órbita de período 2 com componentes $X(1) \approx 0.3737$ e $X(2) \approx 0.8894$, onde $X(1) = f(X(2))$ e $X(2) = f(X(1))$.

Agora suponha que se deseja evitar caos em $r = 3.8$. Em particular, procuram-se trajetórias resultantes de condições iniciais X_0 escolhidas randomicamente que sejam tão perto quanto possível da órbita de período 2, assumindo que esta órbita de período 2 fornece a melhor performance do sistema. Suponha que o parâmetro r pode ser sintonizado de forma fina em um pequeno intervalo em torno do valor $r_0 = 3.8$, isto é, r pode variar no intervalo $[r_0 - d, r_0 + d]$, onde $d \ll 1$. Devido a natureza do atrator caótico, uma trajetória que parte de um valor arbitrário de X_0 cairá, com probabilidade 1, na vizinhança da órbita de período 2 desejada em algum momento posterior. A trajetória deve divergir rapidamente da órbita de período 2 se não houver interferência. A tarefa aqui é programar a variação do parâmetro de controle de forma que a trajetória permaneça na vizinhança da órbita de período 2 enquanto o controle estiver presente. Em geral, as pequenas perturbações no parâmetro serão dependentes do tempo. Enfatiza-se que é importante aplicar apenas pequenas perturbações no parâmetro, então obviamente torna-se possível eliminar o caos variando r de 3.8 para 2.0, por exemplo, mas mudanças desta magnitude não são interessantes.

A estratégia para controlar a órbita é muito flexível para estabilizar órbitas periódicas diferentes em tempos diferentes. Suponha que primeiramente estabiliza-se uma trajetória caótica em volta de uma órbita de período 2. Então deseja-se estabilizar o ponto fixo do mapa logístico, assumindo que o ponto fixo deve corresponder a um desempenho melhor do sistema para um instante posterior. Para conseguir esta mudança de controle, simplesmente é desligado o parâmetro de controle com respeito à órbita de período 2. Sem controle, a trajetória divergirá da órbita de período 2 exponencialmente. Deixa-se o sistema evoluir no valor do parâmetro r_0. Devido a natureza do caos, depois de certo tempo, a trajetória caótica entra em uma pequena vizinhança do ponto fixo. Neste momento é ligado um novo conjunto de perturbações de parâmetros, calculadas com respeito ao ponto fixo. A trajetória pode então ser estabilizada em torno do ponto fixo.

Na presença de ruído externo, uma trajetória controlada vai, ocasionalmente, ser expulsa da vizinhança da órbita periódica. Se esse comportamento acontecer, desliga-se a perturbação do parâmetro e deixa-se o sistema evoluir por si só. Com probabilidade de 100% a trajetória caótica vai entrar na vizinhança da órbita periódica alvo a ser controlada novamente. O efeito do ruído é transformar uma trajetória periódica controlada em uma intermitente, na qual fases caóticas (trajetórias descontroladas) são alternadas com fases laminares (trajetórias periódicas controladas). É fácil verificar que o comprimento médio das fases laminares aumenta quando a amplitude do ruído diminui.

É interessante perguntar quantas iterações são necessárias, em média, para uma trajetória caótica originada de condições iniciais arbitrárias entrar na vizinhança ε de uma órbita periódica alvo. Claramente, quanto menor o valor de ε, maior o número de iterações são necessárias.

A maior vantagem da idéia de controlar o caos é que isto pode ser aplicado a sistemas experimentais, nos quais um conhecimento prévio do sistema não está disponível. Uma série temporal encontrada através da medição de uma das variáveis dinâmicas em conjunto com o método de empregar um atraso temporal, a qual transforma uma série temporal escalar em uma trajetória no espaço de fases, é suficiente para determinar órbitas instáveis periódicas desejadas a serem controladas e as quantidades relevantes requeridas para computar as perturbações dos parâmetros. Outra vantagem do paradigma de controle do caos OGY é a flexibilidade na escolha de órbitas periódicas a serem controladas.

Então, em uma situação ideal, sem ruídos e sem nenhuma imperfeição na identificação nos parâmetros do sistema, aplicar controle cria uma órbita estável, mas para uma condição inicial típica, isto é precedido no tempo por um transiente caótico, no qual a órbita é similar a órbitas no atrator caótico descontrolado. Os comprimentos dos transientes caóticos são diferentes para diferentes condições iniciais, e eles podem ser vistos como realizações de uma variável randômica τ com uma distribuição de probabilidade exponencial:

$$P(\tau) \sim e^{-\frac{\tau}{\langle\tau\rangle}} \qquad \text{Eq.(2)}$$

Onde $\langle\tau\rangle$ é o comprimento médio do transiente caótico. Quando ε decresce, o tamanho da faixa de controle decresce então o tempo médio para alcançar o controle, ou $\langle\tau\rangle$, cresce.

A mensuração do tempo médio para atingir o controle é obtida sob a suposição que o controle está desligado para raios maiores que o estipulado para o controle agir.

Ao adicionar ruído nas medidas do comportamento que se deseja controlar têm-se duas situações possíveis. Se o ruído for limitado, ou seja, se sua amplitude não ultrapassar o raio de controle da órbita, dificilmente o controle será afetado por tal ruído, onde raio de controle é a região do espaço que compreende a vizinhança da órbita periódica desejada. Por outro lado se o ruído for ilimitado há a possibilidade de este desviar uma trajetória controlada para fora de seu raio de controle.

O uso de coordenadas de atraso pode ser bastante útil no controle do caos, pois na maioria das situações experimentais não se possui um conhecimento detalhado das equações do sistema. Geralmente mede-se uma série temporal de uma única variável de estado e então se usa o atraso de coordenadas para representar o estado do sistema.

Na presença da variação do parâmetro, coordenadas atrasadas tendem para um mapa de formas diferentes da Eq.(1).

O método OGY aplica-se para mapas inversíveis. Em geral, sistemas dinâmicos que podem ser descritos por um conjunto autônomo de equações diferenciais de primeira ordem são inversíveis, e o sistema inverso é obtido deixando $t \to -t$ no conjunto original de equações diferenciais. A maioria dos sistemas dinâmicos encontrados na prática cai nesta categoria. Sistemas dinâmicos não inversíveis possuem

propriedades diferentes das encontradas em sistemas dinâmicos inversíveis. Por exemplo, para mapas não inversíveis bidimensionais, um ponto no atrator caótico pode não ter uma única direção estável (instável). Um método para determinar todas essas direções estáveis e instáveis não é conhecido. Se uma ou muitas direções como essas podem ser calculadas, o método OGY pode em princípio ser aplicado a sistemas não inversíveis forçando uma trajetória caótica a cair em uma das direções estáveis da órbita periódica.

A fase transiente, onde a órbita vaga caoticamente antes de focar em uma órbita controlada, pode ser bem encurtada aplicando uma técnica de direcionamento, então a trajetória pode ser trazida rapidamente para uma região alvo no atrator usando pequenas perturbações de controle. A idéia é que, já que sistemas caóticos são exponencialmente sensíveis a perturbações, a escolha cuidadosa de pequenas perturbações de controle pode, depois de algum tempo, ter um grande efeito na locação da trajetória e pode ser usada para guiá-la. Então o tempo para atingir o controle pode, a princípio, ser bem reduzido pela aplicação apropriada de pequenos controles quando a órbita está longe da vizinhança da órbita periódica desejada.

Tem-se considerado o caso onde existe apenas um parâmetro de controle simples disponível para ajuste. Enquanto genericamente um único parâmetro é suficiente para estabilização da órbita periódica desejada, podem existir algumas vantagens em utilizar muitas variáveis de controle. Por isso o único parâmetro de controle p torna-se um vetor. Em particular, a liberdade adicionada em ter muitos parâmetros de controle deve permitir melhores formas de escolher o controle, e então diminuir o tempo para alcançar o controle, da mesma forma para os efeitos do ruído.

Enfatiza-se que o completo conhecimento da dinâmica do sistema não é necessário para aplicar a idéia OGY. Em particular, precisa-se da locação da órbita periódica desejada, a dinâmica linearizada sobre a órbita periódica, e a dependência da locação da órbita periódica em relação a pequenas variações do parâmetro de controle. Coordenadas atrasadas fixas têm sido utilizadas com sucesso em estudos experimentais para extrair tais informações puramente de observações de órbitas caóticas experimentais no atrator sem qualquer conhecimento prévio das equações do sistema, e tal informação tem sido utilizada para controlar órbitas periódicas.

A idéia OGY de controle do caos oferece flexibilidade. Mudando o controle, pode-se mudar o comportamento assintótico temporal de uma órbita periódica para outra. Em algumas situações, onde a flexibilidade oferecida pela habilidade de fazer tal

mudança é desejável, pode ser vantajoso desenvolver um sistema para que o mesmo seja caótico. Em outras situações, onde se apresenta um sistema caótico, o método pode permitir a eliminação do caos e alcançar uma melhora significativa a um custo relativamente baixo.

3.4 - O MÉTODO ADAPTATIVO PARA O CONTROLE DO CAOS

Abordagens alternativas para o método OGY têm sido propostas para estabilização das órbitas periódicas instáveis (OPI) de uma dinâmica caótica. Em geral as estratégias para controlar o caos podem ser classificadas em duas classes principais, nomeadas: métodos de malha fechada e métodos de malha aberta.

A primeira classe inclui métodos que selecionam a perturbação baseado em um conhecimento do estado do sistema, orientados para controlar uma dinâmica prescrita. Entre eles, estão, conjuntamente ao método OGY, a chamada realimentação proporcional ocasional (RPO), o método de Huebler, e o método de Pyragas, o qual aplica uma realimentação atrasada em uma das variáveis do sistema. Todos esses métodos são independentes do modelo, no sentido que o conhecimento do sistema necessário para selecionar a perturbação pode ser adquirido pela simples observação do sistema durante um tempo de aprendizado adequado. A segunda classe inclui aquelas estratégias que consideram o efeito de perturbações externas, independente do conhecimento do estado atual da dinâmica, na evolução do sistema. Aquelas, periódicas ou eventuais, têm sido utilizadas para produzir mudanças na dinâmica dos sistemas caóticos, tendendo eventualmente à estabilidade algum comportamento periódico. Estas abordagens, entretanto, são em geral limitadas pelo fato que suas ações não são orientadas a uma meta, isto é, o estado periódico final não pode ser decidido pelo operador (Boccaletti *et al*, 2000).

3.4.1 - O Direcionamento Adaptativo do Caos

Direcionamento do caos significa perturbar um sistema caótico com a meta de atingir a órbita emergente de um dado ponto para uma vizinhança de algum outro ponto escolhido, chamado alvo, no atrator dentro de um tempo finito específico, chamado tempo alvo. Como já mencionado, mesmo que a ergodicidade garanta que todos os pontos no atrator são visitados sem levar em conta as condições iniciais escolhidas para

a evolução caótica, em muitos casos uma pequena vizinhança de um dado ponto do atrator pode ser visitada esporadicamente; então a dinâmica não perturbada pode levar um longo tempo para abordar um dado alvo. Daí a necessidade de desenvolver métodos eficientes de direcionamento, os quais podem reduzir o tempo de espera.

O procedimento de direcionamento pode ser visto como tarefa preliminar para o controle do caos, porque, como já apontado, os algoritmos de controle apresentam bons resultados apenas em vizinhanças do ponto desejado, e por esta razão é interessante que o sistema direcione tal vizinhança antes que seja ligado.

Métodos alternativos têm sido propostos para aumentar o número de visitas a um alvo fazendo pequenas perturbações das variáveis de estado do sistema (Boccaletti *et al*, 2000).

3.4.2 - Estabilizando Trajetórias Caóticas Desejadas

Um conjunto invariante fechado $L \subset R^n$ é chamado de atrator se existir uma vizinhança U de L tal que:

$$\forall\, x \in U\, e\, \forall\, t \geq 0 \Rightarrow \phi(x,t) \in U\, e\, \lim_{t\to\infty} \phi(x,t) \to L.$$

Suponha que exista um sistema dinâmico não linear cujas trajetórias caem sobre um atrator caótico. Suponha também que uma das incontáveis órbitas imersas no atrator caótico, onde órbita é o lugar geométrico no espaço de fase por onde a solução passa na medida que o tempo evolui, corresponde a um estado operacional desejado do sistema (Savi, 2006). A meta é aplicar apenas um pequeno controle de realimentação para manter trajetórias originadas de condições iniciais aleatórias nas proximidades da órbita desejada.

O método de malha fechada para estabilizar uma órbita caótica desejada possui algumas semelhanças com o método OGY de controle do caos. Primeiramente a órbita alvo é escolhida de acordo com as necessidades. Então estabiliza-se uma trajetória originada de uma condição inicial randômica em torno da órbita alvo. Isto pode ser alcançado se a órbita caótica alvo é uma trajetória gerada por equações de evolução do sistema dinâmico. Finalmente, são aplicados pequenos controles de realimentação para estabilizar a órbita caótica alvo. Intuitivamente, a órbita a ser estabilizada tem período igual ao comprimento da órbita.

3.5 - O SISTEMA DE LORENZ

3.5.1 - Modelagem

Lorenz foi um meteorologista que, interessado na previsibilidade do tempo, notou a existência de uma divergência exponencial nas soluções apresentadas por seus estudos mesmo quando analisados sob condições iniciais próximas.

O sistema de Lorenz é um sistema autônomo que possui um atrator estranho. Esse sistema possui 3 (três) equações diferenciais ordinárias (EDO) de ordem 1 (um). São elas:

$$\dot{X} = -\sigma(X - Y), \qquad \text{Eq.(3)}$$

$$\dot{Y} = rX - Y - XZ, \qquad \text{Eq.(4)}$$

$$\dot{Z} = XY - bZ, \qquad (X, Y, Z)\ \epsilon\ \mathbb{R}^3,\ (\sigma, r, b) > 0. \qquad \text{Eq.(5)}$$

A divergência encontrada por Lorenz foi vista como indício da imprevisibilidade evolucional de fluxos em fluidos turbulentos, graças à falta de precisão na determinação das condições iniciais (FIEDLER-FERRARA; PRADO, 1994).

Para chegar ao sistema de Lorenz é necessário remeter-se aos estudos da instabilidade de Rayleigh-Bernárd, a qual estuda o caso de um fluido que se encontra entre duas placas horizontais, onde a placa superior está a uma temperatura menor que a inferior. Esta diferença de temperatura, expressa aqui por $\Delta\Theta$, determina o modo de transferência de calor entre as placas, ou seja, para uma diferença pequena verifica-se a condução de calor, e para diferenças que excedam certo limiar nota-se que a condução dá lugar à convecção, a qual proporciona o surgimento de "rolos" de convecção. Esta é a chamada instabilidade de Rayleigh-Bernárd. O número de Rayleigh crítico é o valor mínimo para que haja convecção no fluido, que assume o valor $r_c = {}^{27\pi^4}/_{4}$ quando $a^2 = {}^{1}/_{2}$, onde a é o quociente entre a distância entre as placas e a largura do rolo de convecção. Com o aumento de $\Delta\Theta$, os rolos convectivos tendem a tornarem-se instáveis e o sistema passa a apresentar uma dependência em relação ao tempo.

Por aproximação (FIEDLER-FERRARA; PRADO,1994) chega-se a três funções temporais: $X(t), Y(t), Z(t)$, as quais não representam as coordenadas espaciais (x, y, z).

Deste modo, depois de alguma manipulação matemática, chega-se às equações do sistema de Lorenz.

As variáveis $X(t), Y(t)\ e\ Z(t)$ possuem significados físicos bem definidos, são eles: $X(t)$ é proporcional à intensidade da convecção; $Y(t)$ é proporcional à diferença de temperatura entre as correntes de fluido ascendente e descendente; e $Z(t)$ é proporcional à distorção do perfil de temperatura vertical, relativo a um perfil linear.

3.5.2 - Estudo de estabilidade do sistema

Considere o parâmetro de controle do sistema como sendo r, número de Rayleigh. Depois de alguns cálculos é possível ver que os pontos de equilíbrio do sistema são:

$(\bar{X}_1, \bar{Y}_1, \bar{Z}_1) = (0,0,0)$, para todo r , e Eq.(6)

$\left(\bar{X}_{2,3}, \bar{Y}_{2,3}, \bar{Z}_{2,3}\right) = (\pm s, \pm s, r-1)$, para $r > 1$. Onde $s = \sqrt{br - b}$ Eq.(7)

O regime de condução é referente ao ponto (0,0,0) e o regime de convecção é referente aos demais.

É possível verificar que o ponto (0,0,0) é estável para $0 < r < 1$, para $r = 1$ o equilíbrio fica instável e o ponto (0,0,0) é um ponto de sela hiperbólico para todo $r > 1$.

Já para os pontos fixos $(\pm s, \pm s, r-1)$, verifica-se a instabilidade para $r < 1$. Para $r = 1$ tais pontos ganham estabilidade linear (bifurcação supercrítica de forquilha) resultante da invariância do fluxo haja vista a simetria $(X, Y, Z) \leftrightarrow (-X, -Y, Z)$. Tais ramos estáveis da forquilha representam fisicamente o início da convecção. Ambos os ramos tornam-se instáveis quando:

$r = r_c = {\sigma(\sigma + b + 3)}\big/{(\sigma - b - 1)}$, com $r = 24{,}7368$, $\sigma = 10$ e $b = {8}/{3}$ Eq.(8)

Para estes valores verifica-se duplicação de Hopf nos pontos fixos em questão.

As bifurcações que ocorrem quando o valor de $r > r_c$ são subcríticas, isto é, não geram órbitas estáveis (na verdade a estabilidade é perdida localmente em r_c). Dessa forma, como conseqüência desse fato, não existem órbitas fechadas para $r > r_c$. Para

$24{,}06 \leq r \leq 24{,}74$ verifica-se a existência de três atratores. Um atrator com comportamento estranho e duas soluções estacionárias. Para esses valores de r verifica-se a existência de caos e histerese. Para valores de r que superam o crítico (r_c) somente existem soluções não-periódicas e os pontos fixos (diferentes de (0,0,0)) são pontos de sela. Analisando fisicamente o sistema é possível dizer que os rolos de convecção ficam instáveis, dando lugar a arranjos com amplitude de movimento maior. Para $24{,}74 \leq r \leq 30{,}1$, considerando $\sigma = 10, b = {}^{8}/_{3}$, o atrator estranho forma a única solução estável do fluxo. É possível ver nas figuras 6, 7 e 8 que as trajetórias estão dentro de uma região bem definida e que o movimento observado não é determinado, já que a seqüência de pontos visitados é caótica em torno dos dois centros existentes. Para $30{,}1 \leq r \leq 214$ o sistema comporta-se de maneira bastante complicada, variando entre regimes periódicos e caóticos.

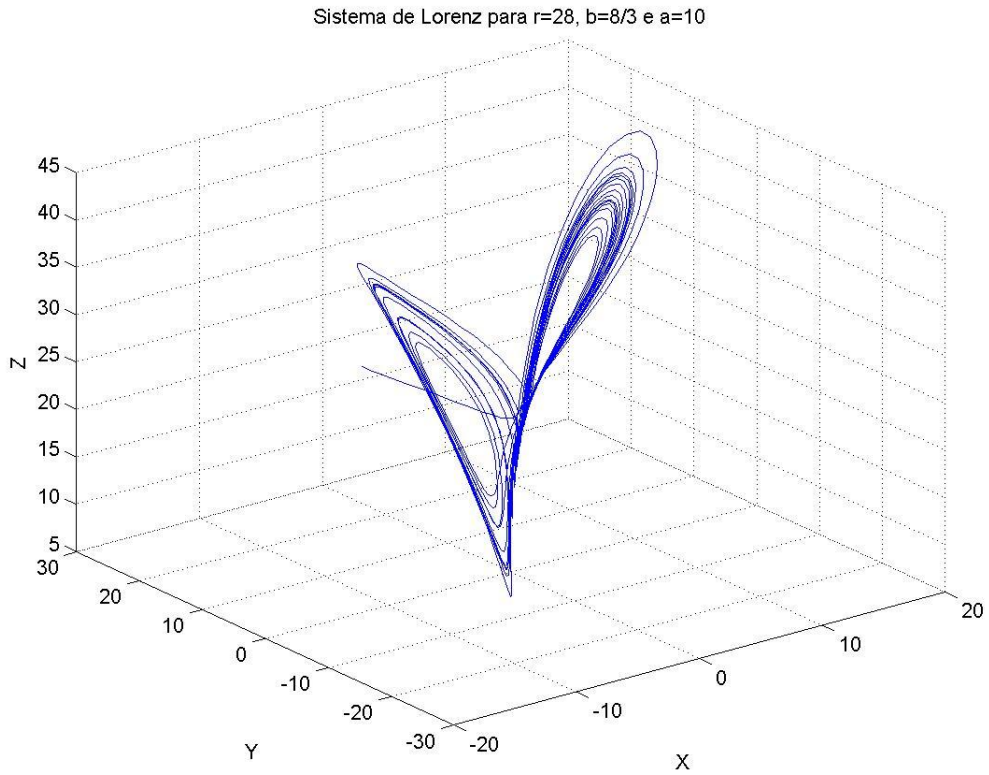

Figura 7: Sistema de Lorenz para $r = 28, b = {}^{8}/_{3}, a = 10$.

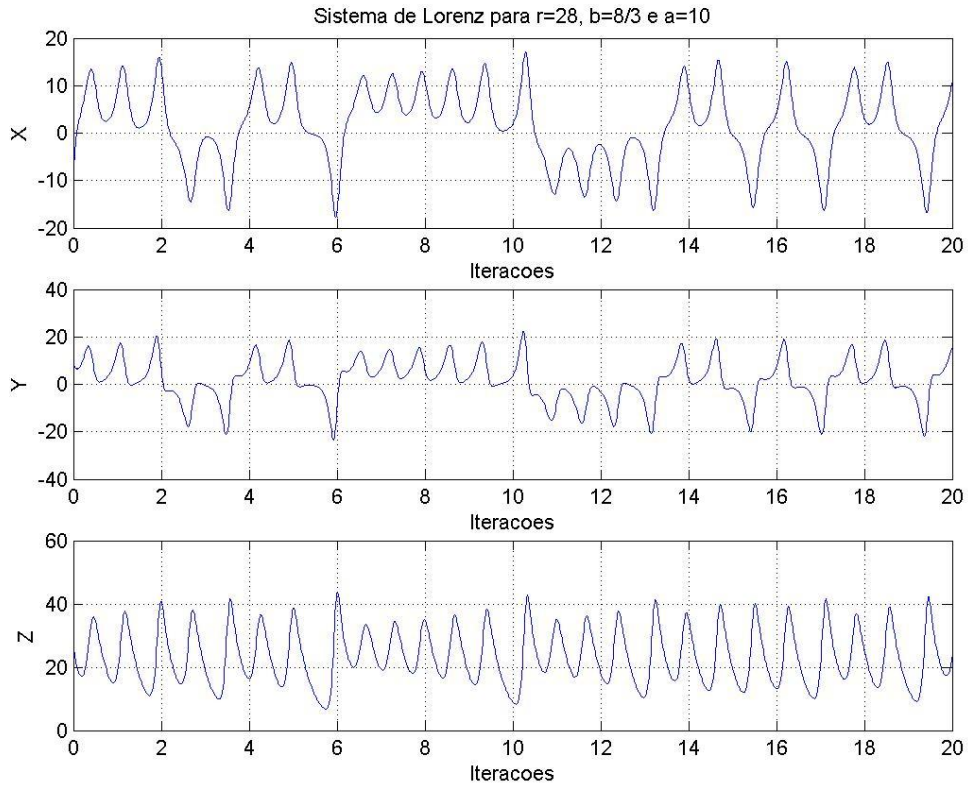

Figura 8: Variáveis X, Y e Z do Sistema de Lorenz em função do tempo de integração.

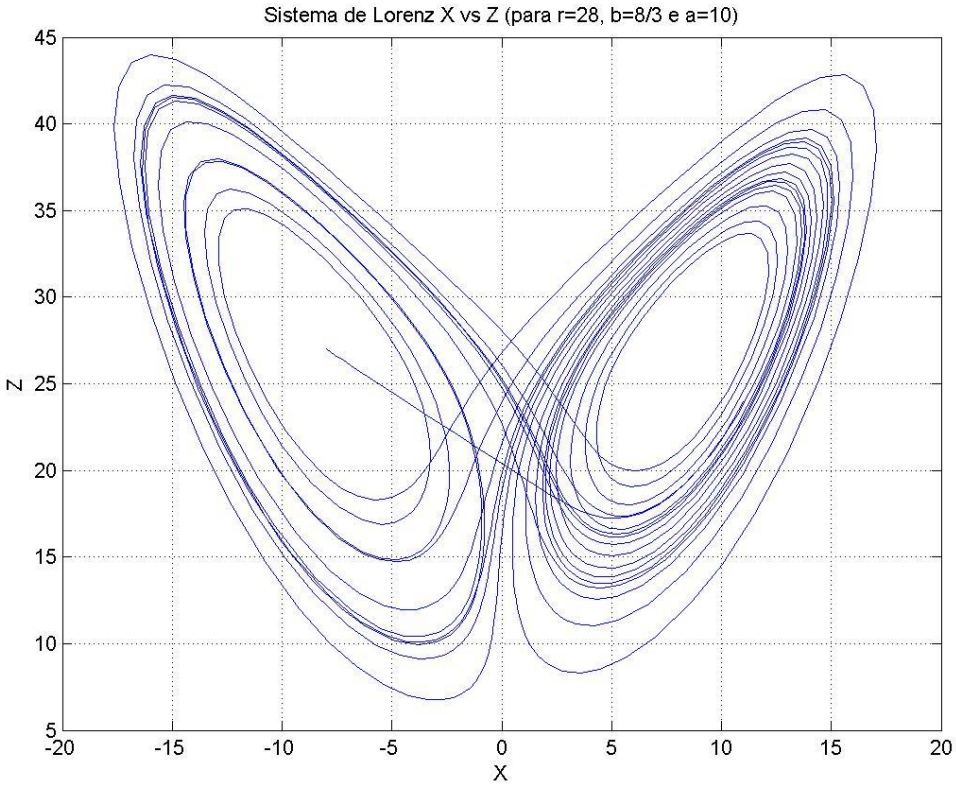

Figura 9: Variável X em função da variável Z do Sistema de Lorenz.

4 - METODOLOGIA

Este capítulo descreve detalhadamente o software criado, passando por cada etapa de seu desenvolvimento e pelos resultados esperados nestas etapas.

4.1 - INTRODUÇÃO

A fim de projetar um sistema de controle capaz de controlar, sobre uma órbita periódica, um sistema dinâmico, instável e não linear, escolheu-se o Sistema de Lorenz como modelo de sistema caótico para o desenvolvimento deste projeto.

4.2 - CONSTRUÇÃO E PROCESSAMENTO DA REDE NEURAL ARTIFICIAL

A rede neural usada nesse trabalho foi escolhida para ser o mais simples possível, sendo formada por quatro camadas, fig.9. A primeira delas é a de entrada, composta por um nó, e recebe uma medida da variável X do sistema de Lorenz. A entrada fornece à rede o estado atual do sistema que se deseja controlar. Isso torna possível o cálculo iterativo da perturbação que será aplicada ao sistema durante a execução do programa.

A segunda camada é chamada *Hidden*, Oculta. Foi decidido, arbitrariamente, que essa camada seria formada por 20 nós. Esse número é diretamente proporcional à capacidade de treinamento da RNA, ou seja, quanto mais nós existirem na camada Oculta, mais fácil é para rede aprender a controlar um determinado sistema. Esses 20 nós são associados, individualmente, a um peso w o qual pondera sua conexão com o nó da camada de entrada. Os nós da camada interna trabalham juntos formando um time que é avaliado segundo sua adequação ao problema proposto.

A terceira camada é denominada Net, e é composta por apenas um nó. Esse nó é gerado a partir da quantificação da adequação da camada interna à situação na qual a rede é submetida. Este valor será inversamente proporcional à perturbação que será aplicada à trajetória que se deseja estabilizar.

A quarta e última camada é o Supervisor da rede. Este conhece o resultado que se deseja obter com o treinamento da rede, ou seja, a órbita periódica alvo. O supervisor recebe o valor fornecido pela camada Net e o analisa, juntamente com a diferença entre o estado desejado e o atual da trajetória. Com essas informações o Supervisor é capaz de

alterar a perturbação a ser aplicada sobre a trajetória na tentativa de se atingir a estabilidade da mesma sobre a órbita desejada. A figura 10 apresenta um diagrama da RNA utilizada nesse trabalho.

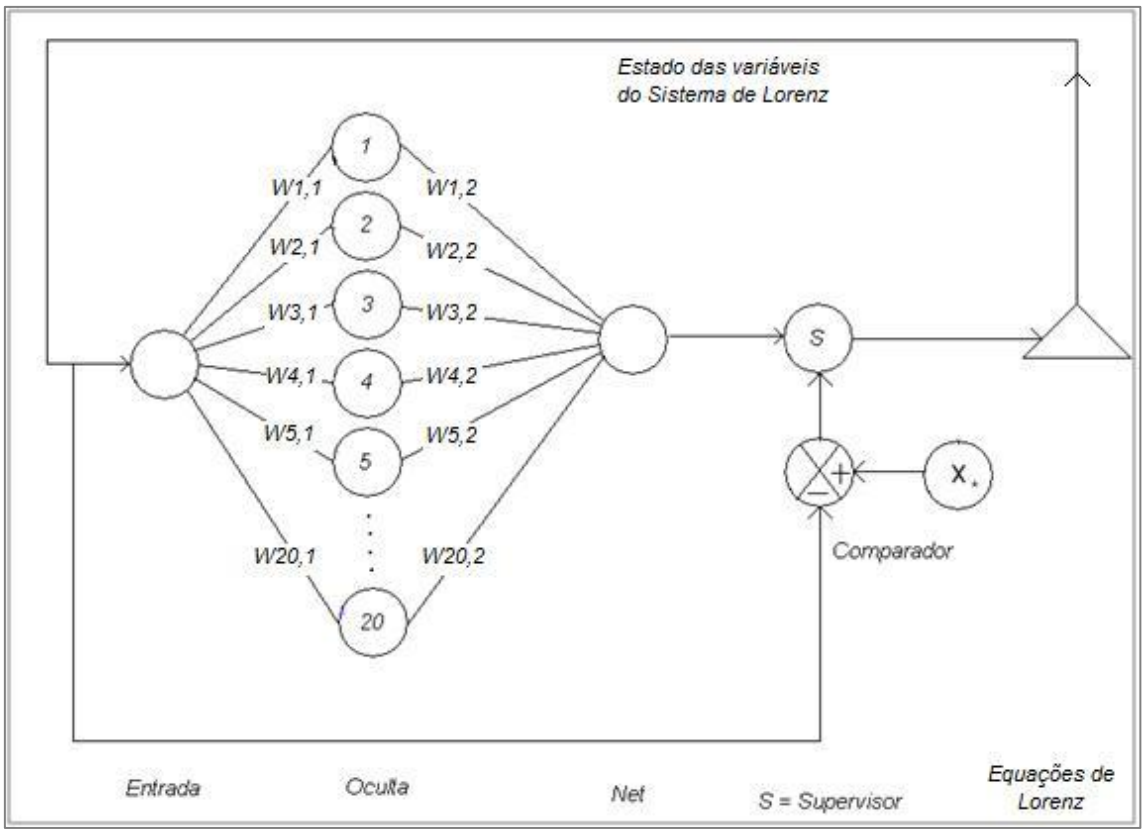

Figura 9: Diagrama da RNA usada nesse trabalho.

4.3 - TREINAMENTO E SIMULAÇÃO DO SISTEMA DE CONTROLE

Tendo sido exposta a topologia da rede neural utilizada nesse projeto, é importante entender alguns termos empregados durante todo este trabalho. Passo do programa é a menor unidade cíclica de execução do software. Ele tem início quando o estado atual da trajetória a ser controlada é aplicado à rede neural e tem fim quando a rede retorna um valor de perturbação que será aplicada sobre a trajetória produzindo um novo estado na mesma.

Depois de um número de passos, escolhido arbitrariamente pelo usuário da RNA, há a substituição da camada interna por outra a fim de testar o comportamento de novos indivíduos na tarefa de controlar o sistema. Cada troca de camada interna é chamada de iteração.

Depois de uma quantidade de iterações, também definida por quem utiliza a RNA, são aplicados os operadores genéticos sobre a população da qual é selecionada a camada interna. Cada vez que os operadores genéticos são aplicados é chamada de geração. Tendo estas definições é possível continuar a explanação sobre o treinamento e a simulação do sistema de controle.

A princípio o programa gera, aleatoriamente, uma população inicial, formada por 100 indivíduos a qual é colocada na "bacia de seleção". Cada indivíduo possui três (3) loci de informação (X_{ij}), onde o primeiro corresponde ao peso w_{ij} com o qual será ponderada a entrada do sistema, estabelecendo ligação entre a entrada e a camada Oculta.

O segundo loci é reservado para armazenar os pesos w_{ij} com os quais os nós da camada Oculta serão ponderados para que sejam avaliados antes de serem descartados ou terem seus valores processados e repassados à camada Net. Para a primeira população esses pesos são gerados aleatoriamente entre dois valores previamente escolhidos, neste caso no intervalo [-1,1].

O fluxograma da fig.10 representa os nós de uma rede hipotética e os pesos nas ligações entre eles.

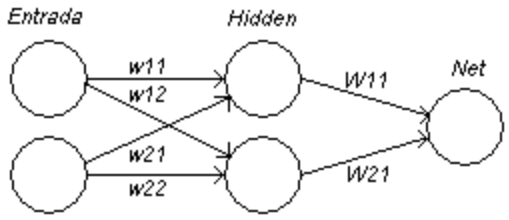

Figura 10: Fluxograma das camadas de uma rede com seus nós e respectivos pesos.

O terceiro loci, inicialmente nulo, é reservado para armazenar o aptidão do nó em questão. A aptidão de cada nó é o valor que mensura o quão adequado é esse nó para que se atinja a estabilidade do sistema. A aptidão é atribuída conjuntamente aos nós da camada Oculta ao fim de cada iteração, ciclo de execução do programa, quando tais indivíduos retornam para a "bacia de seleção", e uma nova Oculta é selecionada.

Cada nó mantém a aptidão obtida na melhor RNA que este tenha participado, melhor time, ou seja, a maior aptidão assumida durante toda a execução da geração. Na formação da população inicial, um processo aleatório seleciona um número pré-

determinado de indivíduos desta população, neste caso 20, para formar a camada Oculta da RNA. Então é gerada uma entrada para o sistema.

A entrada do sistema é um valor da variável X do sistema de Lorenz gerada aleatoriamente em torno de um dos pontos fixos não triviais de Lorenz. A distância entre a entrada e o ponto fixo varia através da alteração de um parâmetro controlado pelo usuário do programa (a distância é diretamente proporcional ao parâmetro). A partir do segundo passo, medida seqüencial, do programa a entrada passa a ser obtida através do cálculo da influência da perturbação sobre o sistema, deixando de ser gerada aleatoriamente.

A entrada será repassada à camada Oculta, a qual irá ponderá-la usando o peso W_{ij}, gerando uma matriz coluna denominada Y. Essa matriz é aplicada a uma função sigmoidal ajustada para fornecer saída no intervalo [-1,1], gerando outra matriz coluna dentro desses limites. Cada elemento desta matriz é ponderado pelo seu peso correspondente $w_{i,2}$ dos nós da camada Oculta, os valores ponderados são somados e o resultado é aplicado à função sigmoidal, de mesmas características que a anteriormente citada, e o resultado obtido forma a terceira camada da RNA, Net. Um maior detalhamento sobre a função sigmoidal é fornecido no anexo A. O diagrama da fig.11 mostra o processo descrito.

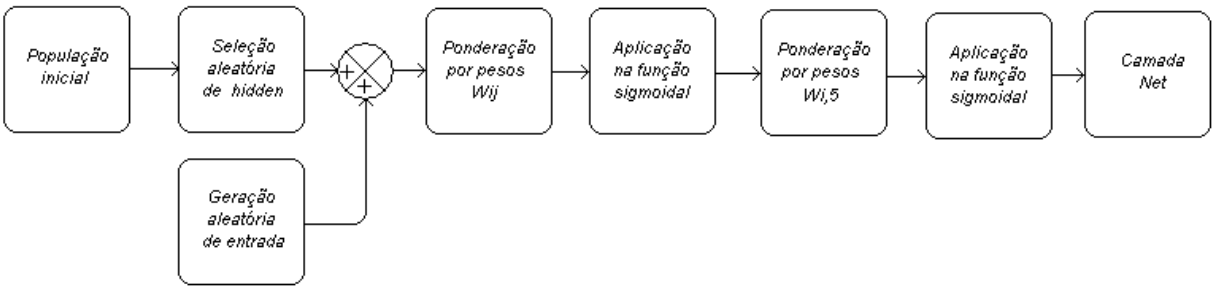

Figura 11: Diagrama seqüencial da execução do software.

Nesse ponto o valor que é armazenado em Net é um escalar, e este é passado para a quarta camada da rede neural, o Supervisor. Ele realiza a parte de controle do sistema que calcula perturbações diretamente proporcionais ao valor de Net medindo a situação do sinal X do sistema de Lorenz. Os parâmetros relacionados pelo supervisor para chegar à perturbação que será aplicada ao sistema são os seguintes. Primeiramente é necessário que o valor atual de X esteja dentro de uma vizinhança, de raio determinado pelo usuário do programa, do valor de X atrasado de um período para que

o controle seja ativado. Caso esta condição não seja satisfeita o programa continua sua execução sem perturbar o sistema até sua que a estocasticidade garanta a proximidade mínima necessária da órbita desejada para que o controle seja ligado. Satisfeita a primeira condição, quanto menor o módulo do valor entregue por Net, menor será a perturbação aplicada ao sistema. Quando o sistema se estabiliza por um período maior ou igual a cinco vezes o período estipulado pelo usuário do programa, a perturbação aplicada será nula. O diagrama da fig.12 mostra esta parte do processo.

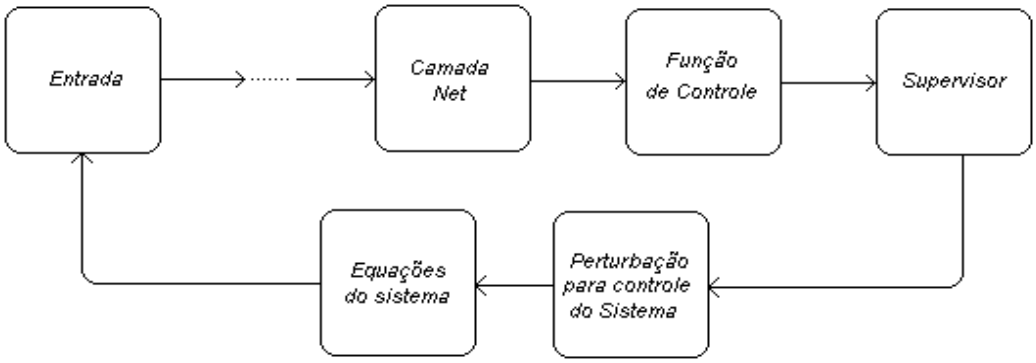

Figura 12: Diagrama da parte de realimentação do sistema de controle.

De posse da perturbação calculada pela função de controle para a iteração atual, tal valor é usado no modelo matemático do sistema a ser controlado, equações de espaço de estados, já mencionado anteriormente. Um fato importante dessa etapa é que, embora o sistema de equações seja utilizado para simular o modelo, as redes neurais não utilizam nenhuma informação acerca deste modelo, tratando-o como uma caixa preta. Essas equações irão gerar o novo valor de X, valor este que será usada nos cálculos da próxima iteração, ou passo, do sistema.

Outra distinção importante a ser feita é dividir a execução do programa em duas etapas: O treinamento da rede neural artificial onde, iterativamente, são calculados os pesos W dos indivíduos da camada Oculta capazes de controlar o sistema, usando no processo os operadores genéticos, e o controle do sistema onde uma rede já obtida na fase de treinamento é aplicada ao sistema a fim de controlá-lo.

Para verificar a estabilidade do sistema não basta que ele passe sobre a órbita desejada, além disso, é necessário que ele permaneça sobre tal órbita periódica, dentro de um intervalo estabelecido pelo usuário do software, por vários passos, medidas seqüenciais, do sistema, neste caso cinco vezes o período da órbita. Caso esta condição seja satisfeita o programa é interrompido, uma mensagem de estabilidade é ativada e um

gráfico do comportamento do sistema em função do tempo é mostrado.

Enquanto esta condição não é satisfeita o programa continua em execução cíclica, aplicando o valor encontrado na saída, valor de X, como realimentação da entrada da RNA. O software realiza, continuamente, o treinamento da RNA e seu teste no sistema, dessa forma, possíveis alterações no sistema a ser controlado são refletidas no comportamento do sistema de Lorenz e a RNA se adéqua, iterativamente, a tais alterações, buscando manter a planta controlada.

Nesse processo de aprendizado da RNA, o sistema pode sair da órbita desejada várias vezes. Quando isso ocorre tem-se que o controle é desligado e permanece assim até que o sistema passe novamente sobre a região programada para ativar o controle.

São realizados no máximo dez mil passos para cada camada Oculta selecionada, tentando estabilizar o sistema. Se ao fim destes passos as condições de estabilidade não forem satisfeitas um valor de aptidão inversamente proporcional ao número de passos que o sistema se manteve em funcionamento e à diferença entre os valores de X atual e atrasado de um período é atribuído aos nós da camada Oculta corrente, conforme mostrado na equação 9, e os mesmos retornam para a bacia de seleção para que um novo processo se inicie.

$$Fitness = \frac{10001 - Num\_Passos}{1 + |Erro\_X|} \qquad Eq.(9)$$

São escolhidas 8 camadas Oculta diferentes. Se ao fim desse processo a estabilidade ainda não tiver sido alcançada, os operadores genéticos são aplicados na população da bacia de seleção, pop.

Primeiramente a população, antes em base decimal, é codificada em base binária, no intervalo [-1,1] em uma string com 12 posições, loci. Esta quantidade de posições foi escolhida para garantir fidelidade aos dados originais após sua codificação em base binária ou decodificação em base decimal. A população binária resultante passa pelo processo de recombinação genética aplicado pela função cruzamento. Esta função inicialmente ordena os indivíduos em ordem decrescente de aptidão, os primeiros indivíduos terão maior probabilidade de estar no cruzamento, visto que são mais "bem adaptados" à estabilização do sistema pela rede neural do que os demais.

A metade dos indivíduos da população mais bem adaptados será submetida ao processo de recombinação genética, no qual se escolhem aleatoriamente indivíduos pais,

separados em pares, e pontos de cruzamento de forma a produzir indivíduos filhos diferentes de seus geradores.

A outra metade dos indivíduos adaptados é substituída por indivíduos da primeira metade que sofreram processo de mutação. Este processo é realizado pela função denominada mutação. Esta função tem como entrada a população binária e uma probabilidade de mutação, entre zero e um, arbitrária. Inicialmente ela gera um número aleatório entre zero e um e, caso esse número seja menor ou igual à probabilidade de mutação pré-estabelecida, a mutação ocorrerá.

A mutação, neste caso, consiste em alterar a posição de um bit escolhido aleatoriamente do indivíduo, ou seja, caso tal bit seja igual a zero, será modificado para um, e vice versa.

Concluídos os processos genéticos de cruzamento e mutação, a nova população está formada, e esta nova população binária é decodificada, convertendo a população para base decimal.

Esse processo de geração de uma nova população é realizado oito vezes, e a cada vez que ocorre, a população gerada passará novamente por até oito escolhas, iterações, de camada Oculta e cada uma destas escolhas serão submetidas a até 10000 passos do programa.

Esse processo é interrompido a qualquer momento caso a estabilidade do sistema seja alcançada.

5 – DESENVOLVIMENTO DO SOFTWARE

Este capítulo aborda de maneira mais detalhada a construção do programa, suas partes e como as mesmas se relacionam.

5.1 – FERRAMENTAL UTILIZADO

O software foi desenvolvido em linguagem de programação MATLAB, versão 7.8.0 (R2009a). Foi utilizado um notebook com processador Intel Core 2 Duo (1.83GHz, 667 MHz FSB, 2MB cache) e 3 GB de memória RAM.

O programa não utiliza *toolboxes* do MATLAB nem funções prontas, sendo completamente desenvolvido utilizando texto estruturado.

5.2 – ORGANIZAÇÃO DO PROGRAMA

O software é organizado em rotinas e sub-rotinas. A rotina principal denominada "*Main*" é executada pelo operador e tem a função de chamar seqüencialmente as sub-rotinas, passando-lhes informações em forma de argumento, para que as mesmas retornem informações que possibilitem a continuação da execução do programa. A fig.13 mostra a estrutura organizacional do programa desenvolvido.

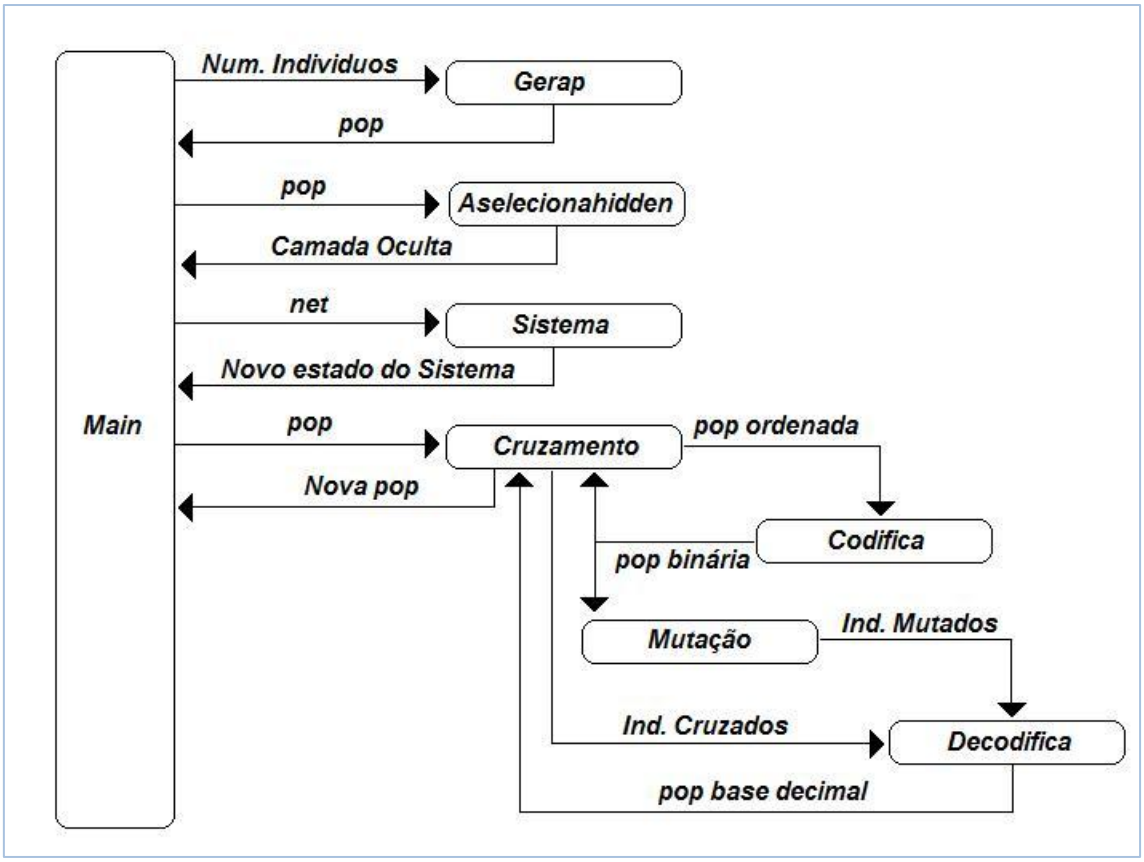

Figura 13: Diagrama estrutural do software desenvolvido.

5.2.1 – Rotina Main

Esta rotina contém todo o programa visto que nela estão todas as sub-rotinas que devem ser executadas para concluir a execução do mesmo. Ela tem início com a declaração e inicialização de todas as variáveis que serão usadas pelo programa. Neste ponto são declarados pelo operador o valor do período da órbita sobre a qual se deseja controlar o sistema caótico bem como a quantidade de passos de execução do programa dentro de cada iteração.

Feito isso, a rotina "*Main"* chamará a sub-rotina "*Gerap"*, que tem a função de gerar uma população de 100 (cem) indivíduos que comporão a bacia de seleção. Em seguida são declarados pelo usuário o número de gerações e iterações máximos que o programa poderá completar sem que atinja a estabilidade do sistema sobre a órbita periódica desejada. Estas declarações são feitas na forma de laços de repetição do tipo "*for*".

Em seguida é chamada a sub-rotina "*Aselecionahidden*", a qual tem a função de selecionar a camada "*Oculta*" da rede neural artificial. Esta sub-rotina deve ser chamada

fornecendo a ela como argumento a matriz que contém a população presente na bacia de seleção e retorna uma matriz contendo uma cópia exata dos 20 (vinte) indivíduos selecionados e suas respectivas posições na bacia de seleção.

A execução continua com a determinação, por parte do operador, das condições iniciais do sistema de Lorenz. A Variável "*Teta*" contém um dos pontos fixos não triviais de Lorenz, escolhido pelo operador. É possível, através da variável "*DeltaSaida1*" selecionar qual o afastamento das condições iniciais em relação ao ponto fixo selecionado.

Então tem início a etapa de aprendizado da rede neural artificial. Primeiramente gera-se uma matriz coluna "Y_1" do produto entre as condições iniciais que são aplicadas na entrada da rede neural e os pesos referentes a cada indivíduo, ou nó, presente na camada "*Oculta*". É produzida uma matriz coluna "F_{Y1}" a partir da aplicação de "Y_1" em uma função sigmoidal, que tem por função modularizar os valores de "Y_1" dentro do intervalo *[-1,1]*. O somatório do produto entre "F_{Y1}" e os pesos existentes no segundo loci dos nós da camada "*Oculta*" gera um escalar que é armazenado em "Z_1". Ao aplicar "Z_1" a uma função sigmoidal, de mesmas características que a anterior, obtém-se o valor que será armazenado pela camada "*Net*" representada no programa pela variável "Net_1". Este número é passado como argumento para a sub-rotina "*Sistema*" que faz o papel do supervisor da rede neural artificial, calculando e aplicando a perturbação sobre a variável disponível do sistema de Lorenz e retornando como resultado o novo estado do sistema na forma de coordenadas *(X,Y,Z).*

A cada passo de execução do programa são registradas as perturbações aplicadas sobre o sistema e o novo estado do mesmo através das variáveis "*registra_f_media*" e "*registra_entrada*" respectivamente. Caso o sistema esteja estável sobre a órbita periódica selecionada no passo vigente um contador de estabilidade representado pela variável "*contaestab*" é acrescido de uma unidade. Caso esta estabilidade não seja verificada em algum passo posterior tal contador é zerado.

O programa é executado sem perturbar o sistema até que seja atingido o número de passos igual ao período estipulado para controle do mesmo. A partir daí torna-se possível a comparação, através de coordenadas de atraso temporal, entre o estado atual do sistema e o estado do mesmo atrasado de um período. A perturbação que será aplicada no sistema em determinado passo é diretamente proporcional a esta diferença e pode ser tanto positiva quanto negativa.

Caso o valor de "*contaestab*" seja superior a cinco vezes o período selecionado para controle do sistema, o programa considera que a estabilidade foi atingida e é interrompido. Uma mensagem de estabilidade é apresentada, um gráfico da evolução temporal do sistema é mostrado e a camada "*Oculta*" vigente é registrada na matriz "*hiddenfinal*" para ser utilizada posteriormente nos testes de controle com a rede neural artificial treinada.

Por outro lado, caso a estabilidade não seja atingida dentro de uma iteração, os indivíduos da camada "*Oculta*" recebem um valor de adequação inversamente proporcional à diferença entre o estado corrente e o atrasado de um período do sistema de Lorenz e retornam para a bacia de seleção. Este número é atribuído ao terceiro loci de informação de cada nó da camada "*Oculta*" e somente será substituído por outro valor, em iterações subseqüentes, caso o novo número seja maior que o previamente registrado. Isto garante que cada indivíduo que tenha participado de uma camada "*Oculta*" durante a execução do programa guardará sempre a nota do melhor time do qual tenha participado, o que será importante durante a aplicação dos processos genéticos na população da bacia de seleção.

Se, mesmo depois do programa ser executado pelo número de iterações determinado pelo operador, a estabilidade não for alcançada, os processos dos algoritmos genéticos são aplicados através de três sub-rotinas: a "*Codifica*" que muda a base dos indivíduos da bacia de decimal para binária; a "*Cruzamento*" que aplica os processos de seleção, cruzamento e mutação sobre a população vigente; e a "*Decodifica*" que retorna para a base decimal a população resultante dos processos genéticos. O programa continua sua execução por até que a estabilidade sobre a órbita selecionada seja alcançada ou até que se atinja o número de gerações máximo determinado pelo operador do software.

5.2.2 – Sub-rotina Gerap

Esta sub-rotina deve ser chamada juntamente com três argumentos: o "*limsup*" ou limite superior, que determina o valor máximo que pode ser assumido por um indivíduo gerado por esta sub-rotina; o "*liminf*" ou limite inferior, que determina o valor mínimo que pode ser assumido por um indivíduo gerado pela mesma; e o "*numind*" ou número de indivíduos que serão gerados pela sub-rotina. Para cada indivíduo gerado, a "*Gerap*" produz, através de laços de repetição, valores randômicos entre os limites

inferior e superior para dois lócus de informação. O terceiro loci é preenchido a princípio com zero e posteriormente guardará o valor da adequação do indivíduo após participar de um time, ou seja, de uma iteração na camada "*Oculta*". Esta sub-rotina retorna a rotina "*Main*" a matriz "*pop*" contendo a população que estará presente na bacia de seleção.

5.2.3 – Sub-rotina Aselecionahidden

Esta sub-rotina recebe como argumento a matriz "*pop*", que contém a população presente na bacia de seleção, e retorna para a rotina "*Main*" indivíduos e suas respectivas posições, na quantidade determinada pelo operador para compor a camada "*Oculta*". Este número é proporcional à capacidade de aprendizado e adaptação da rede neural artificial treinada. Cada indivíduo é selecionado aleatoriamente através da geração de um número randômico entre zero e o número total de indivíduos presentes em "*pop*".

5.2.4 – Sub-rotina Sistema

Esta sub-rotina recebe como argumento o valor presente na camada "*Net*" da rede neural artificial. Primeiramente é verificado se o número de passos realizados é maior que o período da órbita periódica selecionada. Em caso negativo, a execução do programa continua sem perturbar o sistema até que haja dados suficientes para realizar a comparação do estado corrente do sistema e o mesmo atrasado de um período. Em caso positivo, a entrada é comparada com o estado do sistema defasado de um período e, de acordo com o resultado, quatro situações distintas podem ocorrer.

- $Estado_{defasado} + 0{,}1 \leq Entrada \leq Estado_{defasado} + 10$
- $Estado_{defasado} - 10 \leq Entrada \leq Estado_{defasado} - 0{,}1$
- $Estado_{defasado} - 0{,}1 < Entrada < Estado_{defasado} + 0{,}1$
- $\left|Entrada - Estado_{defasado}\right| > 10$

Para o primeiro caso será calculada uma perturbação diretamente proporcional a tanto ao valor passado pela camada "*Net*" quanto à diferença entre o valor da entrada corrente e do estado do sistema defasado de um período e terá o mesmo sinal desta

última. Para o segundo caso o cálculo da perturbação é igual. Na terceira hipótese considera-se que a trajetória está sobre a órbita periódica desejada e nenhuma perturbação é aplicada à mesma. E para a quarta hipótese considera-se que a trajetória está fora da vizinhança da órbita periódica desejada e nenhuma perturbação é aplicada à mesma. Das quatro hipóteses, a única que acresce o contador de estabilidade em uma unidade é a terceira. As demais levam o contador novamente a zero.

De posse da perturbação a ser aplicada sobre a variável disponível, o número de Rayleigh, as EDO's do sistema de Lorenz são resolvidas pela sub-rotina "*Lorenz_novo*" utilizando um passo de integração de um centésimo de segundo. Como resultado é obtido o novo estado do sistema de Lorenz, informação que retorna para a rotina principal "*Main*" como resposta da sub-rotina "*Sistema*".

5.2.5 – Sub-rotina Codifica

Esta sub-rotina tem como função codificar em base binária um conjunto de indivíduos a princípio em base decimal. Tal sub-rotina recebe como argumentos a população presente na bacia de seleção, os limites inferior e superior que cada indivíduo pode assumir em base decimal e o número de loci de informações em que cada indivíduo será codificado em base binária.

Primeiramente é calculado o comprimento da população passada para a sub-rotina, então através de um laço de repetição cada um dos indivíduos desta população é codificado para base binária utilizando a função "*Dec2Bin*" com a quantidade de posições escolhida pelo operador. Por fim os indivíduos codificados são concatenados para formar a nova população, agora em base binária. Esta população é passada para a rotina principal "*Main*".

5.2.6 – Sub-rotina Cruzamento

Esta sub-rotina tem a função de realizar os operadores genéticos sobre a população da bacia de seleção após completa uma geração de execução do programa. Tal sub-rotina recebe como argumentos a população da bacia de seleção, chamada "*pop*" e informações como o comprimento de cada indivíduo em base binária.

Primeiramente é calculado o número de indivíduos da população. Então, através de laços de repetição do tipo "*for e while*" os indivíduos presentes na bacia de seleção

são ordenados conforme seus valores de adequação utilizando o método bolha. Concluída esta etapa, os indivíduos ordenados são codificados em base binária através da sub-rotina "*Codifica*" e a matriz resultante é utilizada para continuar as operações genéticas.

O passo seguinte é realizar o cruzamento. Este processo é realizado utilizando um laço de repetição "*for*", que garante que cinqüenta por cento da população resultante dos processos genéticos sejam resultado de cruzamento dos indivíduos mais adaptados da população da geração anterior. Primeiramente os indivíduos são separados aleatoriamente em pares. Então é selecionado para cada par, aleatoriamente, um ponto de cruzamento. Este ponto será o local, na seqüência de codificação dos indivíduos, que ocorrerá a troca de informação genética.

Por fim o primeiro indivíduo filho será composto pela codificação do primeiro indivíduo pai até o ponto de cruzamento e a partir daí, até o fim da seqüência genética, terá a codificação do segundo indivíduo pai. Por outro lado o segundo indivíduo filho será composto pela codificação do segundo indivíduo pai até o ponto de cruzamento e a partir daí, até o fim da seqüência genética, terá a codificação do primeiro indivíduo pai.

Concluído o processo de cruzamento, esta sub-rotina realiza os operadores de mutação sobre a população chamando a sub-rotina "*Mutação*" retornando à rotina "*Main*" a nova população composta pela união entre os indivíduos gerados por cruzamento e os gerados por mutação.

5.2.7 – Sub-rotina Mutação

Esta sub-rotina recebe como argumentos a população binária ordenada conforme seus valores de adequação e um valor, entre zero e um, que representa a probabilidade de mutação de cada indivíduo.

Primeiramente são calculados o número de indivíduos da população e o comprimento de cada indivíduo. Então, dentro de um laço de repetição do tipo "*for*", é escolhido aleatoriamente o indivíduo da população candidato a sofrer mutação. Em seguida gera-se um número aleatório entre zero e um; caso este seja menor ou igual à probabilidade de mutação passada à sub-rotina na forma de argumento o indivíduo sofrerá mutação; caso contrário o mesmo segue, sem alteração para a próxima geração.

Para os casos em que a ocorrerá a mutação, o processo é feito da seguinte maneira. Primeiramente seleciona-se aleatoriamente um ponto do indivíduo, loci de

informação, que será mutado. Então caso este loci seja igual a “1” tal valor será substituído por “0” e vice-versa.

O processo se repete até que seja realizado, com ou sem alteração do indivíduo, sobre um número igual à metade da população presente na bacia de seleção. Por fim a sub-rotina “*Mutação*” retorna à sub-rotina “*Cruzamento*” a população de indivíduos mutados.

5.2.8 – Sub-rotina Decodifica

Esta sub-rotina tem como função decodificar em base decimal um conjunto de indivíduos a princípio em base binária. Tal sub-rotina recebe como argumentos a população binária resultante dos processos genéticos aplicados pela sub-rotina “*Cruzamento*”, os limites inferior e superior que cada indivíduo pode assumir em base decimal e o número de loci de informações em que cada indivíduo binário possui para ser decodificado.

Primeiramente é calculado o comprimento da população passada para a sub-rotina, então através de um laço de repetição cada um dos indivíduos desta população é decodificado para base decimal utilizando a função “*Bin*2*Dec*” com a quantidade de posições que cada indivíduo possui para ser decodificada. Por fim os indivíduos decodificados são concatenados para formar a nova população, agora em base decimal. Esta população é passada para a rotina principal “*Main*”.

6 - RESULTADOS

Este capítulo apresenta os testes feitos com o software desenvolvido, incluindo seus resultados e análises.

6.1 APRESENTAÇÃO E ANÁLISE DE RESULTADOS

Tendo conhecimento dos tópicos abordados nas seções 3, 4 e 5 sobre o funcionamento do software, torna-se possível a apresentação e compreensão dos resultados obtidos com a execução do mesmo na simulação do sistema de Lorenz.

Para executar o software utilizou-se um computador com processador Intel Core 2 Duo (1.83GHz, 667 MHz FSB, 2MB cache) e 3 GB de memória RAM. No programa implementado a rede neural é treinada comparando o estado atual do sistema de Lorenz a o estado esperado fornecido pelo supervisor da rede. Dessa forma, a escolha de um passo de integração para o modelo, ou taxa de amostragem do estado corrente, é decisiva para que se atinja um bom resultado na tentativa de controlar tal sistema.

Um passo de integração muito elevado gera grandes problemas para o sistema de controle, pois torna necessário o uso de interferências extremas (perturbações de módulo elevado) no sistema a ser controlado, na tentativa de impedir que tal sistema entre em colapso, nesse caso, que o sistema de Lorenz saia da vizinhança da região de controle. O passo de integração de iteração sendo demasiadamente longo, a mesma interferência que atua no sistema buscando estabilizá-lo em um instante de tempo, pode ser o agente que causa sua instabilidade permanente em um intervalo pequeno de tempo seguinte. Caso esse período entre duas amostras do estado do sistema for maior que o tempo necessário para que tal perturbação leve o sistema a um estado próximo à órbita desejada, a trajetória vai passar do ponto de equilíbrio. Nesse ponto a perturbação que atua no sistema acaba por levá-lo novamente ao estado caótico de movimento.

Por outro lado, um período demasiadamente pequeno de iteração pode ser impraticável em sistemas reais, ou seja, mesmo que o uso de períodos de integração muito pequenos produzam controles excelentes em sistemas desejados, se este período de integração não for passível de ser desenvolvido em situações reais sua utilidade se torna limitada, invalidando qualquer resultado teórico obtido na etapa de projeto e simulação.

Assim, busca-se alcançar um resultado que contemple tanto a abordagem teórica quanto a possibilidade de colocar em prática o sistema que foi simulado.

A seguir são expostos vários resultados da execução do software para situações diferentes.

O gráfico da figura 14 mostra o sinal no controle do sistema de Lorenz sobre uma órbita de período de 650 ms, equivalente a 65 passos visto que cada passo tem duração de 10 ms. Neste caso o controle sobre a órbita periódica foi atingido com 5 gerações, 5 iterações e 530 passos.

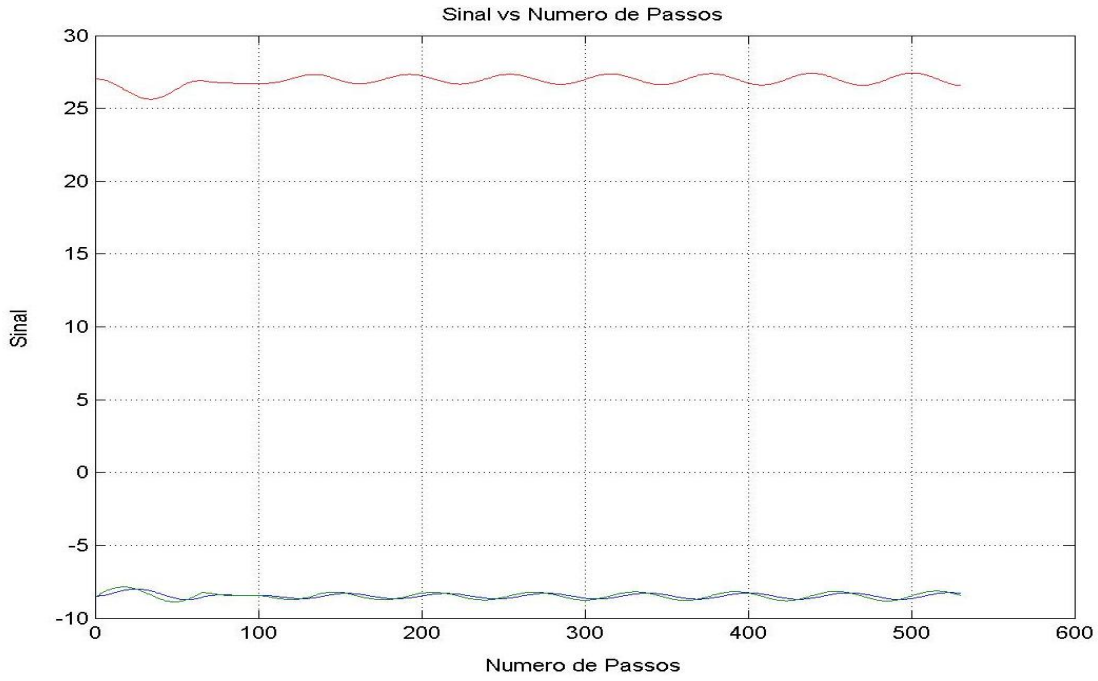

Figura 14: Sistema de Lorenz – Sinal estabilizado

A figura 15 mostra no detalhe o sinal obtido da variável X do sistema de Lorenz, a partir deste ponto chamado de sinal medido, estabilizado, pois é importante lembrar que a perturbação do sistema é aplicada sobre o número de Rayleigh, o qual atua apenas sobre o sinal X. Nesta execução do programa a estabilidade foi atingida na geração 2, com 7 iterações e 391 passos.

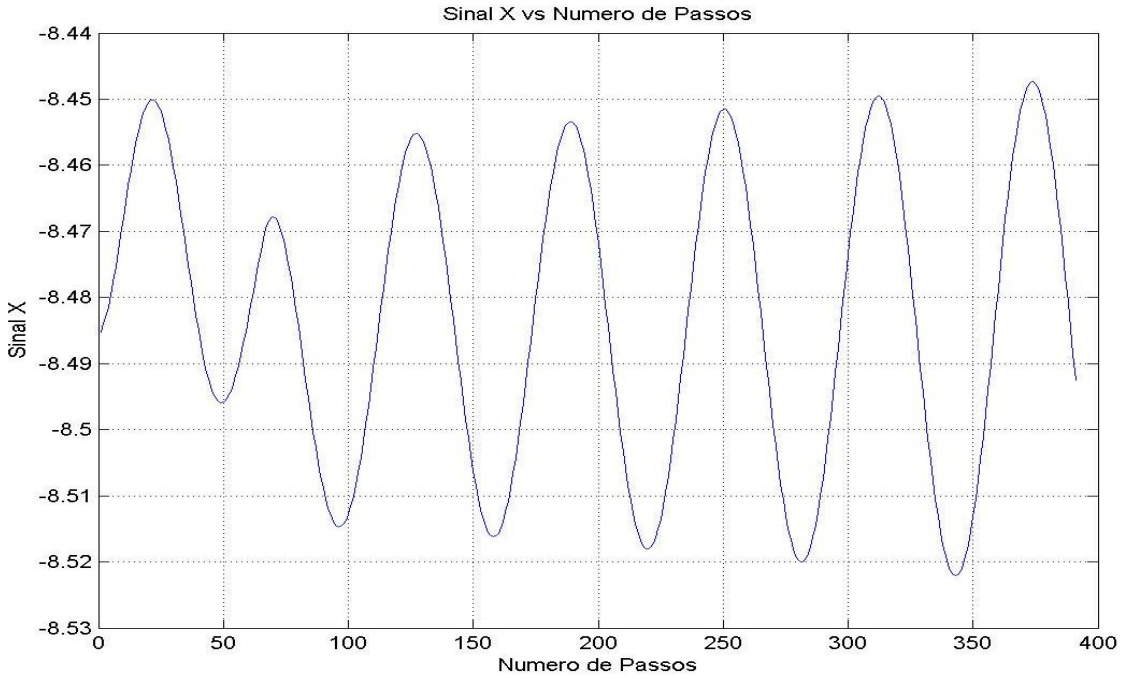

Figura 15: Sistema de Lorenz – Sinal medido estabilizado pelo número de passos

A fim de mostrar o comportamento da perturbação durante a iteração do programa em que foi atingida a estabilidade do sistema foi traçado o gráfico da perturbação pelo número de passos, na figura 16.

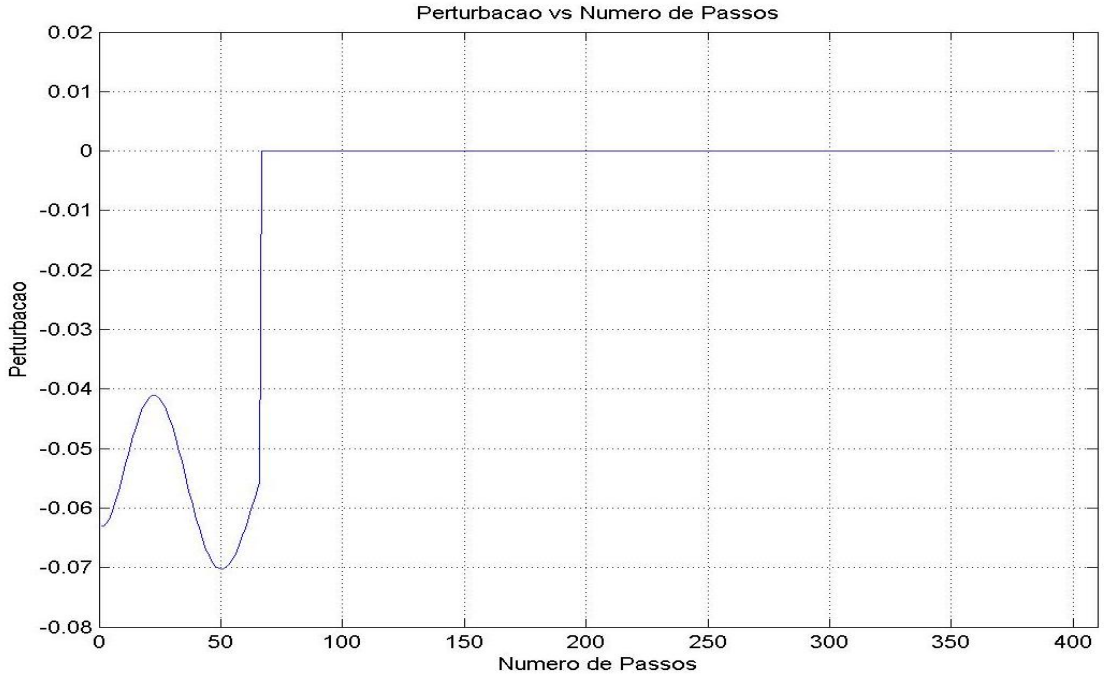

Figura 16: Sistema de Lorenz – Perturbação do Sinal pelo Número de Passos

É possível ver que ao atingir a estabilidade, a perturbação sobre o sinal cessa. Isso é evidenciado pela linha horizontal presente na figura.

Agora, no intuito de determinar qual foi a freqüência resultante do controle periódico, ou seja, conferir a freqüência estipulada no programa com a obtida na execução do mesmo, foi traçado o gráfico da figura 17 onde se tem a transformada de Fourier do sinal medido em função do período estipulado.

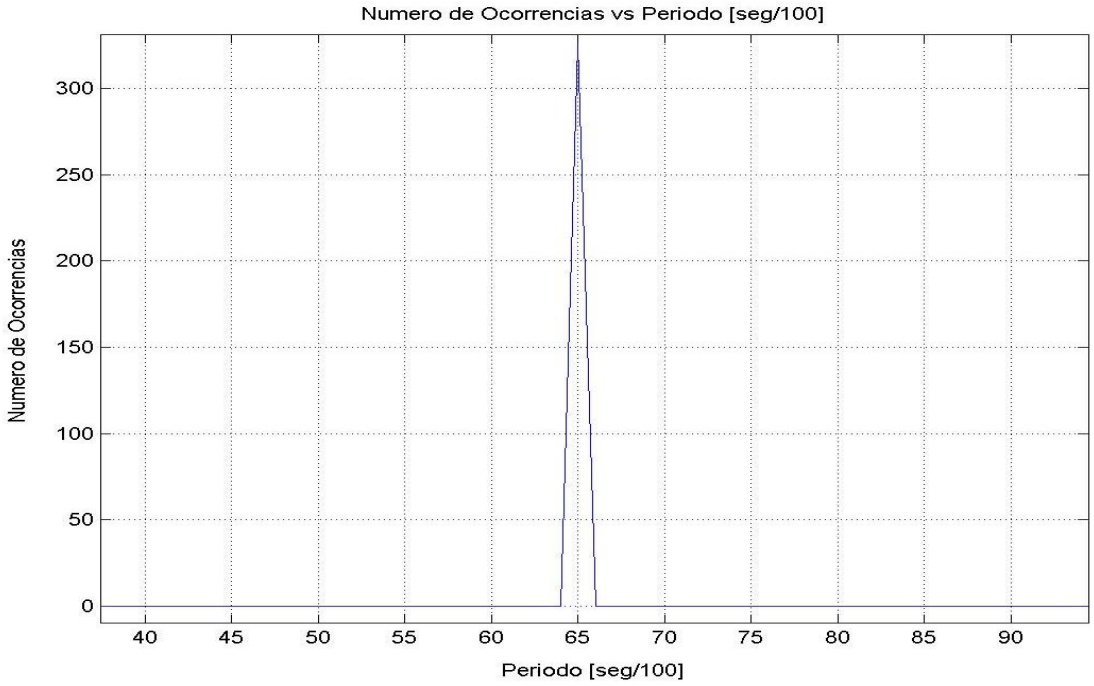

Figura 17: Número de ocorrências de um período pelo valor do período [seg/100]

É possível ver que a maioria dos passos do programa, mais de 80%, na iteração em que se atingiu a estabilidade, encontra-se com o período estipulado de 65 passos, ou seja, 650ms, onde cada passo tem a duração de dez milissegundos. Esse resultado é considerado excelente, pois indica que o controle da freqüência foi efetivo.

6.1.1 - Estabilizando o Sistema com Intervalos de Controle Desligado

Já que a rede de controle já foi obtida, é possível testar a mesma para estabilizar o sistema com intervalos de controle desligado. O esquema deste teste é representado pela figura 18.

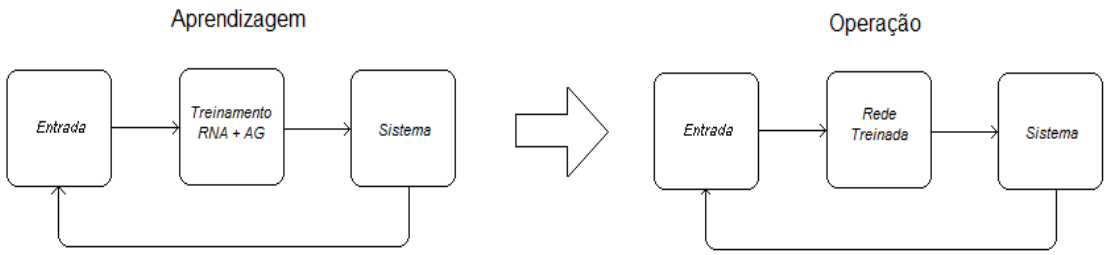

Figura 18: Esquema utilizado durante a execução do teste com o controle desligado.

Para este teste foi traçado o gráfico do sinal medido pelo número de passos com o controle desligado por noventa (90) passos, entre os passos 3610 e 3700, visto na figura 19. Este intervalo foi escolhido, pois durante estes passos o sistema já estava sobre a órbita periódica selecionada aguardando o tempo ajustado para que o controle seja considerado efetivo, 5 vezes o período da órbita, no caso 65 passos.

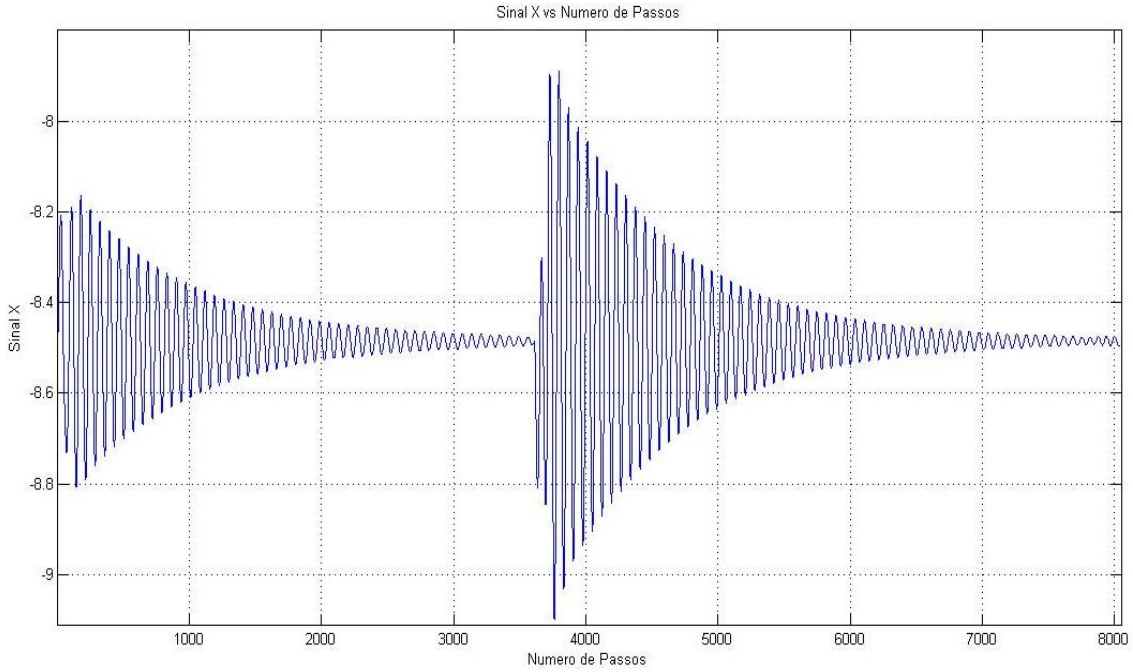

Figura 19: Estabilizando o sistema com sinal de controle intermitente.

Pode ser notado que, mesmo com o controle desligado por noventa (90) passos, entre os passos 3610 e 3700, atingiu-se a estabilidade com 8036 passos. Isso ocorreu, pois, mesmo com o controle desligado por um intervalo de tempo, ao ser religado, a RNA treinada consegue garantir que o sistema se estabilize sobre a órbita periódica selecionada.

6.1.2 - Estabilizando o Sistema Acrescentando Ruído no Sinal Medido

O ruído é um fator constante nos sistemas naturais, na prática dificilmente consegue-se obter experimentos onde não seja necessário considerar o ruído para atingir seus objetivos de forma mais confiável. Pensando nisso foi feito o teste no qual a rede de controle obtida foi aplicada ao sistema acrescentado de ruído em suas medidas, conforme o esquema mostrado na figura 20.

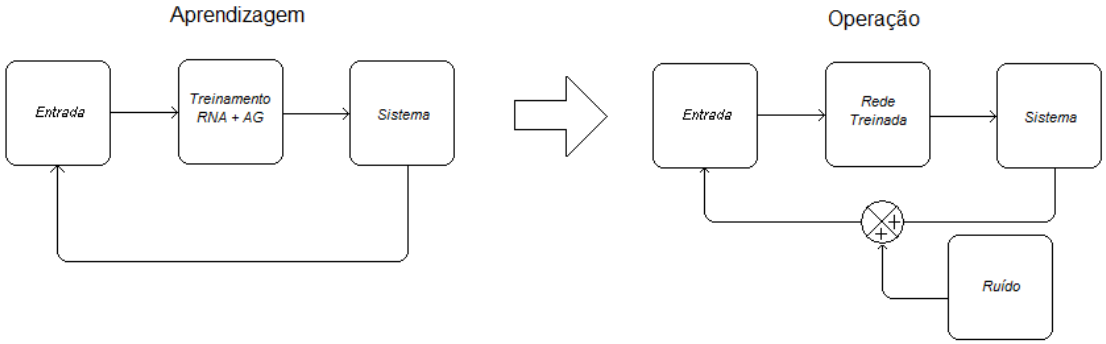

Figura 20: Esquema utilizado durante o teste com acréscimo de ruído no sinal medido.

Com ruído de 5% da amplitude média do sinal controlado e aplicando o controle intermitente, ou seja, desligando o controle por um intervalo determinado de tempo, neste caso entro os passos 3610 e 3700, tem-se o gráfico da figura 21.

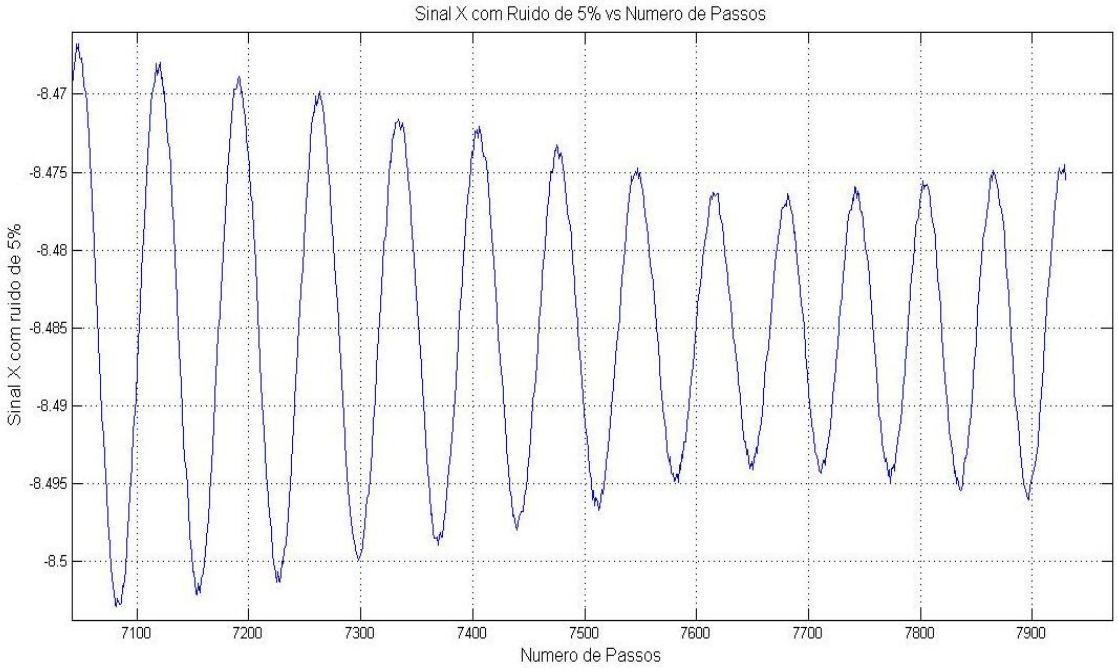

Figura 21: Rede Treinada – Sinal medido com ruído de 5%, com controle intermitente

A fim de visualizar com mais detalhes o intervalo no qual a estabilidade foi atingida foram omitidos os passos que antecedem o passo 7000. Observa-se que o ruído não afetou a capacidade da rede treinada em estabilizar o sistema sobre a órbita periódica desejada. A estabilidade foi atingida em 7929 passos.

Deste modo testar-se-á um ruído de 10% da amplitude média do sinal, figura 22, para verificar a resposta do sistema de controle a tal perturbação.

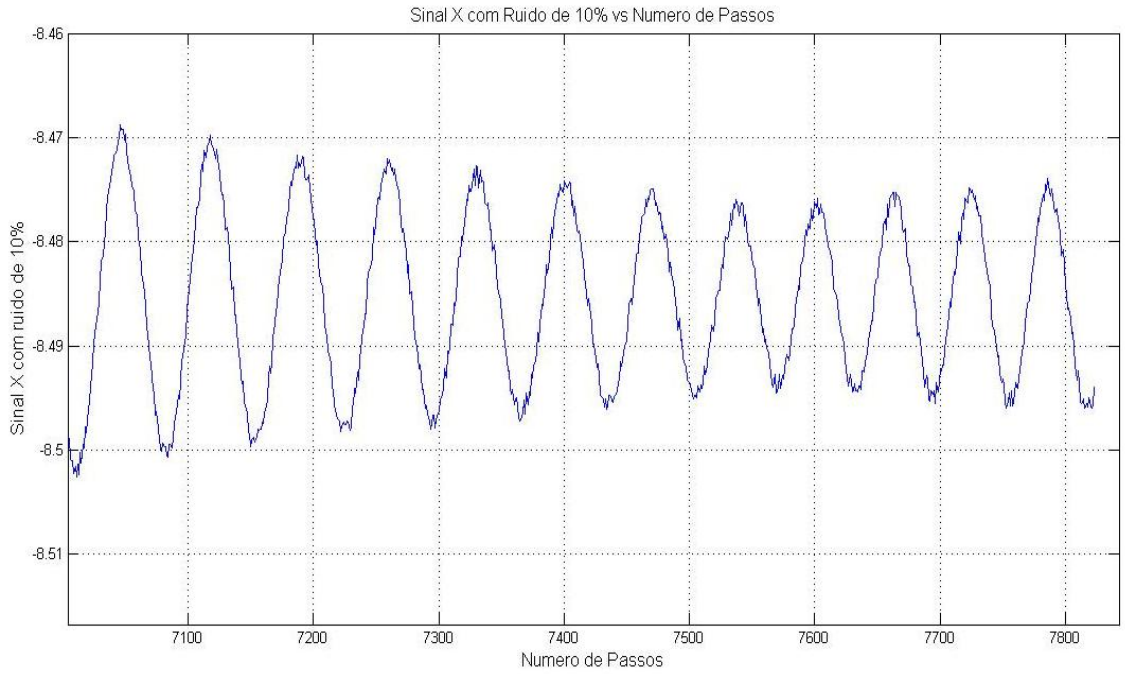

Figura 22: Rede Treinada – Sinal medido com ruído de 10%, controle intermitente

Novamente o ruído não foi capaz de impedir que o sistema fosse estabilizado, desta vez com 7823 passos.

Ainda no intuito de verificar a imunidade do sistema ao ruído, o gráfico da fiugura 23 foi traçado, agora com 15% de ruído sobre a saída.

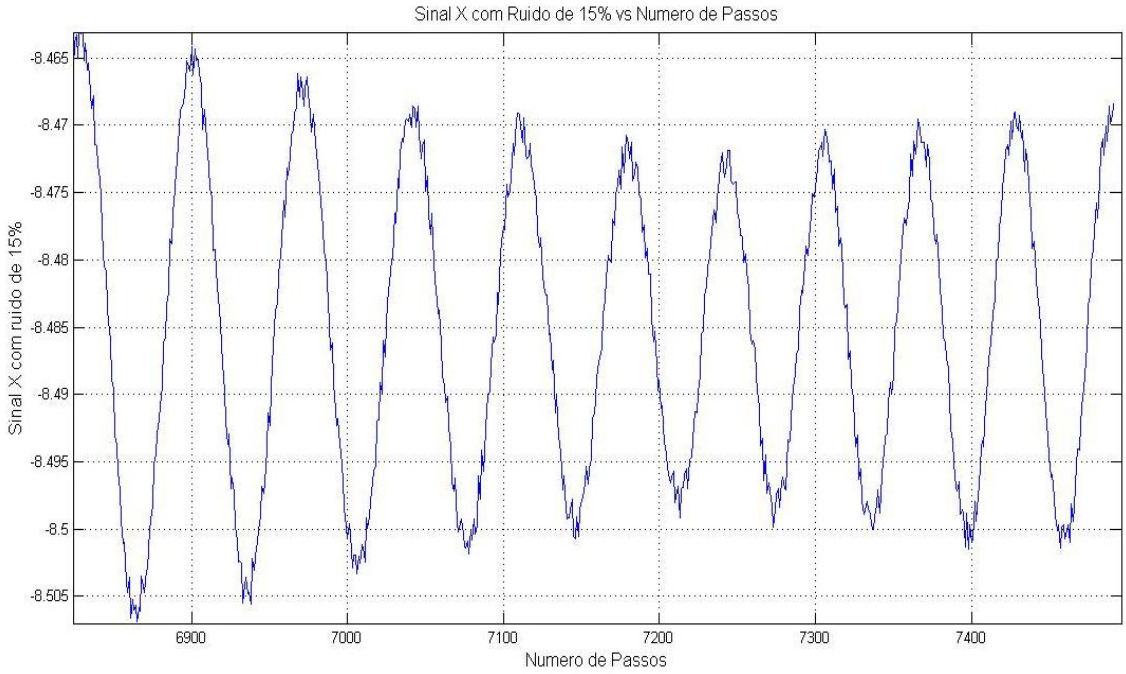

Figura 23: Rede Treinada – Sinal medido com ruído de 15%, controle intermitente

A interferência do ruído fica evidente com 15% da amplitude do sinal medido, porém com os critérios de estabilidade utilizados ainda foi possível atingir a mesma com 7490 passos. Isso mostra a robustez do programa desenvolvido e da rede de controle na estabilização do sistema sobre órbitas periódicas.

6.1.3 - Estabilizando o Sistema Variando as Condições Iniciais

Uma característica inerente aos sistemas caóticos é a sensibilidade crítica às condições iniciais. Dessa forma nisso foram feitos testes com algumas condições iniciais diferentes no intuito de verificar como o programa desenvolvido se comporta.

A figura 24 mostra a rede controlando o sistema com as condições iniciais sobre um dos pontos fixos não triviais de Lorenz ($-\sqrt{72}, -\sqrt{72}, 27$).

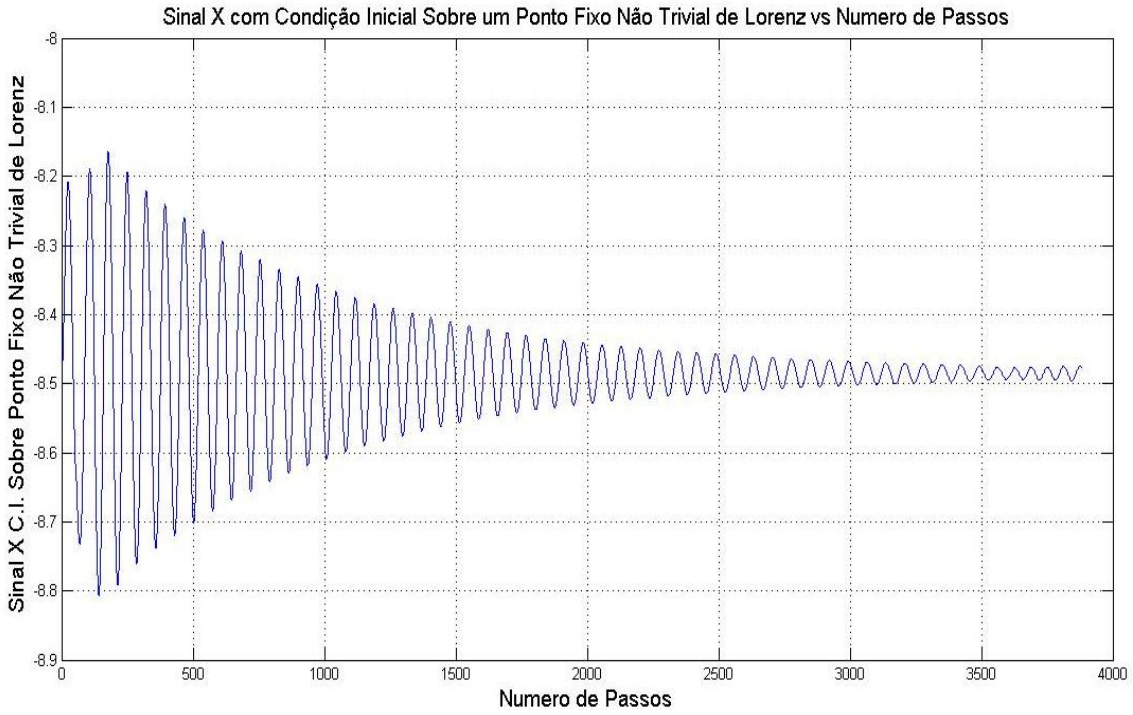

Figura 24: Sinal medido, condição inicial sobre um ponto fixo não trivial de Lorenz

É possível observar que quando as condições iniciais coincidem com o ponto fixo não trivial do sistema de Lorenz, a rede de controle leva o sistema a convergir para a órbita periódica desejada com 3880 passos. Este teste foi realizado utilizando uma rede treinada anteriormente.

A seguir traçar-se-á o mesmo gráfico, da figura 25, com as condições iniciais até 20% (vinte por cento) fora do ponto fixo não trivial de Lorenz. Essa porcentagem é calculada sobre o módulo do estado da variável X do ponto fixo não trivial de Lorenz, ou seja,$\left|-\sqrt{72}\right|$.

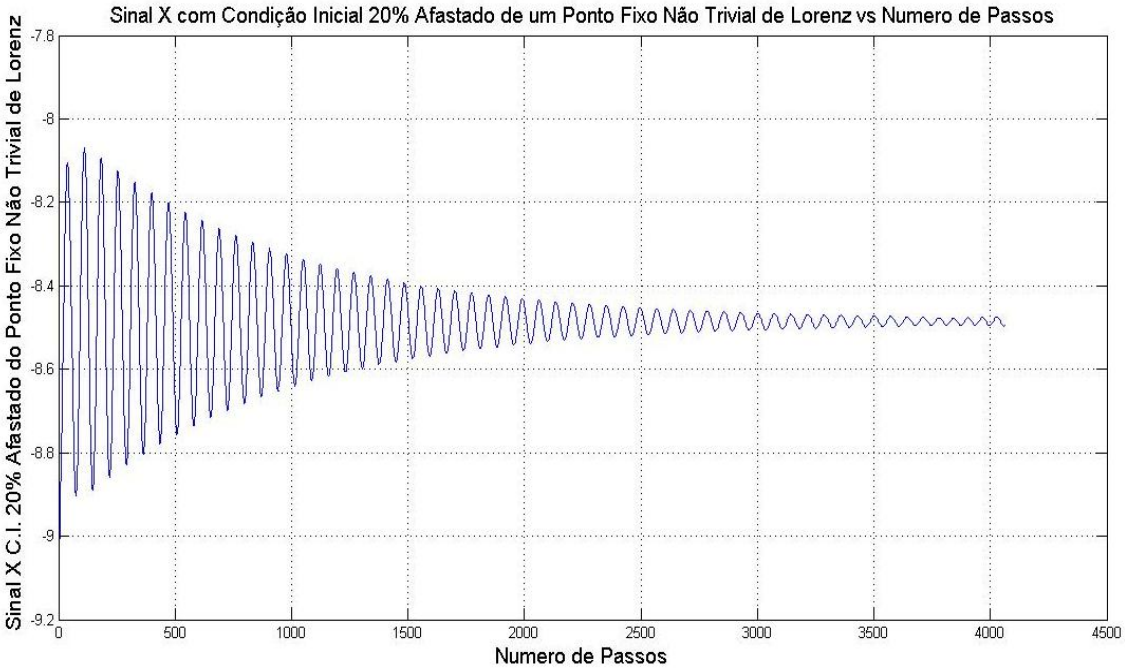

Figura 25: Sinal medido, C.I. 20% afastada de ponto fixo não trivial de Lorenz

Esta alteração nas condições iniciais fez com que fossem necessários 4062 passos para a rede treinada controlar o sistema sobre a órbita periódica desejada, ou seja, alguns passos a mais do que na situação anteriormente testada.

O próximo teste, mostrado na figura 26, foi realizado submetendo o sistema a condições iniciais até 50% (cinquenta por cento) afastado do ponto fixo não trivial de Lorenz mencionado anteriormente. Essa porcentagem foi calculada utilizando o mesmo critério do teste anterior.

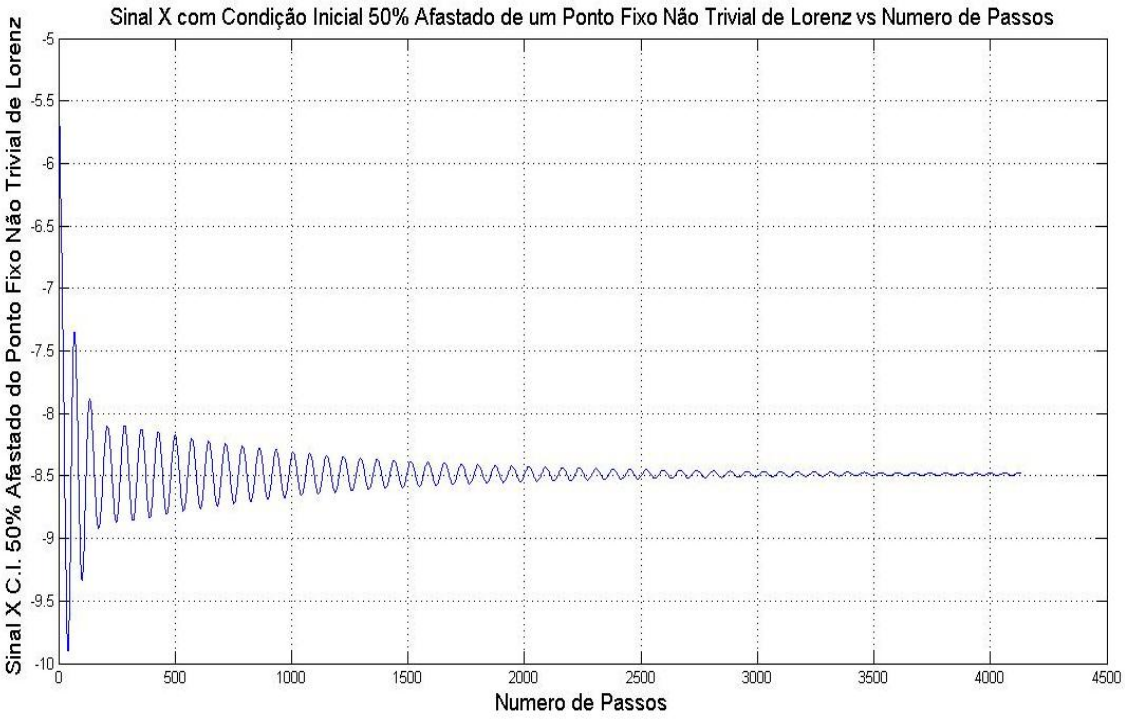

Figura 26: Sinal medido, cond. inic. 50% afastada de ponto fixo não trivial de Lorenz

Esta alteração nas condições iniciais alterou visivelmente o comportamento do sistema visto no gráfico. Isso fica evidenciado também pela quantidade de passos necessária para se obter o controle do sistema ter aumentado para 4128.

Como último teste da influência das condições iniciais na amplitude do sinal controlado testou-se o controle com condições iniciais até 70% (setenta por cento) afastados do ponto fixo não trivial de Lorenz, figura 27.

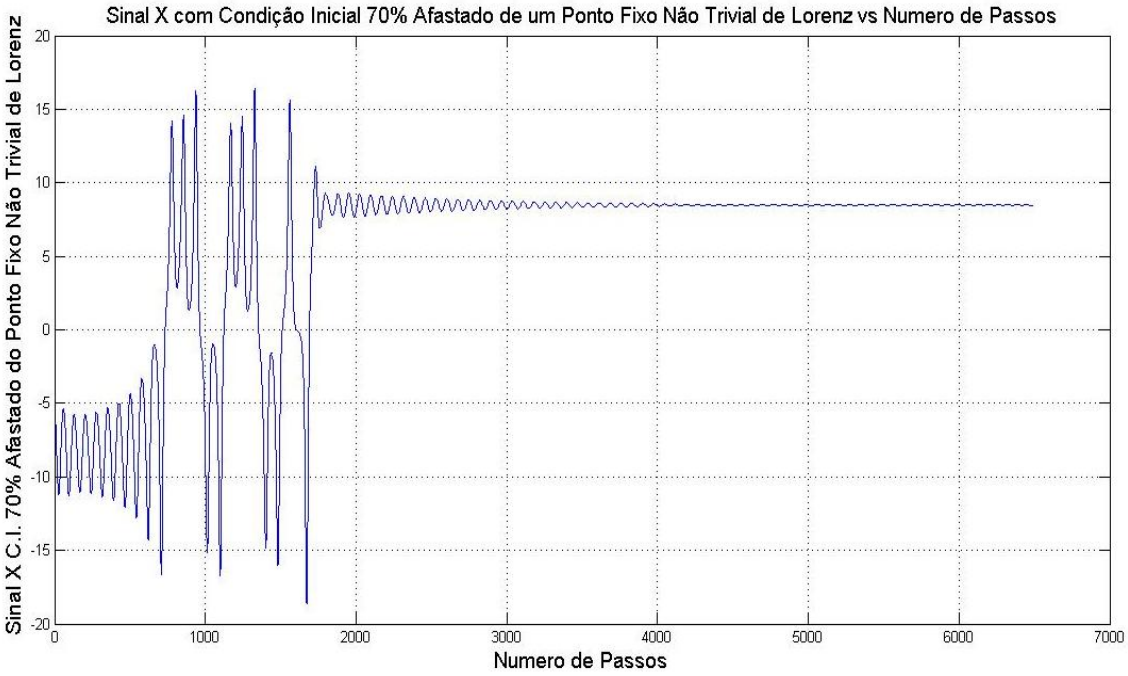

Figura 27: Sinal medido, C.I. 70% afastada de ponto fixo não trivial de Lorenz

Mais uma vez o resultado obtido condiz com o esperado, o número de passos necessários para controlar o sistema aumentou para 6485 visto que as condições iniciais foram afastadas ainda mais do ponto fixo não trivial de Lorenz. No gráfico é possível ver com clareza que o comportamento do sistema no princípio da iteração tende ao caos, porém a rede treinada foi capaz de controlar o sistema sobre a órbita periódica desejada mostrando seu poder de adaptabilidade mesmo sob condições mais exigentes.

6.1.4 - Estabilizando o Sistema – Treinamento da Rede com Ruído no Sinal Medido

Outra questão interessante é treinar a rede já com ruído nas medidas de seu sinal, isso dificulta o trabalho de encontrar uma rede de controle satisfatória, conforme esquema da figura 28.

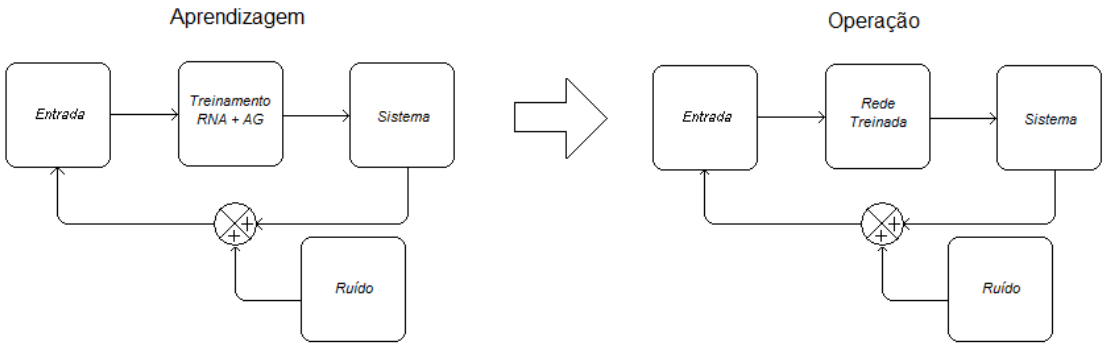

Figura 28: Esquema usado no teste de treinamento da rede com ruído no sinal medido.

O gráfico da figura 29 foi feito para mostrar a resposta do programa a essa solicitação, iniciando com ruído de 1% da amplitude média do sinal controlado.

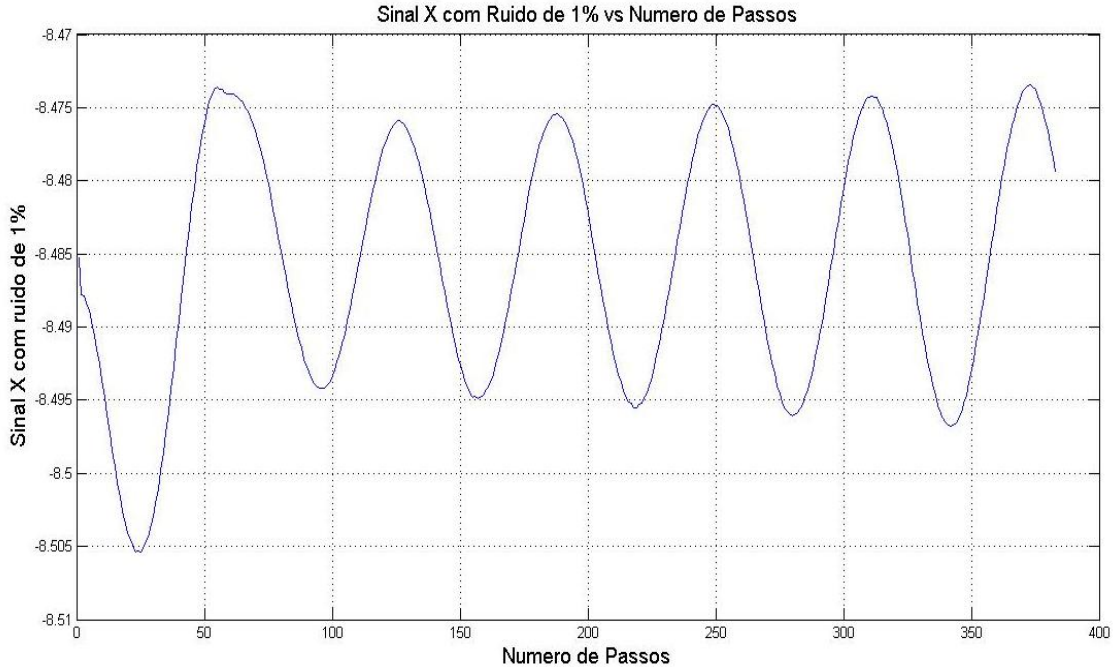

Figura 29: Comportamento do sistema controlado para uma rede treinada com 1% de ruído nas medidas

Ainda na primeira geração, na iteração 4, depois de 382 passos, foi atingida a estabilidade do sistema sobre a órbita desejada, conforme observado na figura 30. Isso mostra que mesmo em situações com ambientes mais exigentes o programa consegue encontrar uma rede de controle satisfatória.

Para reafirmar essa capacidade o gráfico da figura 30 foi elaborado com ruído de 5% da amplitude média do sinal controlado.

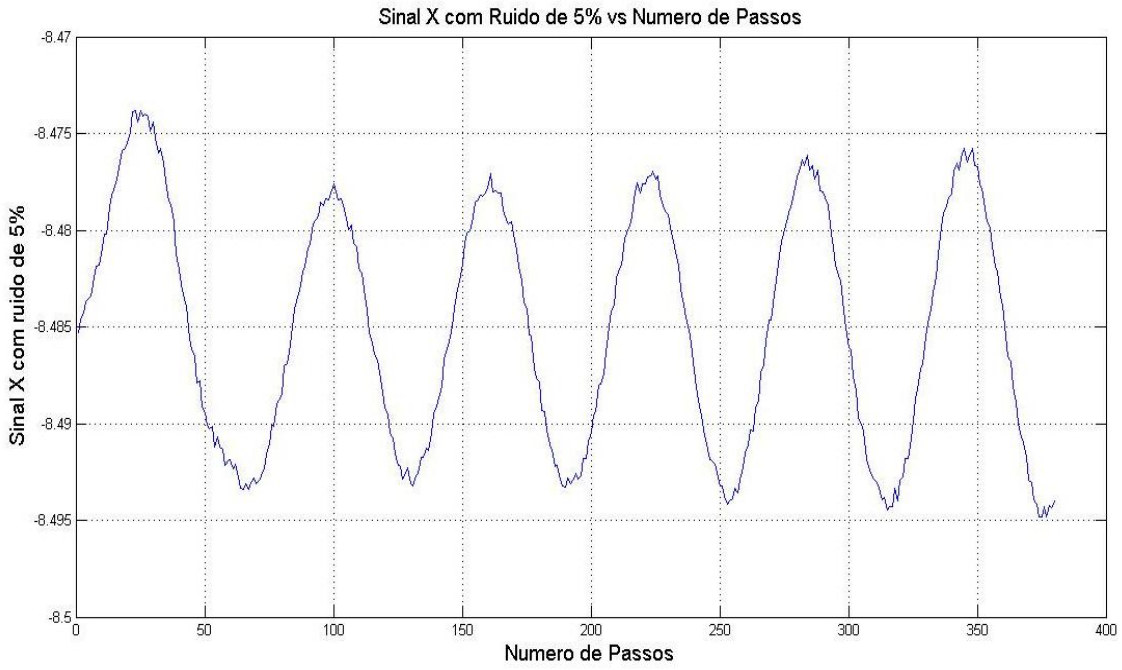

Figura 30: Comportamento do sistema controlado para uma rede treinada com 5% de ruído nas medidas

Com essa porcentagem de ruído o programa levou um pouco mais de tempo para encontrar a rede adequada para controlar o sistema, foram gastas 1 geração, 5 iterações e 379 passos.

Como último teste de treinamento da rede de controle já com ruído inserido no sinal do sistema, foi construído o gráfico, da figura 31, para ruído de 10% da amplitude média do sinal controlado.

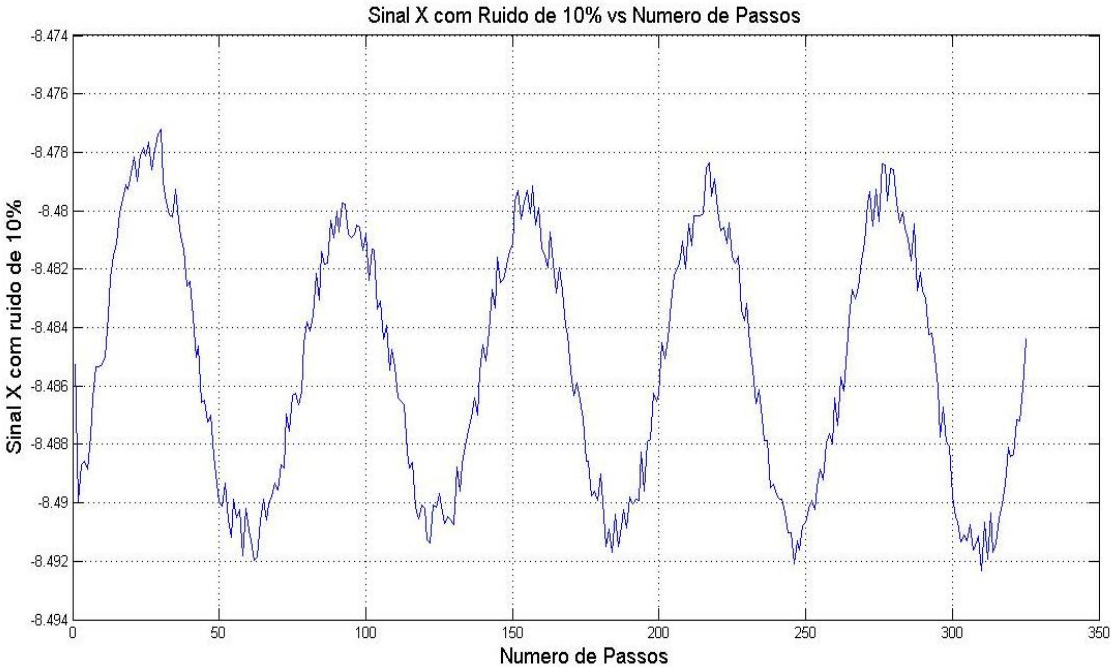

Figura 31: Comportamento do sistema controlado para uma rede treinada com 10% de ruído nas medidas

O sistema foi controlado sobre a órbita periódica com 3 gerações, depois de 8 iterações e 324 passos. Isto mostra o quão robusto é o método de controle adaptativo implementado quando submetido a situações adversas como a presença de ruído durante o treinamento da rede de controle.

7 - CONCLUSÕES

Nessa dissertação de mestrado foi desenvolvido um software que tem a função de controlar sistemas instáveis, lineares ou não, utilizando como método de controle redes neurais artificiais em conjunto com algoritmos genéticos.

Redes neurais artificiais, juntamente com Algoritmos genéticos, apresentaram uma grande capacidade de aprendizado e adaptação, constituindo um sistema de controle flexível e eficiente que não necessita de informações prévias da planta a ser controlada, e respondem bem a sistemas com características complexas, diferentemente de controles mais comumente usados, como o PID, como:

- Não-linearidades ou incertezas no modelo;
- Medidas apresentando incertezas e inadequação;
- Necessidade de tratamento de restrições ambientais;
- Ambientes variantes no tempo e que apresentam incertezas;
- Mudanças nas variáveis externas não-medidas e não-controláveis, como variações de temperaturas e pressão no ambiente.

Essa característica incomum é própria de sistemas adaptativos, em especial das RNAs e dos AGs, e tem proporcionado atualmente grande interesse no estudo e desenvolvimento de métodos de controle baseados nesses tipos de sistemas.

Durante os testes realizados com o software utilizou-se uma rede treinada, e nenhum treinamento adicional foi necessário. Assim, é possível notar o quanto o programa responde bem à sua função de controlar o sistema de Lorenz em várias situações diferentes, como:

- Com intervalos nos quais o controle está desligado, dificultando o controle do sistema por tornar seu estado, quando o controle for religado, cada vez mais imprevisível.
- Com a presença de ruído nas medidas após encontrar a rede de controle que leva ao comportamento periódico desejado, fazendo com que a incerteza gerada pelo ruído presente nas medidas exija muito mais adaptabilidade da rede de controle encontrada.
- Com a presença de ruído nas medidas antes mesmo de encontrar uma rede de controle capaz de promover o comportamento periódico do sinal caótico, o que gera

uma incerteza que se propaga a cada passo durante a execução do programa, fazendo com que o objetivo de encontrar uma rede de controle adequada para garantir o comportamento periódico desejado se torne uma tarefa mais difícil e trabalhosa. A inserção de ruído no sinal simula ruídos brancos e limitações de hardware do sistema real, que podem ser decisivas para o sucesso da implementação mecânica de sistemas de controle simulados.

- Alteração na situação inicial do sistema a ser controlado. Fazendo o sistema ser iniciado em uma posição distante de um de seus pontos não triviais, a RNA ainda assim é capaz de responder de forma que se chegue ao estado de controle do sistema sobre a órbita periódica selecionada.

O software apresentou respostas ótimas ao ser aplicado em sistemas instáveis em regime permanente (sistemas caóticos), ao ser utilizado para estabilizar o sistema de Lorenz (atrator de Lorenz).

Os resultados obtidos comprovam a capacidade de aprendizado do sistema implementado, e mostram de forma clara o quão robusto é o método utilizado para o projeto do sistema de controle.

Como trabalhos futuros pode-se fazer uma análise comparativa detalhada entre métodos de treinamento de RNAs, como o uso de AGs e o uso de back-propagation. Outro ponto que pode ser estudado é a implementação de um sistema real de pêndulo invertido para que seja possível testar na prática o funcionamento do sistema de controle implementado. Um terceiro ponto de estudo seriam comunicações digitais não lineares usando caos. Existem duas abordagens diferentes para o problema. Uma é usar o princípio do caos síncrono para aderir e transmitir informações digitais. O outro é estender o princípio de controle do caos para sistemas dinâmicos com dinâmicas simbólicas bem definidas para codificar informação.

REFERÊNCIAS BIBLIOGRÁFICAS

BASSO, M.; GENESIO, R.; GIOVANARDI, L.; & TESI, A. ***Frequency domain methods for chaos control. In Controlling chaos and bifurcations in engineering systems*** *(ed. G. Chen), p. 179–204. Boca Raton, FL: CRC Press, 1999*

BOCCALETTI, S.; GREBOGI, C.; LAI, Y.; MANCINI, H.; MAZA, D. ***The control of chaos: Theory and applications****. Physics Reports 329, p.103-197, 2000.*

BRAGA, A.; LUDERMIR, T.; CARVALHO, A.. ***Redes neurais artificiais****: teoria e aplicações. Rio de Janeiro: LTC, 2000.*

CASAS, F.; & GREBOGI, C. ***Control of chaotic impacts****. Int. J. Bifurcat. Chaos 7, 951–955,1997*

CHAU, N. P. ***Controlling chaos by periodic proportional pulses****. Phys. Lett. A 234, 193–197, 1997.*

CHEN, G.; & LIU, Z. ***On the relationship between parametric variation and state feedback in chaos control.*** *Int. J. Bifurcat. Chaos 12, 1411–1415, 2002a*

CHEN, L. Q.; & LIU, Y. Z. ***Chaotic attitude motion of a magnetic rigid spacecraft and its control****. Int. J. Nonlin. Mech. 37, 493–504, 2002b.*

DIEDERICH, J. ***Artificial neural networks****: Concept learning. Los Alamitos: IEEE, 1990.*

FIEDLER-FERRARA, N.; PRADO, C. P. C. ***Caos****: uma introdução. São Paulo: Edgard Blücher, 1994.*

FRADKOV, A. L. **Speed-gradient scheme in adaptive control**. Autom. Remote Control 40, 1333–1342,1979.

FRADKOV, A. L.; & POGROMSKY, A. Y. **Introduction to control of oscillations and chaos**. Singapore: World Scientific,1998.

FRADKOV, A. L. **Nonlinear adaptive control: regulation–tracking–oscillations**. In Proc. 1st IFAC Workshop New Trends in Design of Control Systems, Smolenice, Slovakia, pp. 426–431, 1994.

FRADKOV, A. L.; & EVANS, R. J. **Control of chaos: survey 1997–2000**. In Prepr. 15th IFAC World Congress on Automatic Control. Plenary papers, Survey papers, Milestones, Barcelona, pp. 143–154, 2002.

FRADKOV, L.; EVANS, E. R.; ANDRIEVSKY, B. R. *Control of chaos: methods and applications in mechanics.* Phil. Trans. R.Soc. A 364, p. 2279-2307, 2006.

GOLDBERG, D. E. ***Genetic algorithms in search, optimization and machine Learning****. [USA]: Addison-Wesley, 1989.*

GRASSI, G.; & MASCOLO, S. ***Nonlinear observer design to synchronize hyperchaotic systems via a*** **scalar signal**. IEEE Trans. Circ. Syst. I 44, 1011–1014, 1997.

JACKSON, E. A. & GROSU, I. ***An OPCL control of complex dynamic systems.*** *Physica D 85, 1–9, 1995*

KHOVANOV, I. A., LUCHINSKY, D. G., MANNELLA, R. & MCCLINTOCK, P. V. E. ***Fluctuations and*** **the energy-optimal control of chaos.** Phys. Rev. Lett. 85, 2100–2103, 2000.

KOZA, J. R.; KEANE, M. A.; STREETER, M. J. Aperfeiçoando inventos. ***Scientific American Brasil****, São Paulo, v. 2, n. 10, p. 61-67, mar. 2003.*

LIAO, T. L. ***Observer-based approach for controlling chaotic systems****. Phys. Rev. E 57, 1604–1610, 1998.*

LINDEN, R. ***Algoritmos genéticos****: uma importante ferramenta da inteligência computacional. Rio de Janeiro: Brasport, 2006.*

MEUCCI, R.; LABATE, A.; & CIOFINI, M. ***Controlling chaos by negative feedback of subharmonic components****. Phys. Rev. E 56, 2829–2834, 1998.*

MORGUl, O.; & SOLAK, E. ***On the synchronization of chaotic systems by using state observers****. Int. J. Bifurcat. Chaos 7, 1307–1322,1997.*

NIJMEIJER, H.; & MAREELS, I. M. Y. ***An observer looks at synchronization.*** *IEEE Trans. Circ. Syst. I 44, 882–890, 1997.*

OTT, E.; GREBOGI, C.; YORKE, J. A. Controlling chaos. ***Physical Review Letters****, New York, v. 64, n. 11, p. 1196-1199, 1990.*

SAVI, M. A. ***Dinâmica não-linear e caos****. Rio de Janeiro: E-papers Serv. Edit., 304p. 2006.*

ZHAO, H.; WANG, Y. H.; & ZHANG, Z. B. ***Extended pole placement technique and its applications for targeting unstable periodic orbit****. Phys. Rev. E 57, 5358–5365,1998.*

APÊNDICES

Apêndice A – Comportamento da função sigmoidal.

Apêndice B – Teoria de esquemas.

Função sigmoidal original

A função sigmoidal tem a característica de receber entradas no intervalo $[-\infty, \infty]$ e gerar uma saída no intervalo [0,1], segundo a equação a seguir.

$$fy = \frac{1}{(1+e^{-y})}$$

Equação: Função sigmoidal original

A partir desta equação é possível traçar o gráfico de comportamento da função sigmoidal original. Tal gráfico está na fig.32.

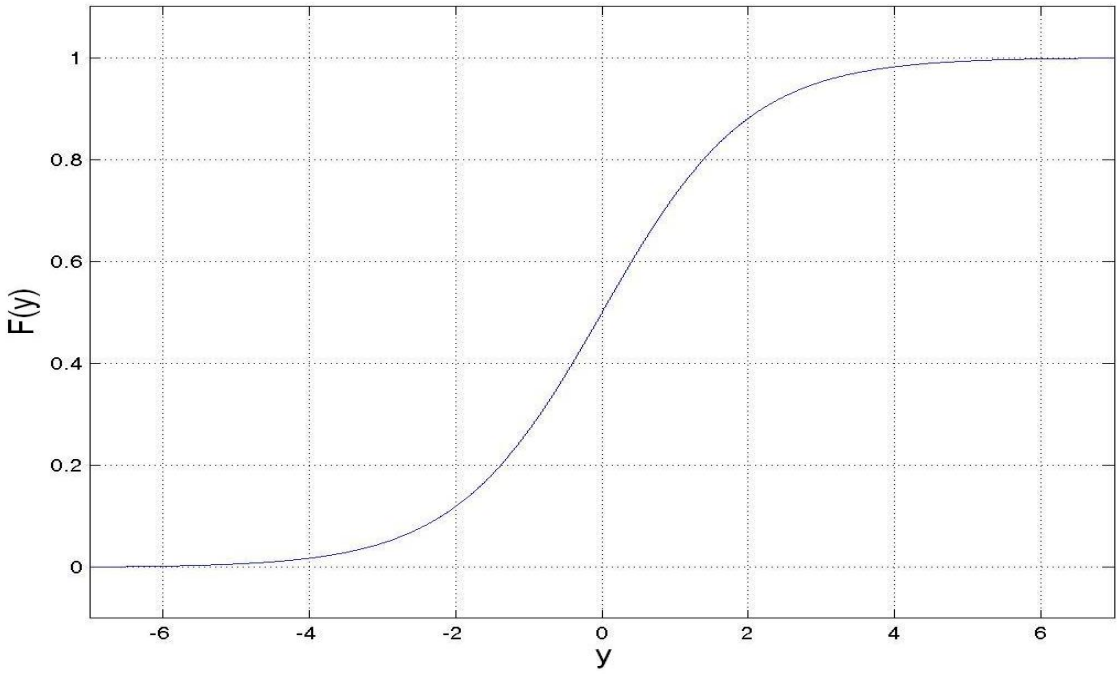

Figura 32: Gráfico do comportamento da função sigmoidal original.

Função sigmoidal usada no trabalho

Neste projeto alterou-se a função de transferência original da sigmoidal para que ela fornecesse saídas no intervalo [-1,1]. A função de transferência usada é:

$$fy = \frac{2}{(1+e^{-y})} - 1$$

Equação: Função sigmoidal usada no programa

A figura 33 mostra o comportamento da função sigmoidal utilizada no software. Esta função é uma versão modificada da função log-sigmoidal, de forma a produzir saídas no intervalo [-1,1], ao invés de [0,1] do modelo original.

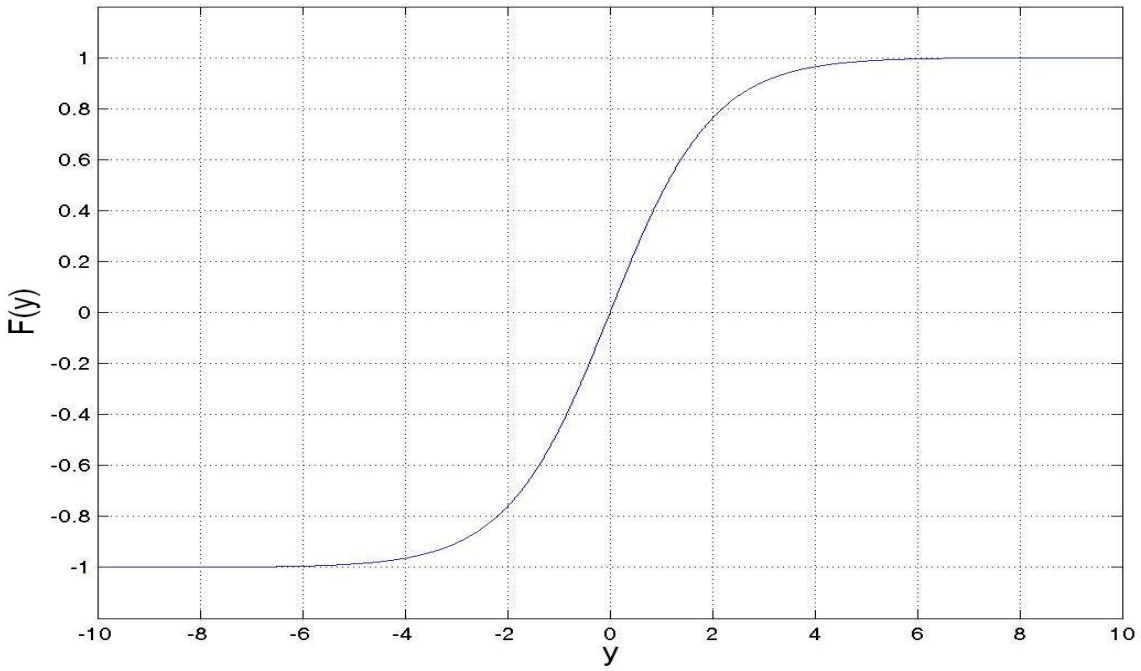

Figura 33: Gráfico do comportamento da função sigmoidal utilizada no programa.

APÊNDICE B – TEORIA DE ESQUEMAS

Dado que uma população é um conjunto de indivíduos codificados em uma determinada base comum, um esquema nada mais é do que um subconjunto dessa população formado por indivíduos que compartilhem alguma semelhança uns com os outros.

Imaginemos o exemplo de uma sala de aula na qual estão presentes 40 pessoas. Um possível esquema representaria todas as pessoas no local que possuem cabelos pretos.

O alfabeto de esquemas é formado pelos símbolos que são usados na representação dos indivíduos acrescido do símbolo *, o qual significa "não-importa", ou seja, que a informação contida na posição em questão não é importante no momento, ou para a análise que está sendo feita, e será temporariamente ignorada. Voltando ao exemplo da sala de aula, para sabermos o número de pessoas que possuem cabelos pretos não há necessidade de se analisar separadamente homens e mulheres, de forma que o esquema que representa a característica de interesse (cabelos pretos), certamente conterá um * na posição destinada à distinção de sexo, caso esta posição exista na representação da população.

Tendo em mente uma população com representação binária, um dado esquema de tamanho x e com y posições com * representará, conseqüentemente, 2^y indivíduos diferentes desta população (Linden, 2006). A tabela 1 apresenta alguns exemplos de esquemas e indivíduos representados por esses, para uma população com representação binária.

Tabela 1: Exemplos de esquemas e os indivíduos por ele representado.

Esquema	Indivíduos representados
10101	10101
1*1*1	10101, 10111, 11101, 11111
**	00, 01, 10, 11
*1	01, 11

É fácil ver que se o alfabeto da população for diferente do binário, suponhamos um que contenha S símbolos, e o esquema analisado possua P posições contendo o símbolo *, o esquema em questão será capaz de representar $(S-1)^P$ indivíduos.

De forma semelhante é possível chegar ao número de esquemas possíveis em um determinada população, o qual é dado pelo número total de subconjuntos possíveis de serem formados com tal população.

Enxergar a importância dos esquemas para os AGs, é chegar mais próximo de uma explicação de porque esse método adaptativo é bem sucedido em problemas de convergência até mesmo para sistemas instáveis não-lineares. Tal explicação não é trivial e está ligada ao fato de que os esquemas carregam características dos indivíduos que são representados por ele e tais características são determinantes para o julgamento, por parte do AG, se tal esquema deve ser descartado ou mantido para uma nova geração.

Ao analisar um indivíduo o AG está, na verdade, analisando vários esquemas simultaneamente e automaticamente selecionando os melhores entre eles. Essa peculiaridade dos AGs é denominada paralelismo implícito, e implica que nos processos genéticos, aplicados posteriormente na população, os esquemas melhor adaptados se reproduzam mais do que os outros menos adaptados, permanecendo, conseqüentemente, por mais tempo na população (Linden, 2006).

Essa análise nos permite ver com mais clareza o foco que os AGs dão aos esquemas em detrimentos dos indivíduos por si só, pois mesmo que um indivíduo bom seja descartado, o esquema que contém as características que assim o fizeram, é mantido e aperfeiçoado de modo a levar a resultados melhores.

Existem duas informações fundamentais de um esquema, que são seu tamanho $\zeta(H)$ e sua ordem $O(H)$. A ordem de um esquema é dada pelo número de posições deste que diferem do símbolo *, e o tamanho de um esquema é dado pelo número de pontos de corte entre a primeira e a última posições que diferem de * dentro do esquema. A tabela 2 apresenta exemplos de esquemas, seus respectivos tamanho e ordem.

Tabela 2: Exemplos de esquemas e seus respectivos tamanho e ordem.

Esquema	Ordem	Tamanho
1	1	0
0*****1	2	6
01*1	3	5
010101	6	5

Analisando os esquemas da tabela 2, temos que o esquema (***1***) só possui um elemento diferente de *, fazendo com que sua ordem seja igual a um, e seu tamanho seja igual a zero. Para o esquema (0*****1), tem-se ordem igual a dois, visto que tal esquema possui dois elementos diferentes de *, e tamanho seis, dado que é possível cortar o esquema em seis pontos diferentes entre suas primeira e última posições diferentes de *. A análise dos demais é dada da mesma maneira e por esse motivo será omitida.

Algo cuja análise é de grande importância é os efeitos do cruzamento em esquemas. Para garantir uma visualização mais fácil do que se deseja expor, a tabela 3 contém exemplos de esquemas que supostamente passarão por cruzamento, as posições de corte de cada um (marcadas pelo símbolo |) e o estado do esquema após concretizado tal processo, este último campo pode assumir duas situações, mantido ou destruído.

Tabela 3: Exemplos de esquema a serem cruzados, seus pontos de corte e seu estado após cruzamento.

Esquema	Situação depois do corte
0111 | ****	Mantido
0* | **	Mantido
1* | *1	Destruído
0*1* | ***0	Destruído

Para o esquema da primeira linha, é fácil ver que, como há o símbolo * em todas as posições depois da posição de corte, significa que tal esquema será mantido integralmente após o processo de cruzamento ser concretizado, visto que qualquer

indivíduo que contenha as quatro primeiras posições iguais a (0111) farão parte do esquema original, independentemente do conteúdo de suas quatro últimas posições.

O mesmo é válido para o esquema da segundo linha, com a diferença que este último é ainda mais genérico que o primeiro, pois possui um * também antes do ponto de corte, fazendo-o capaz de representar duas vezes mais indivíduos que o esquema mostrado na linha um.

Para o esquema apresentado na terceira linha tem-se um resultado diferente dos anteriores. Como esse esquema possui símbolos diferentes de * em ambos os lados do ponto de corte, para que tal esquema sobreviva após o processo de cruzamento são necessárias condições específicas, neste caso, que o indivíduo que vai cruzar com tal esquema possua o valor um (1) na terceira e/ou na sexta posições de sua representação, para que um ou os dois esquemas não sejam destruídos pelo processo de cruzamento. Como não é possível garantir tais restrições, nesses casos considera-se que o esquema original é destruído. O mesmo é válido para o esquema apresentado na quarta linha da tabela 3, tal conclusão pode ser atingida através de análise similar à anteriormente descrita.

Seguindo o mesmo raciocínio pode-se chegar à contribuição do processo de mutação para a destruição de um esquema. Nessa análise o importante deixa de ser o tamanho do esquema e passa a ser sua ordem. Isso porque, como a mutação atua, no sentido de modificar, sobre uma única posição do esquema, quanto maior o número de posições diferentes de * no esquema, maior será a chance deste ser destruído por um processo de mutação. Como o símbolo * representa qualquer valor do alfabeto do esquema, uma mutação sobre uma posição que contenha um * certamente gerará outro indivíduo também representado pelo esquema original, fazendo com que esse último seja mantido íntegro após o processo de mutação.

A partir de todo o exposto nesse anexo, é possível chegar a uma conclusão sobre o tratamento de esquemas por parte dos AGs. É possível observar que algoritmos genéticos têm a tendência de manter esquemas de melhor desempenho, e com valores menores de tamanho e ordem como passar do tempo, manipulando-os de modo a buscar a solução mais apropriada, sem que seja realizada qualquer busca direcionada, seguindo o princípio de que indivíduos bem adaptados geram filhos no mínimo tão bem adaptados quanto os primeiros.