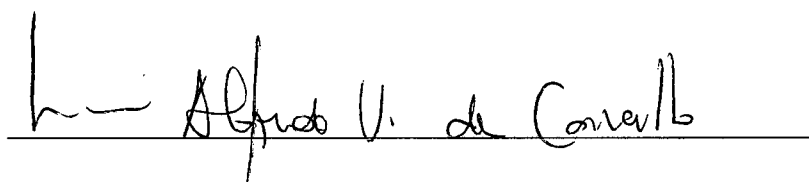


# RETROPROPAGAÇÃO RECORRENTE : APLICAÇÃO NA PREVISÃO DE SÉRIES TEMPORAIS E IDENTIFICAÇÃO DE ASSINATURAS SONORAS

RICARDO BARZ SOVAT

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

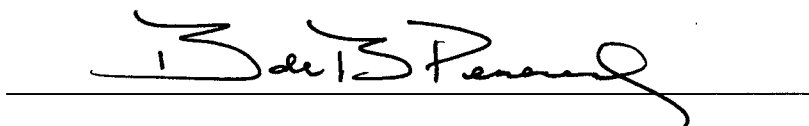
Aprovada por :



Prof. LUÍS ALFREDO VIDAL DE CARVALHO, D. Sc.  
(presidente)



Prof<sup>a</sup> NAIR MARIA MAIA DE ABREU, D.Sc.



Prof. BASÍLIO DE BRAGANÇA PEREIRA, PhD

RIO DE JANEIRO, RJ - BRASIL  
MAIO DE 1994

SOVAT, RICARDO BARZ

Retropropagação Recorrente : Aplicação na  
Previsão de Séries Temporais e Identificação de  
Assinaturas Sonoras (Rio de Janeiro) 1994.

x 153 p. 29,7 cm (COPPE/UFRJ, M.Sc.,  
Engenharia de Sistemas de Computação, 1994)

Tese - Universidade Federal do Rio de Janeiro,  
COPPE

1. Redes Neurais 2. Retropropagação

3. Previsão 4. Assinatura Sonora

I. COPPE/UFRJ II. Título

À memória de meu pai, Dick Sovat,  
meu incentivador e orientador em  
todos os estudos.

## AGRADECIMENTOS

Ao professor Luís Alfredo Vidal de Carvalho (COPPE/ UFRJ), orientador desta tese, pelo apoio e orientação fornecidos ao longo de todo o trabalho.

A minha esposa, Maria Cristina Dias Tavares, que consegue ao mesmo tempo ser companheira, profissional, digitadora e revisora, sem deixar faltar seu carinho e atenção de mãe, por mostrar que com determinação e sentimento tudo é possível.

A minha mãe, Zilda Barz, por seu carinho e compreensão constantes e pela presença confortadora em todas as horas difíceis.

À engenheira Cristina Maria Pinto Silva (LIGHT), por suas contribuições de ordem técnica, por seu incentivo e por todo o apoio na obtenção da bibliografia necessária.

A meu amigo Paulo Martins Magalhães, por toda ajuda prestada, desde o apoio pessoal até o empréstimo de equipamentos.

A meus sogros, Mario e Isa, a Wanda e Marisa pela ajuda na obtenção dos sinais de voz.

A todos meus amigos, pela paciência e compreensão durante a confecção deste trabalho.

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

## **RETROPROPAGAÇÃO RECORRENTE : APLICAÇÃO NA PREVISÃO DE SÉRIES TEMPORAIS E IDENTIFICAÇÃO DE ASSINATURAS SONORAS**

Ricardo Barz Sovat

Maio de 1994

Orientador : Prof. Luís Alfredo Vidal de Carvalho  
Programa : Engenharia de Sistemas e Computação

Neste trabalho são estudadas as características de redes neuronais que utilizam o paradigma de *backpropagation* recorrente, ou retropropagação no tempo, bem como é analisado o desempenho do modelo em dois tipos de aplicação : previsão de valores de séries temporais e identificação de uma determinada série.

No primeiro caso, séries possuidoras de características diferentes foram apresentadas a implementações do modelo de rede, que foi treinado para estimar seus valores futuros. Os resultados obtidos foram comparados àqueles obtidos através de algoritmos não conexionistas, como a Metodologia Box-Jenkins, e às previsões realizadas por *backpropagation* não recorrente. Os testes demonstram um desempenho igual ou superior aos demais métodos, além de apontar algumas correlações entre as topologias ótimas para os dois modelos de rede neuronal.

A capacidade de identificação de uma determinada série foi avaliada empregando-se dados de bancada : as séries temporais utilizadas foram seqüências de sinais de voz no domínio do tempo, obtidas diretamente pelo computador e processadas antes de serem apresentada à rede. As características levadas em consideração no sinal de voz foram as mesmas observadas pelos métodos analíticos convencionais, tendo a rede apresentado um desempenho comparável àqueles, sem contudo necessitar o mesmo volume de

processamento durante a identificação, já que grande parte desse trabalho é realizada durante o treinamento.

Mostra-se, assim que a capacidade de reconhecimento de um determinado padrão levando em consideração sua ocorrência em uma ordem numa série, confere a este modelo de rede uma boa capacidade associativa além de seu já conhecido poder de previsão. O fato de não ser necessário trabalhar-se com uma entrada de dados totalmente atrasada no tempo, ou seja, a série não é apresentada toda de uma vez à primeira camada, e sim trecho a trecho, além do cálculo relativamente rápido, credenciam o modelo a ser empregado em um processamento em tempo "quase real", isto é, de baixo tempo de resposta. O desempenho no processo de validação de voz sugere um possível aproveitamento deste tipo de rede neuronal no reconhecimento de outros tipos de assinaturas sonoras, como, por exemplo, ondas mecânicas provenientes de ensaios de impacto, e na identificação de fonemas, sendo que no segundo caso algumas alterações necessárias na implementação são sugeridas.

Abstract of Thesis presented to COPPE/UFRJ as partial fulfillment of the requirements for the degree of Master of Science (M.Sc.)

**RECURRENT BACKPROPAGATION : APPLICATION IN TIME  
SERIES FORECASTING AND SOUND SIGNATURES  
IDENTIFICATION**

Ricardo Barz Sovat

May, 1994

Chairman : Prof. Luís Alfredo Vidal de Carvalho  
Department : Systems and Computation Engineering

The features of a neural network that joins a recurrent dynamics and backpropagation are studied in this work, as well as the application of such model in two kinds of activities, time series forecasting and sound signatures identification, are described.

At the first application, the network was trained to make previsions of future values of a set of time series having different profiles. The results were compared both to those obtained by non-conexionist algorithms, like Box-Jenkins methodology, and the outputs of a conventional backpropagation network. The tests indicate a at least equal forecast power, also pointing some correlations between optimum topologies of the neuronal models.

The second part of the analysis was developed using laboratory data. Voice signals at the time domain were captured by the computer and utilized like inputs to the model, after some preprocessing. The same features used by traditional analytical methods were extracted from the voice samples and the network showed the ability to identify that kind of time sequence, without a great burden of processing during the recognition, once the main portion of the task is done at learning time.

It is shown that the recognition capacity of a defined patern, considering its occurrence in an order in a series, gives to this network model a good associative capacity

as well as its known forecasting power. The fact of not being necessary to handle an input data completely delayed in time, once the series is not presented as a whole to the first layer, but shot to shot, and also the relatively fast calculation, allows this model to be used as an almost real time processing, or better, of a short time answering. The performance of the voice verification process indicates a possible use of this type of neural network in the recognition of other type of sound signatures, as shock waves caused by input essays, and in phonems identifying, being necessary to implement some modifications for the latter case, that are suggested.



## ÍNDICE

I - INTRODUÇÃO	1
I.1 -Objetivo	1
I.2 -Estrutura do Trabalho	4
II - REDES NEURONAIS E RETROPROPAGAÇÃO	
II.1 -Redes Neurais Artificiais	6
II.2 -O Modelo <i>Backpropagation</i>	7
III - O MODELO DE REDE UTILIZADO : RETROPROPAGAÇÃO NO TEMPO	
III.1 -Inovações do Modelo e Características Principais	16
III.2 -Operação Básica	17
III.3 -Cálculo de Erros	20
III.4 -Treinamento	22
IV - APLICAÇÃO NA PREVISÃO DE SÉRIES TEMPORAIS	
IV.1 -Topologia Utilizada	25
IV.2 -Características das Séries Empregadas	26
IV.3 -Análise do Desempenho	27
V - RECONHECIMENTO DE VOZ	
V.1 - Sistemas de Reconhecimento de Voz	30
V.2 - Análise de Tom	32
V.3 - Cálculo da Energia	35
V.4 - Processamento do Sinal	37
VI - APLICAÇÃO EM RECONHECIMENTO DE VOZ	
VI.1 - Adequação do Modelo	40
VI.2 - Aquisição dos Dados	42
VI.3 - Preparação dos Dados	44
VI.4 - Rede Utilizada	45
VI.5 - Análise do Desempenho	48
VII - CONCLUSÕES E RECOMENDAÇÕES	55
REFERÊNCIAS BIBLIOGRÁFICAS	58

**APÊNDICE A**

Fontes dos Programas Utilizados	60
---------------------------------	----

**APÊNDICE B**

B.1 - Séries Utilizadas	98
B.2 - Tabelas de Resultados dos Testes	101
B.3 - Resultado da Identificação do Orador	127

**APÊNDICE C**

Especificação do Equipamento Envolvido	151
--	-----

**APÊNDICE D**

Formato dos Arquivos de Dados	152
-------------------------------	-----

## CAPÍTULO I

### INTRODUÇÃO

#### I.1 - Objetivo

O objetivo principal deste trabalho é desenvolver uma abordagem conexionista para a identificação de assinaturas sonoras. De início, o ambiente em que dar-se-ia este desenvolvimento seria o de um laboratório equipado com um dispositivo específico para ensaios de impacto em corpos de prova, denominado *barra Hopkinson*. Desejava-se que as ondas geradas por esse dispositivo fossem analisadas por uma simulação em software de um modelo de rede neuronal artificial, que buscaria classificá-las após uma sessão de treinamento. Questões de ordem prática, contudo, impediram que tal ambiente fosse montado, o que deixou dois caminhos possíveis para a continuação do estudo : o teste de modelos conexionistas utilizando-se curvas *simtéticas*, ou seja, sinais não extraídos de um experimento mas gerados pelo próprio computador, o que não se constitui no melhor método de validação disponível, ou realizar-se a análise de um tipo de assinatura análoga de mais fácil obtenção, porém provenientes de dados *reais*.

Optou-se pela segunda idéia, substituindo-se as assinaturas sonoras de impacto por sinais de voz. A montagem de uma bancada capaz de captar, digitalizar e armazenar ondas sonoras geradas pela fala mostrou-se relativamente simples e barata, tornando-se acessível dentro dos recursos escassos disponíveis. Ainda que apresentando o problema de tratar-se de um campo já bastante explorado por técnicas convencionais, os sinais de voz possuem uma semelhança estrutural bastante grande com as assinaturas sonoras ditas industriais, uma vez que são determinados por diferenças físicas por vezes mínimas existentes na conformação anatômica de cada orador. Note-se ainda que, do ponto de vista do teste em si, a fala pode apresentar mesmo restrições mais severas à identificação, já que as diferenças encontradas entre os padrões são comumente mais sutis, isto é, os sinais de assinaturas de impacto, por exemplo, são mais característicos e constantes.

Assim, uma vez que a questão do objeto de estudo encontrava-se resolvida, buscou-se aproveitar tipos de modelos que possuíssem ligação com outros trabalhos desenvolvidos no programa. A partir do interesse em se estudar modelos de redes neuronais que apresentassem a capacidade de aprender padrões temporais e levando-se em conta que pesquisas anteriores analisaram o desempenho do modelo de

retropropagação convencional na previsão de séries no tempo, decidiu-se pelo emprego do modelo de retropropagação recorrente.

Esse modelo tornou-se atraente também por resolver alguns problemas de implementação que comumente surgem quando da análise de assinaturas sonoras. Inicialmente, a grande difusão e relativa familiaridade dos algoritmos de retropropagação convencional indicaram-nos para utilização no trabalho.

Contudo, ainda que sejam largamente usados na análise de séries temporais [1], o número de unidades na camada de entrada, nesses casos é baixo, geralmente na ordem de uma ou duas dezenas. As ondas sonoras, sejam das assinaturas de impacto, sejam de voz, apresentam uma quantidade de valores observados bem maior. No caso de sinais de impacto, emprega-se, no mínimo, uma digitalização em 256 pontos, enquanto que para os sinais de voz chega-se, para uma freqüência de aquisição ótima de 10 kHz, durante 1 s, a um total de 10.000 pontos. Se pudéssemos garantir que essas quantidades fossem fixas, ainda seria possível aplicar alguma estratégia de programação que contornasse o problema do volume de processamento e, o que é mais grave, do aumento da probabilidade de ocorrência de uma instabilidade no treinamento da rede. Ocorre porém que o comprimento do trem de pulsos que forma o padrão sonoro a ser analisado é variável. Isso dificulta o uso de técnicas conhecidas como retardo (*time-delay*), onde o sinal é apresentado à rede de uma só vez e que exigem um sincronismo rígido entre o tempos de apresentação no treinamento e no reconhecimento.

Na retropropagação recorrente, a rede não armazena exatamente as relações entre os vários termos de uma seqüência fixa. O que ocorre é que ela adquire informação a respeito do contexto da ocorrência de um determinado valor, em virtude da existência das unidades recorrentes, que influenciam o valor de saída relativo a um estímulo a partir dos  $n$  valores que lhe antecederam. Assim, a rede não necessita possuir uma camada de entrada de comprimento igual ao número de termos da série observada, ela absorve a estrutura básica geradora da seqüência a partir de observações de grupos de padrões que se sucedem no tempo, ou seja, surge uma característica associativa que identifica um determinado grupo de estímulos e os liga a uma grandeza qualquer desejada. Se a série no tempo, e os sinais de assinatura sonora podem ser encarados como tal, possui um alto grau de coerência, a rede torna-se uma previsora excelente e, mesmo quando a série apresentada é totalmente caótica, a capacidade de memorização da seqüência continua muito boa.

Deste modo, o tamanho da camada de entrada é dado pelo comprimento do grupo de padrões a ser identificado. Os valores a serem obtidos na camada de saída, lembrando-se que o aprendizado em questão é supervisionado, são associados aos próximos valores da série, o que confere a rede a capacidade de previsão. Contudo, deseja-se no presente trabalho conseguir também uma identificação por parte do modelo. O modo mais direto de converter-se uma característica na outra é treinar-se a rede para previsão a partir de uma assinatura padrão, onde cada amplitude de sinal  $n+1$ , onde  $n$  é o tamanho da camada de entrada, é o valor obtido em uma camada de saída de um só neurônio. Ao ser apresentada a série temporal, no caso o sinal de voz que se deseja identificar, a discrepância entre os valores esperado e encontrado para cada amplitude  $n+1$  fornece um grau de similaridade que é associado à identificação ou não da entrada.

Neste ponto deve-se ressaltar que a maior complexidade dos sinais de voz no tocante à identificação foi compensada através de uma mudança de abordagem. No caso das assinaturas industriais procurava-se uma classificação, ou seja, que a onda observada fosse enquadrada em um conjunto de observações realizadas *a priori* e assim conhecida sua fonte geradora. O correspondente a este processo no que diz respeito a voz é a identificação de fonemas: a rede seria capaz de reconhecer que palavra teria sido proferida, a partir de um vocabulário próprio. Ainda que esse processo seja viável, ele exige quatro etapas de processamento do sinal para que as características básicas da onda apareçam. Essas etapas são o cálculo da energia da onda, a determinação de sua frequência fundamental, a análise dos formantes da onda e o cálculo de coeficientes chamados LPC, *linear predictive code*, que fornecem o grau de autocorrelação do sinal de voz. Mesmo para um conjunto de fonemas pequeno, seria processamento demais para resolver problemas não existentes no objetivo inicial, as assinaturas de impacto. Assim, os sinais de voz em vez de classificados são apenas reconhecidos, ou seja, a partir de uma voz padrão, a rede é capaz de validar a identidade de um orador como sendo o dono daquela voz. Esse processo requer, para uma taxa de acerto superior a 90%, apenas os dois primeiros cálculos citados acima : a energia e a frequência fundamental, e presta-se a verificação do desempenho do modelo em associar uma determinada série a uma certa fonte.

Alguns pré-processamentos são ainda realizados no sinal de voz, de modo que ele possa ser analisado pela rede. O mais importante deles é o que liga as características importantes da fala citadas acima à série a ser analisada. Essas características poderiam, talvez, ser extraídas, ou percebidas, pela rede diretamente da série de amplitudes do sinal. Acontece, porém, que teríamos dessa forma o volume de processamento duplamente aumentado : primeiro pela própria necessidade de reconhecer duas

características desconexas na mesma entrada; além disso pelo treinamento extremamente longo decorrente do tamanho considerável da série original. O mais provável, porém, é que essa necessidade de um processamento muito mais exaustivo leve a uma identificação impraticável. Em outras palavras, a representação empregada não se justifica apenas pela redução do consumo de recursos computacionais : ela inicia a atividade de memorização e identificação, através de uma filtragem do que realmente deve ser levado em conta no sinal, é uma representação mais adequada e mais correta. Logo, a série original é representada por duas séries decorrentes, a série de energias e a de tons (frequências fundamentais) calculadas a cada 10 ms. A verificação do orador é efetuada utilizando-se ambas as séries decorrentes. Estas transformações da série original e outras serão descritas posteriormente.

A seguir será apresentado o conteúdo de cada capítulo, para uma visão geral da estrutura do trabalho.

## **I.2 - Estrutura do Trabalho**

Neste primeiro capítulo é fornecida uma visão geral do trabalho, incluindo a motivação, as restrições de ordem prática e teórica encontradas e até onde se estende a análise do modelo.

O Capítulo II trata da idéia da modelagem de redes neuronais artificiais em si, descrevendo os conceitos básicos da área, apresentando o paradigma da retropropagação ("*backpropagation*") e descrevendo suas aplicações. As alterações do modelo que são implementadas neste trabalho para a obtenção da retropropagação recorrente são apresentadas no Capítulo III, fechando-se assim a apresentação do modelo conexionista propriamente dito.

A partir de então são mostradas as aplicações do modelo e analisado o seu desempenho. No Capítulo IV é realizada uma comparação de sua capacidade previsora em relação a uma rede convencional. Foram utilizadas para esse fim três séries temporais de características diversas, já estudadas em trabalhos anteriores. Os detalhes da implementação, como topologia, neurônios e treinamento, são descritos. As séries empregadas e as saídas dos testes são fornecidos no Apêndice B.

Analogamente, o Capítulo V faz uma introdução aos fatores envolvidos no processamento de voz, para em seguida o Capítulo VI descrever a adaptação do modelo

de rede neuronal ao processo de verificação de um orador. Nesta etapa é detalhada toda a montagem do experimento, desde a aquisição do sinal, passando pelas conversões, processamentos preliminares, apresentação à rede e os resultados obtidos. Especificações do *hardware* e dos formatos dos arquivos de dados encontram-se nos Apêndices C e D.

Finalmente o Capítulo VII reúne as conclusões do trabalho, a partir das análises de desempenho encontradas e apresenta algumas sugestões de continuação da pesquisa, envolvendo não só as assinaturas sonoras originalmente desejadas mas também possíveis análises fonéticas baseadas no mesmo princípio. Todos os códigos-fonte empregados nas simulações são fornecidos no Apêndice A.

## CAPÍTULO II

### REDES NEURONAIS E RETROPROPAGAÇÃO

#### II.1 - Redes Neurais Artificiais

Uma rede neuronal é qualquer sistema em que unidades de processamento relativamente simples são interconectadas de uma forma o mais completa que for possível, de modo a adquirirem propriedades coletivas decorrentes de suas atividades. A idéia básica vem do modelo do sistema nervoso de vários animais, formados por células chamadas neurônios que interligam-se em uma rede. Estas são redes neuronais biológicas, modelos por vezes muito complexos e que ocorrem efetivamente em seres vivos. Nosso objeto de estudo são as redes neuronais artificiais, ou melhor, simulações delas em software. A diferença é que as redes artificiais são construídas segundo paradigmas orientados para a solução de algum problema específico, não apresentam plausibilidade biológica marcante, ou pelo menos não são verificadas na prática em organismos vivos e em geral apresentam unidades de processamento relativamente simples.

Apesar de a quantidade de paradigmas existente para as redes neuronais artificiais ser grande, a maioria deles envolve os seguintes componentes:

- Topologia da rede : as redes são formadas por camadas de neurônios que podem, conforme o caso, serem conectados entre ou dentro das camadas. Cada ligação entre duas unidades (dois neurônios) costuma ser chamada de sinapse, ainda em virtude da comparação com as células nervosas. As sinapses passam a ser expressas por um valor real, um peso, que modifica a entrada que o neurônio recebe.

- Tipo de Neurônio : cada tipo de unidade possui uma função de transferência, que determina a relação entre as entradas e a saída do neurônio. Em geral essas funções são não lineares.

- Treinamento : à rede deve ser apresentada uma série de estímulos que causarão saídas associadas e que modificarão seu estado global. Isso é o treinamento da rede. No caso de forçar-se uma saída desejada para um grupo de entradas, o treinamento diz-se supervisionado, pois é como se houvesse um professor corrigindo a resposta encontrada. Um treinamento não supervisionado é aquele em que a rede se auto-acomoda e atinge um ponto de estabilidade determinado para um grupo de estímulos e assim representa e armazena essa entrada.



Um dos paradigmas existentes, bastante difundido[2, 3], é o da retropropagação, ou, mais conhecido pelo termo original em inglês, *backpropagation*.

## II.2 - O Modelo *Backpropagation*

A rede *backpropagation* típica sempre possui uma camada de entrada, uma camada de saída e ao menos uma camada intermediária. Não há um limite teórico para o número de camadas intermediárias, mas normalmente são duas ou uma, sendo que trabalhos realizados [4] verificaram que um número máximo de quatro camadas (três intermediárias mais uma de saída) são suficientes e necessárias para resolver um problema de classificação de padrões complexos. Cada camada é completamente conectada à camada seguinte: na Figura II.1 é apresentado um grafo típico de uma rede *backpropagation*. As setas indicam o fluxo de informação durante o processamento da rede. Durante o aprendizado, a informação é também propagada para trás através das conexões e utilizada para atualizar os seus pesos. A rede pode ser tanto hetero-associativa quanto auto-associativa. No primeiro caso, os padrões associados são representados através de conjuntos de valores diferentes.

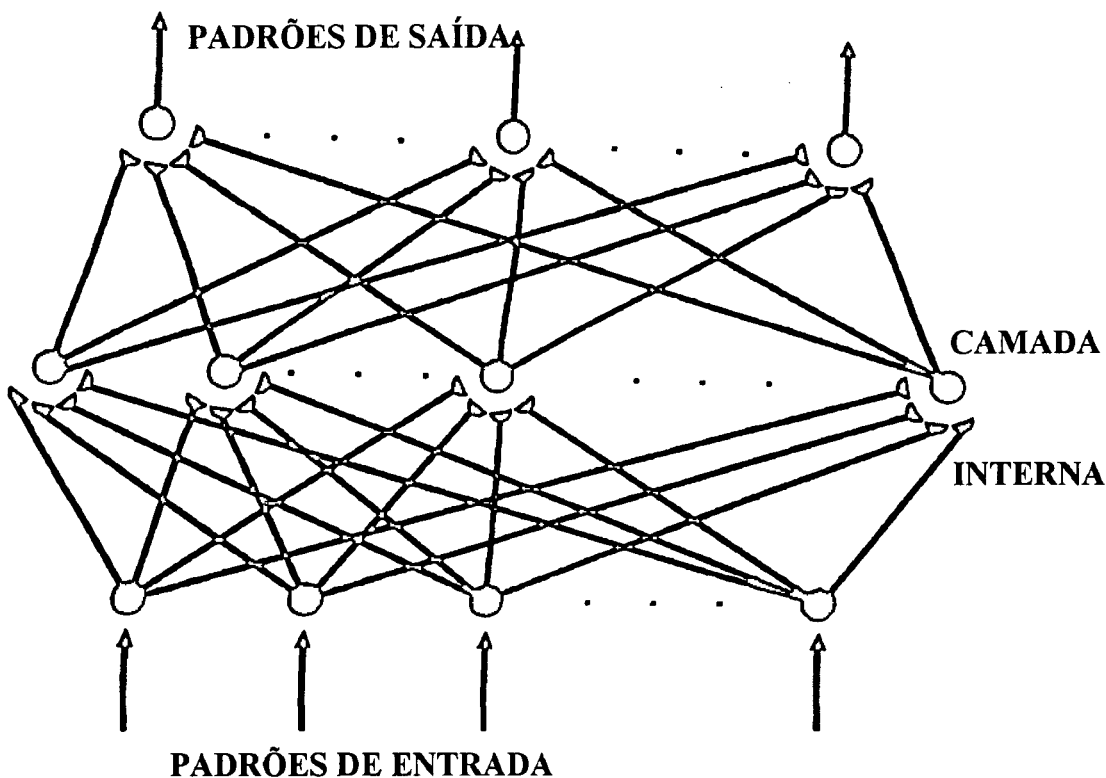


Figura II.1 - Rede de Retropropagação Típica

Uma rede auto-associativa representa internamente um padrão que pode ocorrer tanto na entrada quanto na saída, convergindo para o mesmo estado. Em princípio o modelo aqui descrito é hetero-associativo, ou seja, associará um vetor de valores na camada de saída a um outro apresentado na camada de entrada, utilizando vetores de comprimento diferente.

No que diz respeito ao treinamento, antes de mais nada, para evitar confusões entre uma camada e outra é necessária uma notação clara para descrever a regra de aprendizado. Utiliza-se um índice superscrito entre colchetes para indicar que camada da rede está sendo considerada. O resto da notação é mostrado abaixo :

$x_j^{[s]}$	estado de saída atual do neurônio <b>j</b> na camada <b>s</b>
$w_{ji}^{[s]}$	peso na conexão unindo o neurônio <b>i</b> na camada <b>(s-1)</b> com o neurônio <b>j</b> na camada <b>s</b>
$I_j^{[s]}$	Somatório dos pesos das entradas do neurônio <b>j</b> na camada <b>s</b>

O elemento de retropropagação desta forma transfere as suas entradas como mostrado abaixo :

$$\begin{aligned} x_j^{[s]} &= f\left(\sum_i (w_{ji}^{[s]} \cdot x_i^{[s-1]})\right) \\ &= f(I_j^{[s]}) \end{aligned} \quad (\text{II.1})$$

Na Figura II.2 é apresentado um esboço de um elemento de processamento típico da "backpropagation", onde **f** é tradicionalmente uma função sigmóide mas pode ser qualquer função diferenciável. A função sigmóide mais comum, apresentada na Figura II.3, é definida como

$$f(z) = (1,0 + e^{-z})^{-1} \quad (\text{II.2})$$

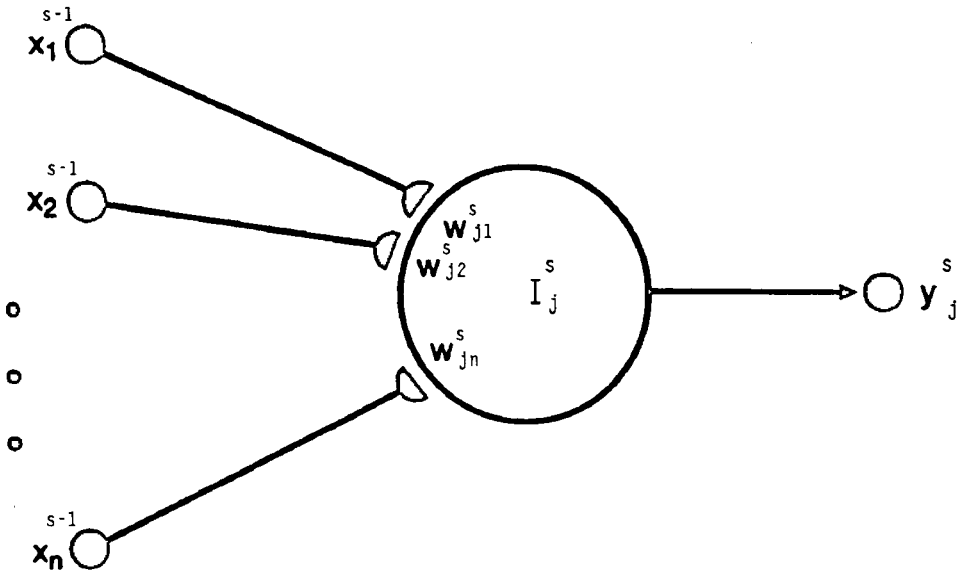


Figura II.2 - Típico Elemento de Processamento de Retropropagação

SAÍDA

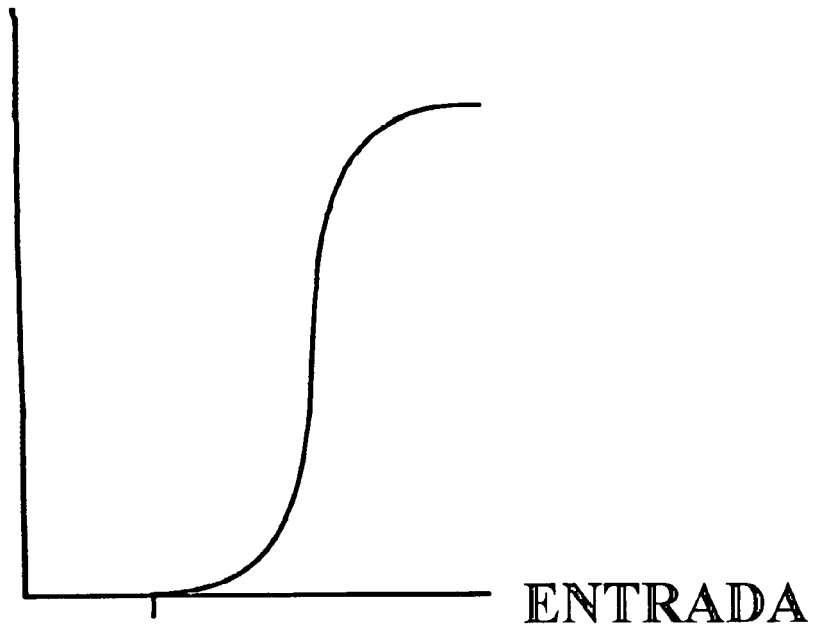


Figura II.3 - Função Sigmóide

Supõe-se que a rede tem uma função de erro global  $E$ , a ela associada que é uma função diferenciável de todos os pesos conectados na rede. A função de erro em si não é

importante para entender o mecanismo de retropropagação, o parâmetro crítico é que o erro é retornado para trás através das camadas e é definido por

$$e_j^{[s]} = -\partial E / \partial I_j^{[s]} \quad (\text{II.3})$$

Este vetor pode ser considerado como uma medida do erro local no processamento do elemento  $j$  no nível  $s$ .

Utilizando-se a regra da cadeia duas vezes em sequência chega-se a relação entre o erro local num determinado elemento de processamento num nível  $s$  e todos os erros locais no nível  $s+1$  :

$$e_j^{[s]} = f'(I_j^{[s]}) \cdot \sum_k (e_k^{[s+1]} \cdot w_{kj}^{[s+1]}) \quad (\text{II.4})$$

Note-se que em (II.4) existe uma camada acima da  $s$ , deste modo esta equação só pode ser usada para camadas que não são do tipo saída. Mais à frente será mostrada a equação para a camada de saída.

Se  $f$  é uma função sigmóide definida em (II.2), então sua derivada pode ser expressa como uma função simples dela própria como segue, sendo esta uma das razões dessa função ser tão comum:

$$f'(z) = f(z) \cdot (1,0 - f(z)) \quad (\text{II.5})$$

Assim, de (II.1), a equação (II.4) pode ser reescrita como

$$e_j^{[s]} = x_j^{[s]} \cdot (1,0 - x_j^{[s]}) \cdot \sum_k (e_k^{[s+1]} \cdot w_{kj}^{[s+1]}) \quad (\text{II.6})$$

sendo que a função de transferência é sigmóide. O termo do somatório em (II.6) que é usado para retropropagar os erros é análogo ao termo do somatório em (II.1) que é usado para propagar para frente a entrada através da rede. Portanto o mecanismo básico numa rede de retropropagação é de propagar para frente a entrada através das camadas para a camada de saída, calcular o erro na camada de saída, e então propagar o erro para trás através da rede da camada de saída para a camada de entrada usando (II.6), ou mais genericamente, (II.4). A multiplicação do erro pela derivada da função de transferência o distribui proporcionalmente.

O objetivo do processo de aprendizado é de minimizar o erro global  $E$  do sistema através da variação dos pesos. Será visto a seguir como fazê-lo baseando-se no conhecimento do erro local para cada elemento processado.

Dado o conjunto de pesos correntes de uma iteração,  $w_{ji}^{[s]}$ , precisa ser determinado como incrementá-lo ou decrementá-lo para reduzir o erro global. Isto pode ser efetuado utilizando-se a regra do gradiente descendente como segue:

$$\Delta w_{ji}^{[s]} = -lcoef \cdot (\partial E / \partial w_{ji}^{[s]}) \quad (II.7)$$

onde *lcoef* é o coeficiente de aprendizado, uma constante positiva que determina a velocidade do aprendizado. Em outras palavras, muda-se cada peso de acordo com o tamanho e a direção do gradiente negativo na superfície do erro.

A derivada parcial em (II.7) pode ser calculada diretamente dos valores dos erros locais apresentados anteriormente, porque, através da regra da cadeia e de (II.1):

$$\begin{aligned} \partial E / \partial w_{ji}^{[s]} &= (\partial E / \partial \mathcal{A}_j^{[s]}) \cdot (\partial \mathcal{A}_j^{[s]} / \partial w_{ji}^{[s]}) \\ &= -e_j^{[s]} \cdot x_i^{[s-1]} \end{aligned} \quad (II.8)$$

Combinando (II.7) e (II.8) obtém-se:

$$\Delta w_{ji}^{[s]} = lcoef \cdot e_j^{[s]} \cdot x_i^{[s-1]} \quad (II.9)$$

A dissertação anterior assumiu a existência de uma função de erro global sem realmente especificá-la. Esta função é necessária para definir os erros locais na camada de saída para que possam ser propagados para trás através da rede. Suponha que o vetor  $\underline{i}$  é apresentado na camada de entrada da rede e que a saída desejada  $\underline{d}$  é especificada pelo supervisor. Seja  $\underline{o}$  a denominação da saída produzida pela rede com seus conjuntos de pesos. Então a medida do erro para alcançar a resposta desejada é dada por:

$$E = 0,5 \cdot \sum_k ((d_k - o_k)^2) \quad (II.10)$$

onde o subscrito  $k$  indexa os componentes de  $\underline{d}$  e  $\underline{o}$ . Aqui, o erro local puro é dado por  $d_k - o_k$ . De (II.3), o "erro local" distribuído para cada elemento processado da camada de saída é dado por

$$\begin{aligned} e_k^{(o)} &= -\partial E / \partial \mathcal{A}_k^{(o)} \\ &= -\partial E / \partial o_k \cdot \partial o_k / \partial \mathcal{A}_k \\ &= (d_k - o_k) \cdot f'(I_k) \end{aligned} \quad (II.11)$$

O valor de  $E$ , conforme apresentado em (II.10), define o erro global da rede para um par ordenado  $(\underline{i}, \underline{d})$  específico. A função de erro global total pode ser definida como a soma das funções de erro específicas para cada padrão. Assim a cada tempo que um par  $(\underline{i}, \underline{d})$  específico é mostrado, o algoritmo de retropropagação modifica os pesos para

reduzir o determinado componente da função de erro global. A regra de aprendizado descrita nesta seção é conhecida como regra delta generalizada.

Um dos problemas do algoritmo do gradiente descendente é o ajuste da taxa de aprendizado apropriada. Mudar os pesos através de uma função linear da derivada parcial como definida em (II.7) parte do pressuposto que a superfície do erro é localmente linear, onde "localmente" é definido pelo coeficiente de aprendizado. Nos pontos de alta curvatura esta hipótese não se mantém e um comportamento divergente pode ocorrer nesta região. É portanto necessário que se mantenha o coeficiente de aprendizado baixo para evitar tal comportamento.

Por outro lado um pequeno coeficiente de aprendizado pode resultar num aprendizado muito lento. O conceito de "termo de momento" foi introduzido para resolver esta dicotomia. A equação de variação de peso (II.9) é modificada para que a porção da variação anterior seja alimentada através da variação corrente:

$$\Delta w_{ji}^{[s]} = Icoef \cdot e_j^{[s]} \cdot x_i^{[s-1]} + momentum \cdot \Delta w_{ji}^{[s]} \quad (II.12)$$

Isto funciona como um "filtro passa baixa" nos termos de variação de peso, uma vez que ligações que mais contribuem para a convergência são reforçadas enquanto que um comportamento oscilatório cancela-se. Isto permite um coeficiente de aprendizado baixo mas um aprendizado rápido.

Uma técnica também empregada que proporciona um efeito de aceleração na taxa de convergência é a de apenas atualizar os pesos *depois* que pares ( $i$ ,  $d$ ) são apresentados à rede, em lugar de a cada apresentação. Isto é conhecido como retropropagação cumulativa, ou em *batch*, porque as variações são acumuladas até que o conjunto de pares seja apresentado. O número de pares entrada/saída apresentados durante a acumulação é referido como um "período". Isto pode corresponder ou a um conjunto completo de pares de treinamento ou a um subconjunto.

Uma outra forma de olhar esta abordagem é em termos de uma função de erros composta como oposto ao padrão individual dependente das funções de erro. A função de erro composta é definida como a soma das funções de erro individual, consequentemente sua derivada parcial com respeito a qualquer peso é a soma das derivadas parciais correspondentes às funções de erro individuais. Uma vez que as variações são proporcionais às derivadas parciais, acumulando as primeiras sobre o conjunto completo de padrões é simplesmente a mesma coisa que fazer retropropagação usando a função de erro composta.

Se o período não for tão grande, isto pode levar a uma convergência mais rápida, uma vez que as atualizações individuais somente reduzem a função de erro para um par de padrão particular, mas pode aumentar outra função de erro composta, onde a atualização global irá sempre trabalhar para reduzir a função de erro global. Por outro lado, com a retropropagação cumulativa, muito mais cálculos precisam ser feitos para alcançar uma atualização simples e o benefício de utilizar uma função de erro global pode ser perdido se o período for grande.

Associada a essas otimizações do processo de treinamento de uma rede *backpropagation*, são muito empregadas ainda duas estratégias. Primeiramente a iniciação randômica do valor das sinapses. Esse tipo de iniciação não determinística dos estados das conexões antes do início do treinamento, mantendo-se os valores randômicos dentro de uma faixa segura, protege o aprendizado do risco de permanecer preso em um ponto de mínimo local, ou seja, um vetor de saída que, mesmo sendo mínimo na seqüência de treinamento escolhida, não é o mais preciso que pode ser atingido. Outra forma de se diminuir esse risco é alterar-se a ordem em que os padrões são apresentados. Como será visto posteriormente, essa última hipótese não é adequada ao tipo de abordagem desse trabalho.

Finalmente, tentando ainda melhorar o desempenho de algumas das técnicas aqui mencionadas, costuma-se introduzir o algoritmo conhecido por EDBD (Extended Delta-Bar-Delta), apresentado por Jacobs [5] e adaptado por Minai e Willems [6].

A idéia do EDBD é estabelecer tanto coeficientes de aprendizado como termos de momento variáveis, de modo a que eles possam adaptar-se à curvatura da superfície da função de erro. Essa variação dá-se de forma heurística, levando-se em conta o sentido das últimas variações de cada sinapse. Assim, cada sinapse possui seu próprio coeficiente de aprendizado e seu próprio termo de momento, que variam segundo a equação (II.13), onde o coeficiente de aprendizado foi denotado por  $\alpha$  e o momento por  $\mu$ :

$$\begin{aligned}\alpha(t+1) &= \alpha(t) + \Delta\alpha(t) \\ \mu(t+1) &= \mu(t) + \Delta\mu(t)\end{aligned}\tag{II.13}$$

onde

$$\Delta\alpha(t) = \begin{cases} K_\alpha e^{-\gamma_\alpha |\bar{\delta}(t)|} & \text{se } (\bar{\delta}(t-1) \cdot \delta(t)) > 0 \\ -\varphi_\alpha \alpha(t) & \text{se } (\bar{\delta}(t-1) \cdot \delta(t)) < 0 \\ 0 & \text{nos demais casos} \end{cases} \quad (\text{II.14})$$

$$\Delta\mu(t) = \begin{cases} K_\mu e^{-\gamma_\mu |\bar{\delta}(t)|} & \text{se } (\bar{\delta}(t-1) \cdot \delta(t)) > 0 \\ -\varphi_\mu \mu(t) & \text{se } (\bar{\delta}(t-1) \cdot \delta(t)) < 0 \\ 0 & \text{nos demais casos} \end{cases} \quad (\text{II.15})$$

E a variação da sinapse é dada por:

$$\Delta\omega(t+1) = \alpha(t) \cdot \delta(t) + \mu(t) \cdot \Delta\omega(t) \quad (\text{II.16})$$

onde:

$$\delta(t) = \frac{\partial E(t)}{\partial \omega(t)} \quad \text{erro local proporcional à sinapse}$$

$$\bar{\delta} = (1 - \theta) \cdot \delta(t) + \theta \cdot \delta(t-1) \quad \text{média ponderada entre o erro corrente e o anterior}$$

Daí o nome de delta-bar-delta.

$\theta$  - peso dos erros

$t$  - passo do treinamento

$K_\alpha$  e  $K_\mu$  fator de escala da taxa de aprendizado e do termo de momento, respectivamente.

$\gamma_\alpha$  e  $\gamma_\mu$  fator do expoente de decaimento da taxa de aprendizado e do termo de momento, respectivamente.

$\varphi_\alpha$  e  $\varphi_\mu$  fator de decremento da taxa de aprendizado e do termo de momento, respectivamente.



Note-se que emprega-se um crescimento linear das taxas em casos de baixa curvatura, ou seja, as duas últimas variações tiveram o mesmo sinal, enquanto o decaimento é exponencial, e decrementado proporcionalmente a seu próprio valor, quando a curvatura aumentar. Isso reduz a possibilidade de oscilações violentas, mas mesmo assim são estabelecidos máximos para os termos:

$$\alpha(t) \leq \alpha_{\max} \quad \text{e} \quad \mu(t) \leq \mu_{\max}$$

O resultado é um período de treinamento razoavelmente mais curto na maioria dos casos, em virtude da eliminação dos saltos desnecessários na descida do gradiente.

## CAPÍTULO III

### O MODELO DE REDE UTILIZADO : RETROPROPAGAÇÃO NO TEMPO

#### III.1 - Inovações do Modelo e Características Principais

O modelo de rede neuronal apresentado até agora, apesar de bastante disseminado e de ser aplicável na resolução de diversos problemas ligados ao reconhecimento de padrões, é essencialmente estático. Com isso quer-se dizer que o padrão analisado é considerado isolado de qualquer outro relacionamento no tempo. No estudo de séries, o tempo não é uma característica que se possa desprezar: torna-se interessante que o padrão a ser reconhecido possua uma conotação temporal, de modo a ser melhor representado. De certa forma isto é tentado em implementações de *backpropagation* que utilizam retardo (*time delay*) [7], mas tal abordagem apresenta inconvenientes como um tamanho de série pré-estabelecido e baixa tolerância a ruídos.

Surgiu a idéia, então, de dotar-se uma rede de retropropagação de uma memória, de um contexto, isto é, possibilitar que a rede levasse em consideração valores encontrados anteriormente, em suma, que o modelo recorresse no tempo [8, 9]. Assim, o paradigma empregado é o da **retropropagação recorrente no tempo**. Este tipo de rede pode basear sua resposta para um evento em um contexto de tempo onde o problema ocorre [10, 11].

Uma rede recorrente é qualquer rede em que a atividade dos neurônios flui tanto da camada de entrada para a de saída (flui para frente) quanto da camada de saída para a de entrada (flui para trás) quando está gerando a resposta da rede para um estímulo de entrada particular. Numa rede de retropropagação usual, a atividade da rede (a atividade de espalhamento, *spreading*) flui somente em uma direção: da camada de entrada para a de saída. Enquanto os erros se propagam para trás, da camada de saída para a de entrada, durante o treinamento para alterar os pesos, a atividade da rede flui somente na outra direção. A rede de retropropagação, apesar do nome, é uma rede de propagação para frente, os erros é que voltam.

Ao contrário, se considerarmos uma rede atratora como a de Hopfield [12] e a BAM de Kosko [13], veremos que estas redes operam permitindo que a atividade de um estímulo de entrada flua ciclicamente dentro da rede até encontrar um ponto de atividade

estável. Na rede de Hopfield de uma camada, ocorre a competição para trás o que geralmente implica na estabilização da rede em um dado padrão de saída. Na rede de duas camadas BAM, a atividade flui de uma camada para a próxima e de volta até que as camadas atinjam um estado ressonante mutuamente reforçado estável. Ambas as redes são simples exemplos de redes recorrentes; ambas são redes de retropropagação, mas não no tempo.

A idéia da retropropagação no tempo é unir a noção de recorrência e operação de retropropagação ao já confiável algoritmo de treinamento da regra delta generalizado. O resultado é uma rede que tem algumas capacidades interessantes.

### III.2 - Operação Básica

Existem alguns modos de se ilustrar a arquitetura de retropropagação no tempo. Uma forma mais fácil para visualizar é fazer uso do conceito de neurônios especiais, "virtuais" ou "recorrentes" dentro da rede. A Figura III.1 ilustra esta rede.

Caso se ignorem os neurônios sombreados na Figura III.1, a arquitetura básica da rede é a mesma de qualquer rede *backpropagation* comum. Neste esquema, a rede possui uma camada de entrada que é constituída por três neurônios, possui dois neurônios na camada intermediária, e um neurônio na camada de saída. O esquema de conexão é o típico esquema de conexão completa comumente utilizado nas redes de retropropagação.

Aonde a rede de retropropagação no tempo diferencia-se é nos neurônios extras nas camadas de entrada e intermediária. No esquema acima, a camada de entrada tem dois destes neurônio virtuais. A do meio tem um neurônio virtual correspondente a cada neurônio da camada de saída. Cada um destes neurônios recorrentes recebe um único sinal de entrada, que é a saída do seu neurônio regular correspondente na camada seguinte. Em outras palavras, o neurônio recorrente na camada intermediária recebe a saída do neurônio da camada de saída. Analogamente, cada um dos dois neurônios recorrentes da camada de entrada recebe a saída de um dos dois neurônios da camada intermediária.

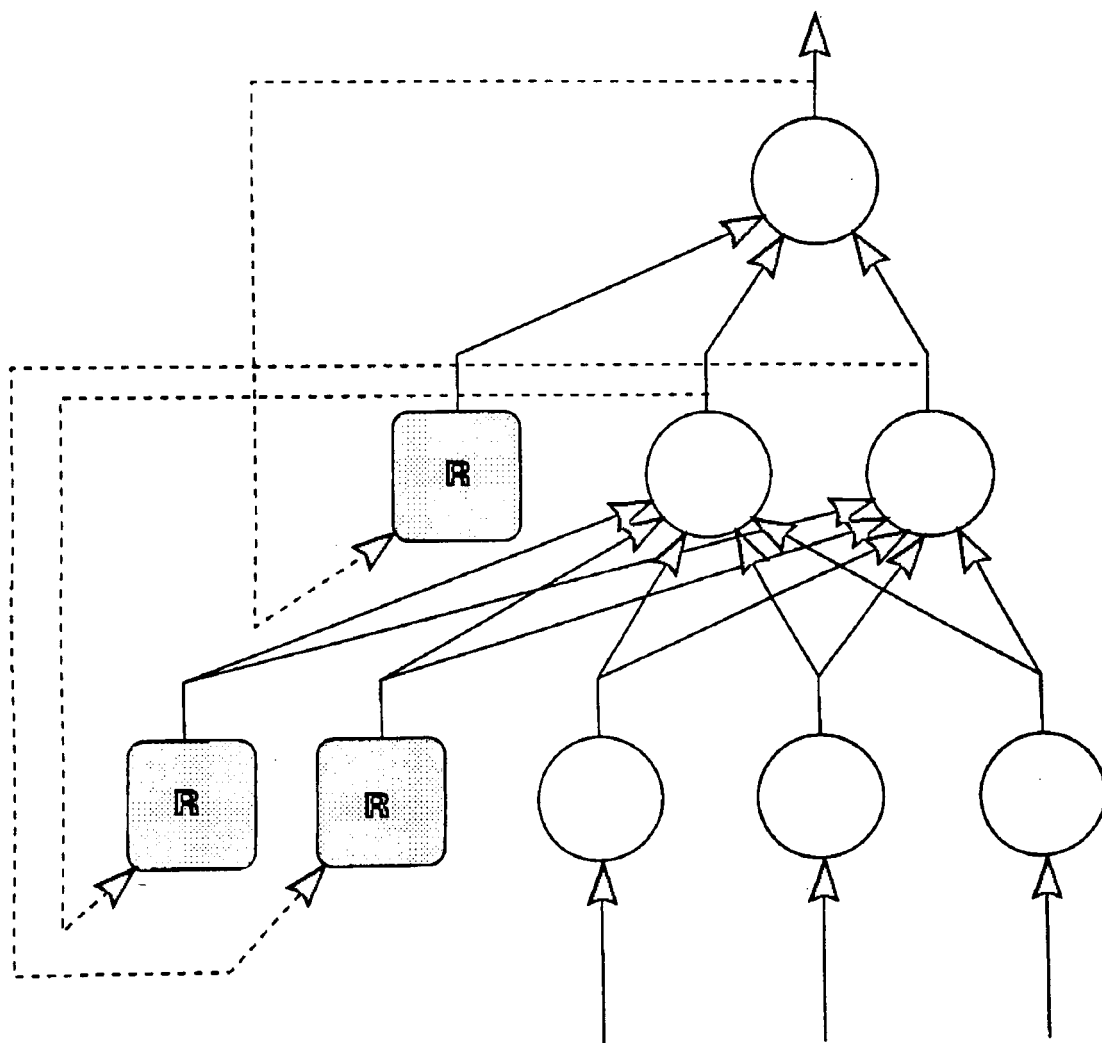


Figura III.1 - Rede de Retropropagação Recorrente no Tempo

A idéia de efetuar esta alteração na rede de retropropagação é que a entrada do neurônio recorrente reflita sua saída do neurônio normal correspondente no tempo anterior. Isto porque esta rede tenta aprender padrões dentro do contexto temporal. Os neurônios recorrentes são a chave para permitir este contexto.

O primeiro efeito deste arranjo é reconhecer que os padrões de treinamento apresentados à rede devem ser seqüências de padrões, e não pontos específicos. Em outras palavras, esta rede pode aprender a produzir a saída **C** quando é fornecido um determinado estímulo que chega logo depois do estímulo que produz a saída **B**. Se o mesmo estímulo gerador de **C** chega depois de algum outro padrão, a saída pode ser completamente diferente. A rede aprende a produzir **C** somente quando o contexto temporal produziu recentemente **B**.

A Figura III.2 ilustra o fluxo geral através da rede. A rede começa com um arbitrário mas geralmente pequeno valor "reset" de atividade em todos os neurônios na camada intermediária e da saída. O valor de "reset" é a atividade dos neurônios no tempo  $t=0$ . Normalmente este valor é o mesmo para todos os neurônios na rede; foi suposto 0,25 neste caso.

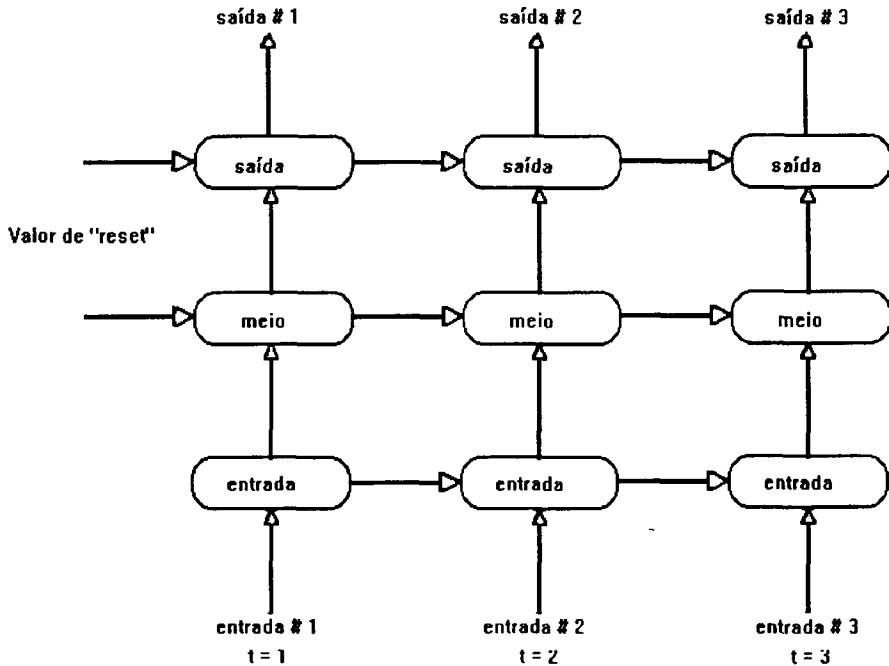


Figura III.2 - Fluxo de Ativação da Rede

Uma vez que esse valor representa a atividade dos neurônios no instante  $t=0$ , também significa a atividade no neurônio recorrente no tempo  $t=1$ . Referindo-se à Tabela III.1, este também é o tempo no relógio em que representa-se o primeiro padrão de entrada, ou seja, 0 0 1 em código binário, que representará o estímulo para a primeira saída da rede, ou seja, 0,25. A atividade no neurônio de entrada é transmitida através das conexões dos pesos para os dois neurônios da camada intermediária, exatamente como ocorre na rede de retropropagação comum. Acrescente-se, portanto, a atividade dos neurônios da camada intermediária no tempo  $t=0$ , o valor reset de 0,25, é também transmitido através das conexões adequadas para a camada intermediária. Estas duas fontes de entrada combinadas na forma normal se somar pesos produzem o valor de entrada da rede dado na Equação II.1, aqui repetida:

$$I = \sum_{i=1}^n w_{ij} \cdot x_i \quad (\text{III.1})$$

onde  $w_{ij}$  representa o peso de cada conexão do neurônio de entrada  $i$  para o  $j$  da camada intermediária e  $x_i$  representa o sinal de atividade do neurônio  $i$  da camada de entrada. Deve-se tomar cuidado para o fato de as atividades dos neurônios recorrentes serem da camada intermediária no tempo anterior (neste caso no tempo  $t=0$ ), enquanto as

atividades do neurônio regular da camada de entrada são as atividades normais do tempo atual ( $t=1$ ).

Fornecida esta entrada para a rede, as duas camadas intermediárias produzem suas próprias saídas usando uma função sigmóide. A mais comum é (II.2):

$$f(I) = \frac{1}{(1,0 + e^{-I})} \quad (\text{III.2})$$

mas qualquer função parecida do tipo sigmóide, isto é, apresentando uma curva em S, irá servir. Este valor representa a saída de cada uma das atividades dos neurônios da camada intermediária.

Os neurônios da camada intermediária fornecem as atividades para o único neurônio da camada de saída através das suas conexões apropriadas. Além disto, este neurônio também recebe um sinal do neurônio recorrente da camada intermediária. A saída deste neurônio recorrente representa a saída do neurônio da camada de saída no instante de tempo anterior ( $t=0$ ). Como na camada intermediária, o neurônio da camada de saída recebe uma combinação de estímulos, alguns da atividade atual e outro da sua própria atividade no tempo anterior. As duas fontes de entrada combinadas levam a um valor de entrada da rede, que de volta produz a atividade de saída do neurônio. Esta atividade é também o padrão de saída da rede como um todo.

A operação básica da rede não é, como pode ser observado, tão diferente da da rede de retropropagação comum. A diferença principal, no entanto, é que a rede recorrente tem mais o que lembrar, deve constantemente saber se uma determinada atividade representa uma atividade atual ou alguma outra antiga.

Quando o segundo estímulo chega no lado de entrada da rede, a camada intermediária recebe não somente o padrão de entrada atual, mas também sua própria atividade do instante anterior. Esta combinação gera a atividade da camada intermediária no tempo  $t=2$ . Analogamente, a camada intermediária vê a atividade sua atual,  $t=2$ , mas vê também sua própria saída no tempo  $t=1$  do nó recorrente. A atividade recorrente proporciona o contexto temporal em que a rede aprendeu a produzir a saída correta.

### III.3 - Cálculo de Erros

Considere-se inicialmente a saída da rede no tempo  $t=2$ . A atividade da camada de saída é determinada pela atividade da camada intermediária em  $t=2$  e sua própria

atividade em  $t=1$ . Para se calcular o erro no tempo  $t=1$  é necessário saber o erro do tempo  $t=2$ , que ainda não existe. Portanto, o cálculo do erro é feito de trás para frente, do instante de tempo final,  $N$ , até o instante  $t=1$ . Ou seja, os erros numa rede de retropropagação recorrente são retropropagados, não somente espacialmente, da camada de saída para a intermediária, mas também no tempo, do último instante da seqüência para o instante anterior e assim sucessivamente até o instante inicial.

Começa-se com o último instante da seqüência. A camada de saída neste instante tem o erro específico igual ao padrão desejado menos a saída atual. Assumindo-se que são  $N$  padrões na seqüência, o erro é dado por:

$$E(N)_j = (y(N)_j^{\text{desejado}} - y(N)_j^{\text{atual}}) \frac{df(I(N))}{dI} \quad (\text{III.3})$$

Ressalta-se que uma das razões para se utilizar a função sigmóide é que a sua derivada é fácil de ser calculada:

$$\frac{df(I(N))}{dI} = f(I(N)) \cdot (1 - f(I(N))) \quad (\text{III.4})$$

onde  $f(I(N))$  é a saída do neurônio no tempo  $t=N$ , que é conhecida. Este é o erro comum que é utilizado em qualquer rede de retropropagação. Seja o símbolo a seguir utilizado para representar o resultado "desejado menos o atual":

$$\in(N)_j = (y(N)_j^{\text{desejado}} - y(N)_j^{\text{atual}}) \quad (\text{III.5})$$

substituindo na equação (III.3), tem-se, para o erro da camada de saída no tempo  $t=N$ ; onde alterou-se a notação entre colchetes para maior clareza:

$$E(N)_j^{\text{saida}} = \in(N)_j^{\text{saida}} \frac{df(I(N))}{dI} \quad (\text{III.6})$$

Para instantes de tempo anteriores, no entanto, não se pode utilizar somente o erro normal da camada de saída, mas também o erro retropropagado do instante seguinte. Isto resulta na equação de erro mais complexa (III.7). Observa-se que o novo termo do somatório do peso é exatamente o mesmo que qualquer outro termo de erro retropropagado numa rede de retropropagação comum.

$$E(t)_j^{\text{sai}} = (\in(t)_j^{\text{sai}} \frac{df(I(t))}{dI}) + \sum_{i=1}^{T_{\text{amSai}}} w_{ij}^{\text{sai-sai}} \cdot E(t+1)_i^{\text{sai}} \frac{df(I(t)_j)}{dI} \quad (\text{III.7})$$

Para calcular os erros na camada intermediária para o tempo final efetua-se a retropropagação normalmente, tomando-se o somatório dos pesos e fatorando a derivada da função sigmóide. Para todos os tempos anteriores, o erro desta camada é

calculado conforme apresentado na Equação III.8. Como com a camada final, a retropropagação temporal é realizada ao longo das conexões recorrentes.

$$E(t)_j^{meio} = \left( \sum_{i=1}^{TamSai} w_{ji}^{meio-sai} \cdot E(t)_i^{sai} + \sum_{k=1}^{TamMeio} w_{jk}^{meio-meio} \cdot E(t+1)_k^{sai} \right) \frac{df(I(t)_j^{meio})}{dI} \quad (III.8)$$

Finalmente, o cálculo das alterações dos pesos para a rede de retropropagação recorrente é realizado do mesmo modo que numa rede comum, através da regra delta generalizada. A única diferença é que o neurônio recorrente usa a atividade do instante de tempo anterior, enquanto os neurônios usuais usam a atividade do instante atual, como mostrado nas equações a seguir:

$$\Delta w_{ij}^{ent-meio/meio-sai} = lcoef \cdot E(t)_i \cdot y(t)_j \quad (III.9)$$

e

$$\Delta w_{ij}^{meio-meio/sai-sai} = lcoef \cdot E(t)_i \cdot y(t-1)_j \quad (III.10)$$

### III.4 - Treinamento

O algoritmo de treinamento precisa saber a qual instante de tempo está sendo referido. A chave complicadora é que a rede está aprendendo a responder seqüências de padrões, e não a padrões individuais. Isto significa que o conjunto de treinamento consiste num conjunto de seqüências. Suponha-se que, por exemplo, cada seqüência no conjunto de treinamento é constituído de uma seqüência de três padrões. Pode-se desejar treinar a rede exemplo para produzir as seguintes respostas para as duas seqüências:

Tabela III.1 - Exemplo de Conjunto de Seqüências para Treinamento

SEQÜÊNCI DE ENTRADA			SEQÜÊNCIA DE SAÍDA		
A					
t1	t2	t3	t1	t2	t3
1 0 0	0 1 0	0 0 1	0,25	0,0	1,0
0 0 1	0 1 0	1 0 0	0,5	1,0	0,75

Deve ficar claro, somente observando estas seqüências, que uma rede de retropropagação comum não poderá aprender a resposta correta. Uma vez que uma rede *backpropagation* comum deve sempre responder a uma entrada de 1 0 0 da mesma



forma, não poderá adivinhar que algumas vezes deveria responder com 0,25 e outras com 0,75, dependendo do contexto da entrada anterior.

Obviamente, num exemplo simples como este, a rede de retropropagação pode ser construída para aprender este problema. Se a camada de entrada é modificada para ter nove neurônios e a camada de saída para ter três, a sequência de entrada completa pode ser apresentada de uma vez, e a rede pode aprender a produzir a saída desejada completa. Em casos de padrões de entrada mais complexos, no entanto, o tamanho da camada de entrada fica muito grande para implementações práticas. Nestes casos a rede de retropropagação recorrente é mais apropriada.

A chave para o treinamento da rede recorrente é que o treinamento é feito em padrões de seqüências e num modo de treinamento em lote (*batch*). O lote de treinamento significa que acumula-se a mudança dos pesos através de todos o padrões de entrada antes de usar qualquer um deles para alterar os pesos da rede.

O regime de treinamento consiste num processo de três estágios para cada passo de treinamento. Primeiro, é realizado o processo para a frente para toda a seqüência padrão do instante  $I$  até o  $N$ , onde  $N$  representa o número de padrões na seqüência de padrões (três no exemplo). Em seguida, computam-se os erros para as camadas de saída e intermediária, começando com os erros do último instante na seqüência e trabalhando os erros voltando para o instante de tempo inicial. Finalmente, utilizam-se esses erros para computar e acumular mudanças nos pesos para cada conexão em cada camada para cada instante da seqüência, começando com o tempo inicial e indo até o tempo  $N$ . Somente quando todos estes passos forem completados devem-se adicionar as alterações nos pesos acumuladas às conexões na rede.

O primeiro passo do processo é computar o fluxo de ativação para frente. Numa forma geral, a entrada da rede para cada neurônio nas camadas intermediária e final é computada como mostra nas Equações III.11 e III.12. Nestas equações, o superscrito *ent-meio*, *meio-meio*, *meio-sai* e *sai-sai* distingue os pesos de cada conjunto utilizado. Os pesos que vão da camada do meio-para-meio ou do meio-para-saída são os pesos normais entre os neurônios comuns da rede. Os termos *TamEnt*, *TamMeio* e *TamSai* referem-se ao número de neurônios comuns de cada camada - os neurônios recorrentes não são levados em conta aqui.

$$I(t)_j^{meio} = \sum_{i=1}^{TamEnt} w_{ij}^{ent-meio} \cdot y(t)_i^{ent} + \sum_{k=1}^{TamMeio} w_{kj}^{meio-meio} \cdot y(t-1)_k^{meio} \quad (III.11)$$

$$I(t)_j^{sai} = \sum_{i=1}^{TamMeio} w_{ij}^{meio-sai} \cdot y(t)_i^{meio} + \sum_{k=1}^{TamSai} w_{kj}^{sai-sai} \cdot y(t-1)_k^{sai} \quad (III.12)$$

Estas entradas da rede são então passadas através da função sigmóide  $f(I)$  para gerar as saídas de cada neurônio  $y(t) = f(I)$ . Quando este passo é completado para cada padrão na seqüência de treinamento, pode-se começar a computar os erros. A rede em geral é considerada treinada quando o conjunto dos erros, ou um erro global, são minimizados.

A atividade de treinamento da rede recorrente no tempo é sem dúvida a principal dificuldade que este modelo apresenta. Em aplicações em que o erro global de treinamento necessite ser muito pequeno e a seqüência a ser aprendida possuir muitos termos, as sessões de treinamento geralmente se estendem por vários dias, utilizando-se o equipamento computacional normalmente encontrado no meio acadêmico. O modelo pode ser alterado para aprender uma só grande seqüência, em vez de várias pequenas. Esta abordagem foi utilizada nas aplicações deste trabalho, mesmo nas séries temporais que não provinham de sinais de voz. Foi construída uma rede neuronal para cada série. É possível alterar-se o esquema do exemplo deste capítulo para levar em consideração um número maior de tempos anteriores, com um aumento da abrangência da recursão mas implicando em um aumento drástico de tempo de treinamento. A seguir será visto como o modelo pode ser aplicado para a previsão de séries.

## CAPÍTULO IV

### APLICAÇÃO NA PREVISÃO DE SÉRIES TEMPORAIS

#### IV.1 - Topologia Utilizada

Conforme visto no Capítulo II, as redes neuronais que seguem o modelo *backpropagation* apresentam uma ou duas camadas intermediárias. A implementação empregada neste trabalho visava a uma capacidade de generalização mais acentuada, além de manter uma preocupação com o volume de processamento envolvido. Assim sendo, as camadas foram construídas da seguinte forma :

Camada de saída : um neurônio, cujo estado, representa o termo  $n+1$  da série sendo analisada, onde  $n$  é descrito em seguida.

Camada de entrada : três neurônios, determinando assim o comprimento dos padrões a serem detectados na série observada.

Camada intermediária : apenas uma, com dois neurônios.

Camadas recorrentes : uma intermediária e uma de saída.

Para as sessões de treinamento da rede foram utilizadas as primeiras  $n = t \times 3$  ocorrências da série. Esse número de ocorrências foi determinado pelo comprimento da primeira camada, três neurônios. Uma camada de entrada muito longa memoriza padrões mais complexos, mas torna o aprendizado mais penoso e diminui o poder de generalização da rede. Decidiu-se manter a primeira camada nesse valor reduzido, levando-se em conta o comportamento bem definido das séries estudadas. Já o tamanho das seqüências observadas,  $t$ , variou entre 8 e 15, de modo a observar-se o ponto de melhor resposta, i.e., com que número de observações de ocorrências passadas a rede efetua sua melhor precisão, das  $m$  ocorrências à frente do trecho utilizado para treinamento, número esse que variou entre 1 e 3. Ambos os parâmetros,  $n$  e  $m$ , tiveram seu trecho de variação escolhido baseando-se no tamanho das seqüência utilizadas no caso de *backpropagation* convencional que foi usado para comparação [14].

De cada vez foi considerado que se treinava uma só grande seqüência, em vez de treinarem-se várias seqüências curtas. Tal procedimento foi escolhido já tendo-se em mente a aplicação em assinaturas sonoras, onde essa abordagem é mais adequada, pois encara a série como um fenômeno apenas e não uma composição de pequenas séries.

Empregou-se também um termo de momento de 0,1 em conjunto com um coeficiente de aprendizado de 0,5 e um valor de reset, i.e., um valor inicial para o tempo  $t_0$  de 0,25. Já a iniciação das sinapses foi realizada através de valores randômicos, variando entre 0 e  $1/\sqrt{k}$ , onde  $k$  é o total de conexões do neurônio da camada mais alta na conexão. Assim as sinapses foram iniciadas com no mínimo  $\sqrt{3}$  nas ligações entre a entrada e a camada intermediária e até  $\sqrt{2}$  nas conexões com a camada de saída, de modo a evitarem-se mínimos locais causados pela saturação dos neurônios. Na maior parte dos casos, foi suficiente um total de 150.000 iterações, de modo que o erro de cada passo do aprendizado foi suficientemente pequeno para a rede ser considerada treinada. Pode-se verificar nas tabelas de resultados que o primeiro valor da precisão possui um erro bastante reduzido.

Também a fim de evitar a saturação dos neurônios, os valores a serem apresentados à camada de entrada passaram por uma normalização, de modo a situarem-se proporcionalmente dentro de uma faixa segura. Os valores fornecidos pela camada de saída foram então reconvertidos ao domínio original, para fins de inclusão no relatório. Essa normalização deu-se segundo a Equação (IV.1), mostrada a seguir.

$$S_i = f(Z_i) = (Z_i \cdot Escala) + Deslocamento \quad (IV.1)$$

onde

$$Escala = 0,8 / (ValorMáximo - ValorMínimo)$$

$$Deslocamento = 0,1 - ValorMínimo * Escala$$

As tabelas contendo a saída de cada teste realizado encontram-se no Apêndice B.

## IV.2 - Características das Séries Empregadas

Três séries temporais foram submetidas à implementação do modelo de retropropagação recorrente no tempo descrito na seção anterior. A principal razão da escolha dessas séries é o fato de as mesmas já terem sido objeto de um estudo comparativo entre um modelo *backpropagation* convencional e os modelos analíticos clássicos [14], naquele caso, Box-Jenkins. Os valores foram obtidos de livros e monografias [15] que descrevem essa metodologia, de modo que apresentam perfis que contém características adequadas à análise comparativa. O referido estudo que empregou um modelo BP convencional descreveu um procedimento onde, a partir de análises numéricas tradicionais, o número de neurônios da primeira camada para um desempenho ótimo era determinado. Esse número situou-se sempre na faixa de quantidade de padrões

adotada no presente trabalho. Contudo, a não coincidência do mínimo erro para a mesma quantidade de padrões talvez possa ser explicada pelo fato de a rede recorrente trabalhar com um padrão mais complexo, no caso formado por três componentes da seqüência, possibilitando uma sensibilidade maior da estrutura da série.

As séries em questão representam : o consumo mensal de energia do estado de Ohio, USA, de 1955 a 1970 em MWh; o total de unidades de um determinado equipamento elétrico vendidas mensalmente entre 1961 e 1972 nos Estados Unidos; o número mensal de passageiros em vôos internacionais entre 1949 e 1960, também nos Estados Unidos. A primeira série apresenta uma tendência de alta crescente, mantendo, porém, uma periodicidade, i.e., uma proporcionalidade dentro de períodos de tempo iguais. Já a segunda série torna-se um pouco mais difícil de prever, uma vez que tanto os valores encontrados como os períodos de regularidade são variáveis, ou seja, tanto o volume de vendas como a sazonalidade variam no tempo. A terceira série por outro lado mantém seus valores aproximadamente constantes, apresentando contudo uma forte sazonalidade.

Portanto, nessas três séries estão contidas combinações de características encontradas em assinaturas: variações de amplitude e freqüência. Da análise do desempenho do modelo recorrente no aprendizado da estrutura básica, ou, por assim dizer, da lei de formação dessas seqüências de números, pode-se inferir sua capacidade de memorização das mesmas no tempo. Na próxima seção serão mostrados os resultados da comparação entre as previsões dos modelos convencional e recorrente.

### IV.3 - Análise do Desempenho

As tabelas abaixo mostram as taxas de erro percentuais para cada série estudada, variando-se o comprimento da seção empregada para treinamento. Ao lado são mostrados valores encontrados em [14] para a mesma série, utilizando-se um rede retropropagada convencional.

Rede Recorrente:

Número de Camadas Diretas	3(3:2:1)
Número de Camadas Recorrentes	2(2:1)
Momento	0,1
Coefficiente de Aprendizado	0,3
Valor de Reset	0,25
Função de Transferência	Sigmóide

Tabela IV.1 - Resumo dos Erros na Previsão da Série do Consumo de Energia  
(Valores em %)

n	Retropropagação		Recorrente			Convencional		
	8	10	12	13	15	12	13	14
n+1	2,67	6,98	2,01	1,19	4,68	3,41	3,07	3,05
n+2	0,91	6,42	0,58	4,14	7,71	-	-	-
n+3	2,91	7,67	5,94	0,62	1,64	-	-	-
menor erro	0,35	0,09	0,58	0,62	0,21	3,41	3,07	3,05

A última linha das tabelas indica o menor erro encontrado para uma previsão de uma ocorrência até 10 termos além da seqüência de treinamento. Note-se que este melhor resultado não coincide sempre com o primeiro valor, sugerindo que a resolução do modelo cause uma resposta melhor dentro de um número inteiro de períodos determinados, ou seja, às vezes o primeiro valor é um "pedaço" de padrão para a rede, encontrando esta resultados excelentes em ocorrências subseqüentes. Além da décima ocorrência, após a série conhecida, os erros crescem bastante.

Tabela IV.2 - Resumo dos Erros na Previsão da Série do Número Mensal de Passageiros de Vôos Internacionais (Valores em %)

n	Retropropagação		Recorrente			Convencional		
	8	10	12	13	15	12	13	14
n+1	5,20	0,41	1,71	18,62	0,55	6,32	4,53	5,00
n+2	20,94	9,93	10,07	24,11	5,61	-	-	-
n+3	19,26	26,29	23,23	11,56	28,58	-	-	-
menor erro	5,01	0,06	1,48	2,29	0,23	6,32	4,53	5,00

É interessante notar que estas duas primeiras séries são as que apresentam a melhor resposta, ao mesmo tempo em que possuem uma periodicidade mais definida. Particularmente a primeira série, de consumo de energia, é a que se assemelha mais a uma assinatura, já que considerando-se os valores relativos dentro de um mesmo

período, um mesmo padrão bem definido repete-se. Seu aprendizado foi o mais eficiente pelo modelo, conforme visto nas tabelas de resultados.

Tabela IV.3 - Resumo dos Erros na Previsão da Série de Venda de um Equipamento Elétrico (Valores em %)

n	Retropropagação		Recorrente			Convencional
	8	10	12	13	15	13
n+1	0,84	9,69	20,56	11,00	16,61	10,2
n+2		3,01	34,25	16,27		-
n+3		13,64	10,76	3,45		-
menor erro	0,84	0,61	0,55	3,45	16,61	10,2

Analogamente, a série de vendas foi a de pior assimilação pelo modelo neuronal. Contudo, ainda que sua previsão não seja muito melhor do que a realizada pelos demais métodos, sua memorização pode ser facilmente obtida, aprimorando-se o treinamento.

É importante ressaltar que os valores apresentados para a previsão utilizando-se uma rede *backpropagation* convencional referem-se ao melhor ponto encontrado. Isso significa que, para cada uma das três séries, a rede foi treinada a cada  $n$  valores toda vez que se desejou prever o termo  $n+1$  deslocando-se o treinamento dentro da mesma série. O modelo recorrente obteve resultados melhores para até  $n+10$ , **sem** novo treinamento, com um desempenho em média superior mesmo para comprimentos da série diferentes daquele apontado como o ótimo pelos métodos numéricos.

O que se pretendeu neste capítulo foi assinalar a facilidade com que o modelo recorrente no tempo, mesmo com pouco treinamento, mostra um desempenho compatível com os demais métodos de previsão. A partir de agora, será tentado o aproveitamento dessa capacidade na identificação de séries temporais.

## CAPÍTULO V

### RECONHECIMENTO DE VOZ

#### V.1 - Sistemas de Reconhecimento de Voz

Durante o processo de reconhecimento de voz, as técnicas de processamento digital constituem normalmente o primeiro passo no que é essencialmente um problema de reconhecimento de padrões. Isto é representado na Figura V.1. Como visto na figura, a representação do sinal de voz (o vetor de padrões) é obtido através de técnicas de processamento digital para voz que extraem as características do sinal sonoro que são relevantes para a identificação do orador. As características mais comuns utilizadas nesses casos, no domínio do tempo, são a energia média do sinal, a amplitude pico a pico, a frequência fundamental e a taxa de cruzamento do zero, entre outros. O padrão resultante é comparado com padrões de referência previamente armazenados e a subsequente lógica para a tomada de decisão é usada para efetuar a escolha entre as várias possíveis alternativas. Existem duas áreas distintas de reconhecimento de orador - verificação do orador e identificação do orador. Em um processo de verificação do orador, uma identidade é solicitada ao usuário, e a decisão esperada da verificação do sistema é estritamente binária, isto é, aceitar ou recusar a identificação alegada, sim ou não. Para fazer a escolha, um conjunto de características escolhidas de modo a reter as informações essenciais relativas a um orador específico é medido através de uma ou mais expressões do mesmo, sendo as medidas resultantes comparadas (sempre usando alguma medida de comparação preferencialmente não linear) a um conjunto de referências padrão armazenado do dito orador. Assim, para verificação do orador, somente uma única comparação entre o conjunto (ou conjuntos) de medidas, e o padrão de referência é necessária para realizar-se a escolha final de aceitar ou rejeitar a identificação pedida. Geralmente, uma distribuição da medida de distância entre os valores fornecidos e a referência guardada é computada. Baseado nos custos relativos de se fazer os dois tipos de erros possíveis (i.e., confirmação de um impostor, ou rejeitar o orador correto) um limite apropriado é ajustado na função distância. É facilmente mostrado que a probabilidade de se incidir nos dois tipos de erro é essencialmente independente do número de padrões de referência guardados no sistema, uma vez que os padrões de referência para todos os outros oradores entram na formação da distribuição estável que caracteriza todos os oradores. Dito em termos mais matemáticos, se for representada a



distribuição de probabilidade do vetor de medida  $x$  para o  $i$ -ésimo orador como  $p_i(x)$ , então uma regra de decisão para a verificação do orador pode ter a forma:

$$\begin{aligned} &\text{confirmar orador } i \text{ se } p_i(x) > c_i p_{med}(x) \\ &\text{rejeitar orador } i \text{ se } p_i(x) < c_i p_{med}(x) \end{aligned} \quad (V.1)$$

onde  $c_i$  é uma constante para o  $i$ -ésimo orador que determina a probabilidade de erro para o  $i$ -ésimo orador, e  $p_{med}(x)$  é a distribuição de probabilidade média (sobre todos os oradores do teste) da medição  $x$ . Note-se que  $p_i(x)$  representa a probabilidade de um determinado conjunto de medidas  $x$  pertencer a um usuário  $i$ . A constante  $c$  é um ajuste que determina um grau de discrepância aceitável para cada orador. Variando-se a constante  $c_i$ , um controle simples da mistura entre os dois tipos de erros é prontamente obtido.

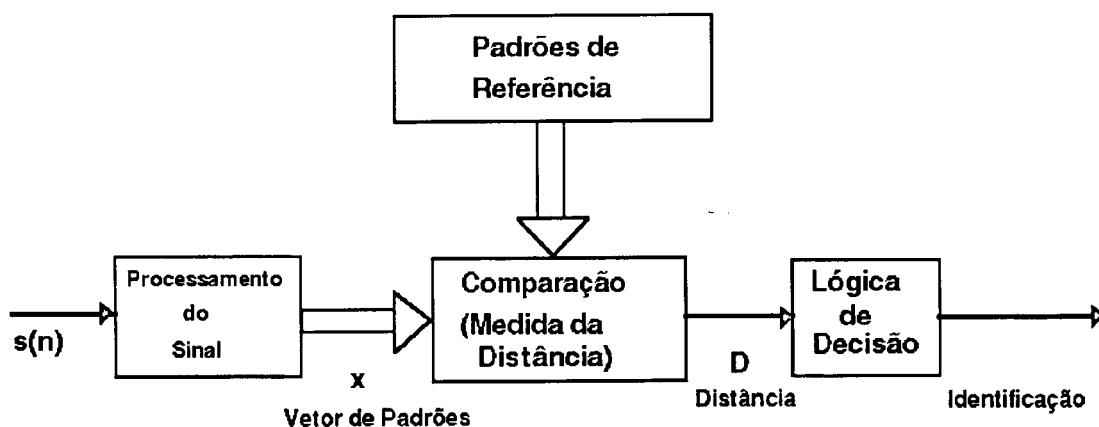


Figura V.1 - Representação Geral do Problema de Reconhecimento do Orador

Já o problema da identificação do orador difere significativamente do anterior. Neste caso é necessário que o sistema faça uma verificação absoluta entre  $N$  oradores da população de onde provém o usuário. Assim, em vez de uma simples comparação entre um conjunto de medições e um padrão referência guardado,  $N$  comparações completas são necessárias. A regra decisória para este sistema é do tipo:

$$\text{escolha o orador } i \text{ tal que } p_i(x) > p_j(x), \quad j = 1, 2, \dots, N, \quad j \neq i \quad (V.2)$$

Ou seja, compara-se a probabilidade de o vetor de medidas  $x$  pertencer ao usuário  $i$  com a probabilidade de ele pertencer aos demais oradores: "escolha o orador com a menor probabilidade absoluta de erro". Neste caso parece plausível que à medida que a população usuária fique muito grande, a probabilidade de erro tenda à unidade, uma vez que um número infinito de distribuições não pode mantê-los distintos num universo finito de parâmetros, ou seja, irá crescer a chance de dois ou mais oradores situarem-se muito próximos entre si. Nestas circunstâncias, uma identificação confiável do orador torna-se impossível.

O objetivo da explanação acima é mostrar que existe um número grande de semelhanças, assim como de diferenças, entre sistemas de verificação de um orador e sistemas de identificação de orador. Cada processo necessita que o orador fale uma ou mais frases testes, que se façam testes nestas frases, e então que compute-se uma (ou mais) funções distâncias entre o vetor de medida e o vetor de referência guardado. Assim, em termos de aspectos do processamento de sinal destes dois problemas, os métodos são bastante similares. A maior diferença ocorre na decisão lógica.

O escopo deste trabalho é justamente o algoritmo que calcula a distância e executa a decisão lógica do reconhecimento. É neste ponto que o algoritmo da rede neuronal recorrente no tempo é introduzido, como será mostrado no próximo capítulo. Antes, porém, é necessário especificar-se e descrever as características levadas em consideração neste trabalho, quando da representação do sinal de voz a ser analisado pela rede neuronal. Isto será feito nas próximas seções.

## V.2 - Análise de Tom

A estimativa do período do tom (ou o que significa o mesmo, a estimativa da frequência fundamental) é um dos mais importantes problemas no processamento da fala. Detetores de tom são utilizados em codificadores de voz, identificação de orador e verificação de sistemas [16], e ajuda para deficientes físicos. Devido à sua importância, muitas soluções para estes problemas já foram propostas [17]. Todos os esquemas propostos têm suas limitações, e é necessário ressaltar que em geral não há um esquema de detetor de tom disponível que forneça resultados satisfatórios para qualquer gama elevada de oradores, aplicações e ambientes de utilização. O que ocorre é uma abordagem adequada a cada caso.

Nesta seção será apresentado um esquema particular de detecção de tom primeiramente proposto por Gold e depois modificado por Gold e Rabiner [18]. As razões para discutir este detetor particular são:

- 1- foi utilizado com sucesso num grande número de atividades;
- 2- é baseado unicamente em processamento no domínio no tempo;
- 3- pode ser implementado para operar muito rapidamente num computador comum, ou pode ser contruído com facilidade um *hardware* digital.
- 4 - ele ilustra o uso dos princípios básicos de processamento paralelo no processamen-to da fala.

Os princípios básicos deste esquema são:

- O sinal de voz é processado de modo a criar uma quantidade de trens de pulso que retenham a periodicidade do sinal original, descartando as formas que são irrelevantes para o processo de deteção do tom.
- Este processo emprega detetores de tom muito simples que podem ser usados para estimar o período de cada trem de pulso.
- A estimativa de vários destes detetores de tom simples são logicamente combinadas para inferir o período da forma de onda da voz.

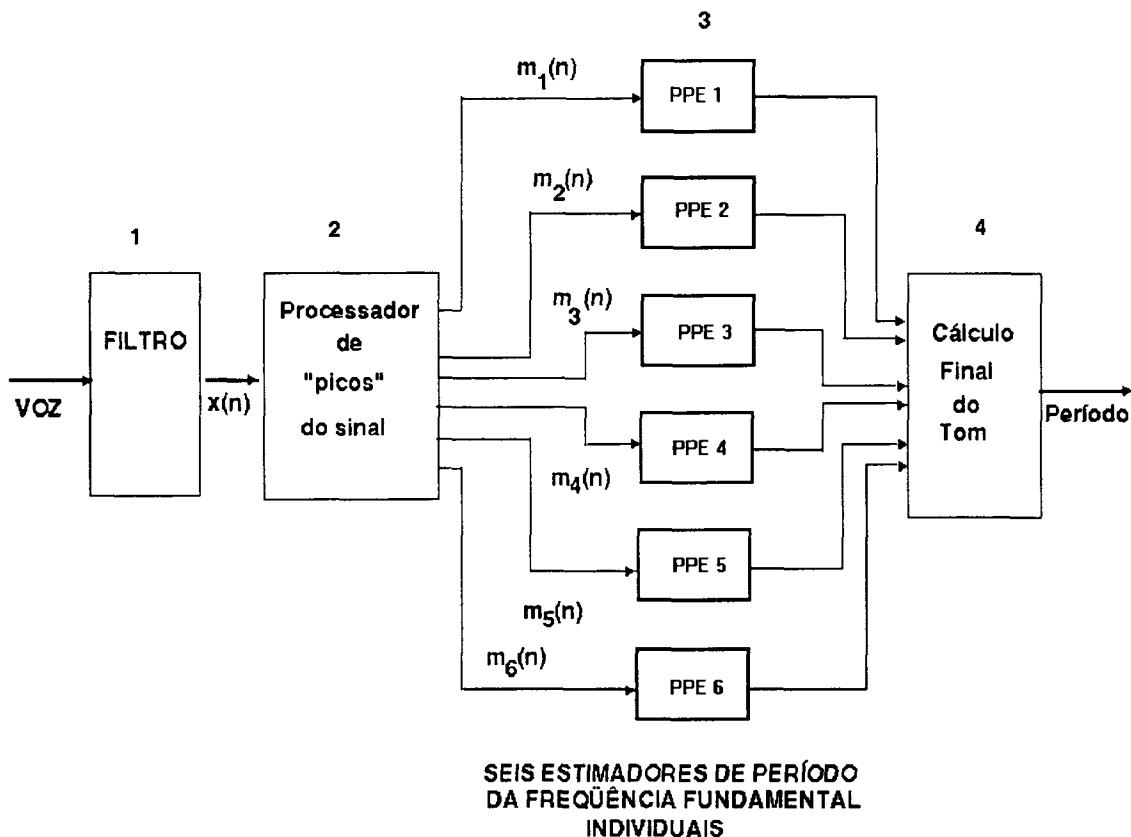


Figura V.2 - Diagrama de Bloco do Processamento Paralelo do Detetor de Tom no Domínio do Tempo

O esquema particular proposto por Gold e Rabiner [18] é apresentado na Figura V.2. A forma de onda da voz é amostrada a uma taxa suficiente para fornecer uma resolução adequada, isto é, se as amostras forem obtidas a uma frequência de 10 kHz, possibilita-se que o período seja determinado dentro de  $T = 10^{-4}$  s. O sinal de fala é filtrado através de um filtro passa-baixa, neste caso com uma frequência de corte em

torno de 900 Hz, a fim de produzir uma forma de onda razoavelmente suave. Um filtro passa-banda passando frequências entre 100 Hz e 900 Hz comumente é necessário para remover ruídos de 60 Hz em algumas aplicações. Esta filtragem poderia ser efetuada ou com um filtro analógico antes da amostragem, ou com um filtro digital depois da amostragem. Não foi o caso neste trabalho. A placa digitalizadora não possui um filtro passa-banda implementado e nenhuma filtragem por software foi realizada posteriormente à digitalização, uma vez que uma das características a serem testadas no modelo é a capacidade de trabalhar com sinais ruidosos. A interferência causada por sinais de 60 Hz, em geral provenientes da rede elétrica envolvida, quando ocorreu, deu-se tanto no processo de aprendizado quanto de verificação, presumivelmente tornando-se irrelevante.

Após a filtragem, os picos e os vales (máximos e mínimos locais) são localizados, e a partir da sua localização e de suas amplitudes, vários trens de pulsos são extraídos do sinal filtrado. Cada trem de pulso consiste de pulsos positivos que ocorrem nos locais ou dos picos ou dos vales. Os 6 casos usados por Gold e Rabiner são:

- 1 -  $m_1(n)$ : Um pulso igual à amplitude do pico ocorre no local de cada pico.
- 2 -  $m_2(n)$ : Um pulso igual a diferença entre a amplitude do pico e a amplitude do vale anterior ocorre a cada pico.
- 3 -  $m_3(n)$ : Um pulso igual a diferença entre a amplitude do pico e a amplitude do pico anterior ocorre a cada pico. (Se esta diferença é negativa o pulso é ajustado em zero).
- 4 -  $m_4(n)$ : Um pulso igual ao negativo da amplitude num vale ocorre a cada vale.
- 5 -  $m_5(n)$ : Um pulso igual ao negativo da amplitude num vale mais a amplitude do pico anterior ocorre a cada vale.
- 6 -  $m_6(n)$ : Um pulso igual ao negativo da amplitude num vale, mais a amplitude do mínimo local anterior ocorre a cada vale. (Se esta diferença é negativa o pulso é considerado igual a zero).

O propósito geral destes trens de pulsos é facilitar a estimativa do período em termos de curto tempo. A operação dos estimadores de períodos de tom simples é descrita na Figura V.3. Cada trem de pulso é processado por um sistema não linear variável no tempo (chamado de janela de circuito de detecção exponencial de pico). Quando um pulso de amplitude suficiente é detetado na entrada, a saída é restabelecida para o valor deste pulso e depois mantida por um intervalo surdo,  $\tau(n)$ , durante o qual nenhum pulso é detetado. No final do intervalo surdo, a saída começa a decair exponencialmente. Quando um pulso excede o nível do decaimento exponencial da saída,

o processo é repetido. A taxa de decaimento e o intervalo surdo são dependentes as mais recentes estimações do período do tom. O resultado é um tipo de suavização do trem de pulso, produzindo uma sequência de pulsos quase estáticos como mostrado na Figura V.3. O comprimento de cada pulso é uma estimativa do período do tom. O período do tom é estimado periodicamente (por exemplo, 100 vezes/s) através da medida do comprimento do pulso medindo o intervalo de amostragem.

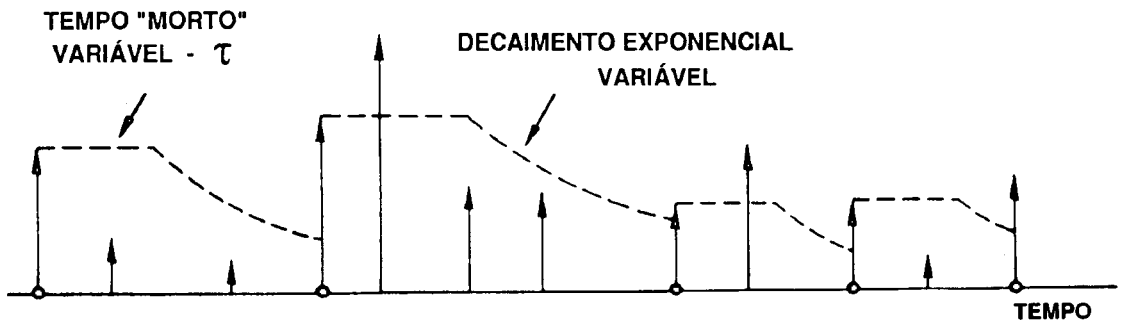


Figura V.3 - Operação Base de Cada Estimador de Período de Tom Individual do Detetor de Tom no Domínio do Tempo

Esta técnica é aplicada para cada um dos seis trens de pulso deste modo obtendo seis estimativas do período do tom. Estas seis estimativas são combinadas com duas das mais recentes estimativas para cada um dos seis detetores de tom. Estas estimativas são então comparadas e o valor com maior número de ocorrência (dentro de alguma tolerância) é declarado o período do tom naquele instante. Este procedimento produz uma estimativa muito boa do período da voz falada. Para fala sem voz existe uma falta de consistência distinta entre as estimativas. Quando esta falta de consistência é detetada, a fala é classificada como sem voz. Todo o processo é repetido periodicamente para produzir uma estimativa do período do tom e da classificação da voz/não voz como uma função do tempo.

### V.3 - Cálculo da Energia

Observa-se claramente na Figura VI.3 que a amplitude do sinal de voz varia apreciavelmente com o tempo. Em particular, a amplitude de segmentos sem voz é geralmente muito menor do que as dos segmentos com voz. A chamada energia de curto termo do sinal de voz permite uma representação adequada que reflete estas variações de amplitude [18]. Em geral, pode-se definir a energia de curto termo como:

$$E_n = \sum_{m=-\infty}^{\infty} [x(m) \cdot w(n-m)]^2 \quad (\text{V.3})$$

Esta expressão pode ser reescrita como:

$$E_n = \sum_{m=-\infty}^{\infty} x^2(m) \cdot h(n-m) \quad (\text{V.4})$$

onde

$$h(n) = w^2(n) \quad (\text{V.5})$$

A Equação (V.4) pode ser interpretada como descrito na Figura V.4. Isto é, o sinal  $x^2(n)$  é filtrado pelo filtro linear com a resposta impulso  $h(n)$  como dado em (V.5).

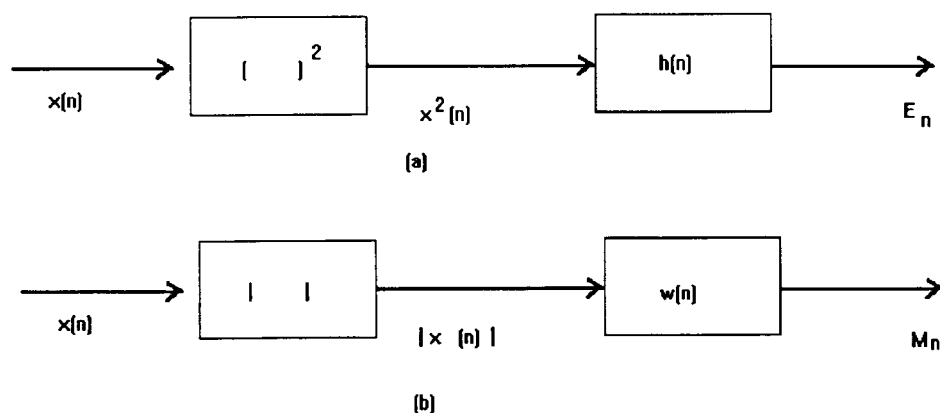


Figura V.4 - Representação em Diagrama de Blocos de (a) Energia de Curto Termo; e (b) Amplitude Média de Curto Termo

O maior significado de  $E_n$  é que ele proporciona a base para distinguir os segmentos com voz dos sem voz. Os valores de  $E_n$  para os segmentos sem voz são significativamente menores do que os com voz, como pode ser visto na Figura VI.2. A função de energia pode ser usada para localizar aproximadamente o instante de tempo em que a voz falada fica sem som, e vice versa, e para falas de alta qualidade, (alta taxa de sinal-voz), a energia pode ser utilizada para distinguir a fala do silêncio.

Uma dificuldade com a função de energia de curto termo como definido em (V.4) é que ela é muito sensível a níveis de sinal muito largos (uma vez que eles entram na Equação (V.4) ao quadrado), deste modo enfatizando as grandes variações entre amostras em  $x(n)$ . Um modo simples de melhorar este problema é definir uma função de amplitude média

$$M_n = \sum_{m=-\infty}^{\infty} |x(m)| \cdot w(n-m) \quad (\text{V.6})$$

onde o somatório do peso de sinais de valores absolutos é computado ao invés do somatório dos quadrados. A Figura V.4b mostra como a Equação (V.6) pode ser implementada como uma operação de filtragem linear em  $|x(n)|$ . Nota-se que uma simplificação de aritmética é alcançada eliminando-se a operação de elevar ao quadrado.

#### V.4 - Processamento do Sinal

O problema de localizar o início e o fim da fala da expressão num fundo de ruído é importante em várias áreas do processamento de sinal. Em particular, no reconhecimento automático de palavras isoladas, que é o caso desse trabalho, é essencial localizar as regiões do sinal de voz que corresponde a cada palavra. Um esquema para localizar o início e o fim do sinal de voz pode ser utilizado para eliminar cálculos significativos em sistemas que não sejam em tempo real tornando possível o processo somente de parte da entrada que corresponde a fala.

A questão de discriminar a fala do ruído de fundo não é trivial, exceto no caso de ambientes acústicos com uma relação de som-para-ruído extremamente alta - isto é, gravações de alta fidelidade feita em quartos à prova de som. Para tal ambiente com relação sinal-ruído alta, a energia do menor nível de som de fala (por exemplo consoantes, fricativas fracas) excede a energia do ruído de fundo, e assim uma simples medida de energia será suficiente. No entanto, as condições de gravações ideais não são práticas para a maioria das aplicações.

Um algoritmo que pode ser utilizado para esse tipo de determinação proposto por Rabiner e Sambur [18], lança mão de duas medidas simples, ambas no domínio do tempo: o cálculo da energia média, descrito na seção anterior, e a taxa de cruzamento do zero. Essas duas características, combinadas, na maioria das vezes permitem que as bordas do sinal de fala possam ser determinadas, permitindo que o processamento posterior do sinal possa abstrair-se do ruído de fundo.

O algoritmo pode ser melhor compreendido referenciando-se à Figura V.5. As representações básicas usadas são o número de passagens da curva pelo valor zero em cada divisão de 10 ms (Equação V.7) e a amplitude média (Equação V.6) calculada também em uma divisão de 10 ms. Ambas as funções são calculadas para todo o

intervalo de gravação numa taxa de 100 vezes/segundo. É assumido que os primeiros 50 ms do intervalo não contêm fala. A Equação V.7 é apresentada a seguir.

$$Z_n = \sum_{m=-\infty}^{\infty} |\text{sgn}[x(m)] - \text{sgn}[x(m-1)]| \cdot w(n-m) \quad (\text{V.7})$$

onde

$$\begin{aligned} \text{sgn}[x(n)] &= 1 & x(n) \geq 0 \\ &= -1 & x(n) < 0 \end{aligned} \quad (\text{V.8})$$

e

$$\begin{aligned} w(n) &= \frac{1}{2N} & 0 \leq n \leq N-1 \\ &= 0 & \text{o resto} \end{aligned} \quad (\text{V.9})$$

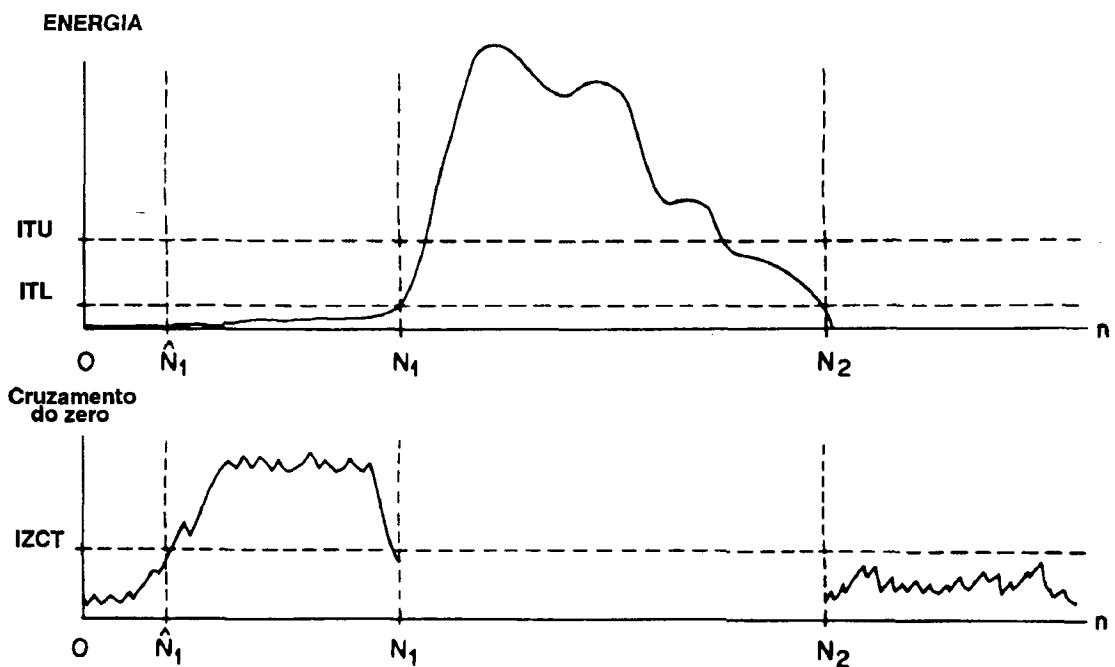


Figura V.5 - Exemplo Típico da Amplitude Média e do Cruzamento por Zero das Medições para uma Palavra com uma Fricativa Forte no Início

A média e o desvio-padrão da amplitude média e a taxa de cruzamento por zero são calculadas para este intervalo para dar uma caracterização estatística do ruído de fundo. Utilizando esta caracterização estatística e a amplitude média máxima no intervalo, são calculados a taxa de cruzamento por zero e os limites de energia. O perfil da amplitude média é então percorrido a fim de se encontrar o intervalo onde sempre seja excedido um limite (ITU na Figura V.5). Assume-se que o ponto inicial e final ficam fora



deste intervalo. Então, combinando para trás a partir do ponto onde  $M_n$  primeiro excede o valor do limite ITU, o ponto (chamado  $N_1$  na Figura V.5) onde  $M_n$  primeiro cai abaixo do limite ITL é seleccionado através de tentativas como o ponto inicial. Um procedimento semelhante é feito em seguida para descobrir o possível ponto final  $N_2$ . Este procedimento duplo para os limites assegura que quedas na função de amplitude média não mascarem o ponto final. Neste estágio é razoavelmente seguro assumir que os pontos iniciais e finais não estejam no intervalo  $N_1$  a  $N_2$ . O próximo passo é deslocar-se para trás de  $N_1$  (em direcção a  $N_2$ ) comparando a taxa de cruzamento por zero para um limite (IZCT na Figura V.5) determinado por estatística da taxa de cruzamento po zero para o ruído de fundo. Isto é limitado a 25 divisões precedendo  $N_1$  (seguindo  $N_2$ ). Se a taxa de cruzamento por zero exceder o limite 3 vezes ou mais, o ponto inicial  $N_1$  é movido para trás para o primeiro ponto aonde o limite de cruzamento por zero foi excedido. De outro modo  $N_1$  é definido no início. Um procedimento semelhante é realizado para o fim. O sinal é então considerado como fala entre o início da janela do ponto  $N_1$  e o fim da janela do ponto  $N_2$ .

## CAPÍTULO VI

### APLICAÇÃO EM RECONHECIMENTO DE VOZ

#### VI.1 - Adequação do Modelo

No Capítulo IV foi mostrada a aplicação do modelo de *backpropagation* recorrente naquela que é a área de trabalho típica das redes de retropropagação, a previsão. O que se pretende aqui é explorar as características do modelo estudado na tentativa de aplicá-lo também no **reconhecimento** de séries temporais, ou seja, na capacidade de produzir uma determinada seqüência de padrões como resposta a uma série específica de estímulos. Em princípio não existe a necessidade de se reproduzir a série de entrada.

Duas características aparentam ser aproveitáveis para tal. Em primeiro lugar, como já foi dito, o que torna este modelo de rede especial é sua capacidade de aprender a produzir uma sucessão de padrões dependente do contexto do estímulo. Nenhuma outra rede de retropropagação faz isto. Além disso, já que a camada de entrada da rede observa os tempos anteriores nas imediações de cada grupo de padrões, um grupo de cada vez, não há a necessidade de se apresentar a série inteira de uma só vez, como acontece em implementações *backpropagation* convencionais, o que torna a rede ótima para aprender tipos de assinaturas complexas e longas, onde abordagens utilizando-se "instantâneos" da série não são adequadas.

Assim, trata-se de aproveitar a capacidade do modelo de assimilar a estrutura básica de uma seqüência e convertê-la num instrumento de medida do desvio de uma série observada em relação a uma série padrão. As características apontadas no parágrafo anterior indicam a adequação, não só para o reconhecimento de séries caóticas (baixa coerência) como também possuidoras de um número relativamente elevado de termos. Um sinal de voz representado no domínio do tempo encaixa-se nos dois casos.

Na verdade para a tarefa de puramente reconhecer uma seqüência no tempo, não haveria, inicialmente, a necessidade de empregar-se um modelo recorrente no tempo. Contudo, o desempenho superior apresentado quando da previsão de uma série credencia esse tipo de rede a uma adaptação proveitosa.

A adaptação implementada no presente trabalho foi determinada por dois fatores principais : o direcionamento para padrões importantes em uma onda de voz e o volume de processamento computacional que seria necessário.

Conforme visto no Capítulo V, um reconhecimento eficaz de um sinal de voz pode ser realizado observando-se, entre outras características, sua frequência fundamental e sua energia. Esses valores, obtidos a partir da onda, são calculados em "janelas", grupos de impulsos no tempo, ao longo de toda série, o que gera uma segunda seqüência de grandezas, diretamente associada à série original. O que se pretende é demonstrar a possibilidade do modelo reconhecer a série original. Em vez de trabalhar-se diretamente a partir de uma série de amplitudes sonoras, que possui em torno de 10.000 termos por segundo armazenado, optou-se por reconhecer duas séries derivadas, a de tons (frequências fundamentais) e a de energias médias, que apresentam aproximadamente 100 termos cada. O ponto crucial aqui não é o processo de reconhecimento propriamente dito, que se efetua na mesma ordem de tempo da *backpropagation* convencional, e sim o período de treinamento da rede, acentuadamente mais longo nos modelos recorrentes.

É preciso ressaltar, contudo, que na verdade é essa transformação da série no tempo para uma série de características derivadas que produz uma assinatura sonora, no caso, a do orador associado. A energia e a frequência fundamental de uma voz caracterizam-na intrinsecamente, exibindo as influências físicas específicas da pessoa que fala. O sinal no tempo é uma superposição de várias influências anatômicas, e por isso mesmo apresenta sua estrutura básica mascarada, sendo difícil seu reconhecimento. As séries de energia e tom empregadas neste trabalho, paralelamente ao fato de reduzirem as exigências computacionais, aproximam-se conceitualmente mais ainda das assinaturas sonoras industriais que motivaram o estudo.

Logo, para que a série de impulsos sonoros que forma o sinal de voz a ser testado seja considerada como tendo sido gerada pelo mesmo orador, cada grupo de  $N$  impulsos tem seu tom e sua energia calculados e a série resultante é submetida à rede. A discrepância entre as séries observadas e a original é avaliada, termo a termo, para que seja estabelecida a similaridade. Esse grau de discrepância tolerado nos casos de validação do orador é obtido a partir de um segundo nível de treinamento, expondo-se a rede a oradores diferentes e a múltiplas ocorrências da mesma fala pelo orador em questão.

Diferentemente dos processos convencionais, onde cada valor de tom e energia é apenas comparado tempo a tempo, com pequena possibilidade de variação, a rede admite alterações sutis de tempo na fala padrão, aumentando o poder de generalização do sistema de reconhecimento.

Antes de mais nada é preciso considerar-se que, numa aplicação prática envolvendo assinaturas sonoras industriais, o processo de identificação do sinal descrito no Capítulo V seria o desejado: dispor-se-ia de tipos conhecidos de sinais provenientes de corpos de prova e o desejado seria que o sistema classificasse um sinal de teste. Diferentemente dos sinais de voz, nesses casos não é esperado que a probabilidade de erro tenda à unidade, já que o universo considerado não é muito grande. Contudo, na análise que se efetuou com os sinais de voz, o processo de classificação de um orador implicaria em série de maior resolução, portanto maiores, ou um número maior de características [19, 20, 21], portanto um número maior de séries. Além disso, o treinamento deveria ser mais exaustivo, os erros do aprendizado teriam que se aproximar mais de zero. A eficácia da rede neuronal, entretanto, não precisa ser posta à prova somente desta forma. O processo de validação de um orador, correspondente à memorização de um determinado sinal de voz, demonstra claramente a capacidade do modelo de memorizar seqüências temporais, com a vantagem operacional de necessitar de períodos de treinamentos extremamente curtos e de um volume de processamento muito menor. A diferença maior residiria na precisão do cálculo da função distância, posteriormente ao reconhecimento. Dessa forma, a abordagem escolhida foi considerada mais adequada não só por sua maior praticidade, mais pela proximidade maior, no que diz respeito ao grau de dificuldade, à atividade de reconhecimento que mostram a pesquisa.

Em seguida será descrita a montagem experimental destinada a captar e processar o sinal de voz, além dos detalhes de implementação da rede neuronal.

## **VI.2 - Aquisição de Dados**

A Figura VI.1 mostra o diagrama de blocos da montagem empregada para a aquisição dos dados do sinal de voz.

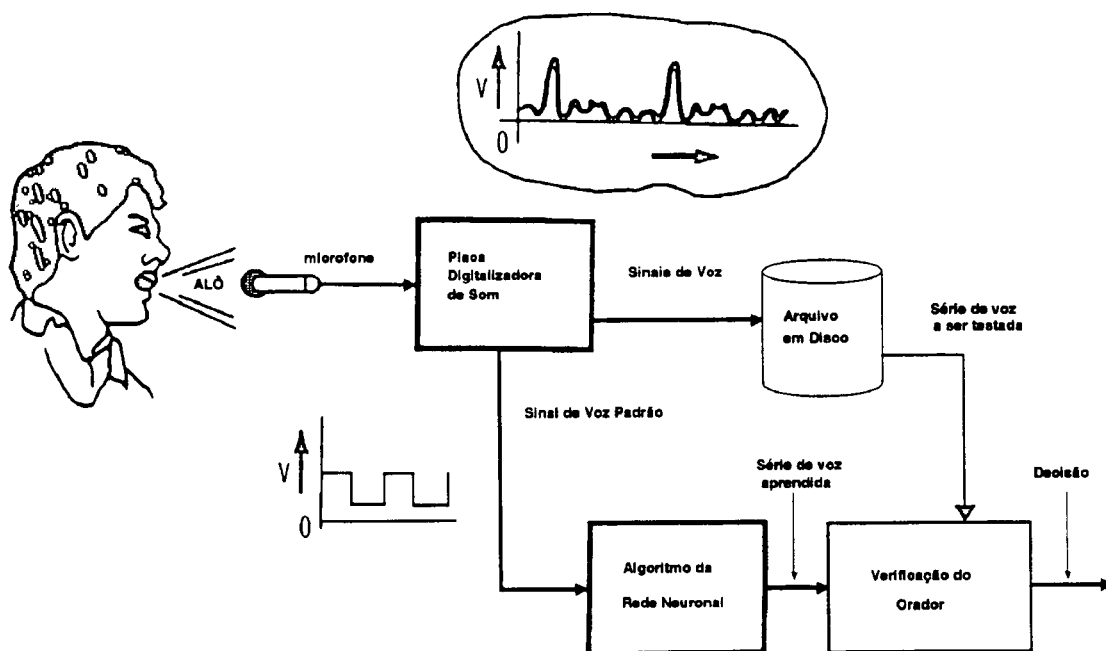


Figura VI.1 - Diagrama de Blocos da Montagem

O ambiente utilizado baseia-se na arquitetura IBM-PC, sendo que o microcomputador utilizado possui uma CPU 386 de 40 MHz de frequência de "clock", bastante comum. A digitalização do sinal foi realizada através de uma placa Sound Blaster 16, produto da empresa Creative Labs Inc. [22], que tem sua especificação relacionada no Apêndice C, onde também se encontram os parâmetros do microfone escolhido : modelo MK-2 fabricado pela empresa Leson Ltda, dinâmico, unidirecional.

Foi escolhida uma expressão padrão, "OBRIGADO", pronunciada por 5 oradores, cada um deles, repetindo a operação 3 vezes. Um orador foi escolhido para ter uma de suas falas aprendidas pela rede. Essa fala, o sinal padrão, na realidade foi calculada tomando-se a média aritmética de 7 capturas de voz. Essa média constitui um sinal, e mais dois outros não utilizados no cálculo foram reservados para teste. Os demais oradores, bem como todas as falas do escolhido foram utilizados durante os procedimentos de teste do sistema. As falas foram limitadas a um tempo máximo de 3 segundos, durante a captura pelo microfone, que se deu sempre no mesmo local, com um nível de ruído ambiente aproximadamente constante e à mesma distância do orador.

Os sinais de voz foram convertidos em dados de 16 bits, digitalizados a uma frequência de amostragem de 10 kHz, em um só canal, sendo submetidas a um filtro passa-baixa de 900 Hz de frequência de "cut-off". O armazenamento em arquivo deu-se

segundo o formato empregado pelo programa de gravação, VREC, fornecido juntamente com a placa de som, que produz arquivos de extensão ".VOC", cujo formato é apresentado no Apêndice D.

Visando a uma padronização da entrada, os sinais foram processados por dois algoritmos, descritos na próxima seção, e em seguida apresentados à rede para as sessões de treinamento.

### VI.3 - Preparação dos Dados

Antes da série de amplitudes sonoras, diretamente resultante da digitalização, estar pronta para ser utilizada na rede neuronal, três processos são necessários : a determinação do início e fim da fala, o cálculo das energias médias e o cálculo dos tons.

A determinação do início e fim da fala, ou seja, a identificação dos trechos de voz/não voz encontrados no sinal original foi realizada implementando-se o algoritmo descrito no capítulo anterior, seção V.4. A onda foi subdividida em janelas de 10 ms, sendo que os primeiros 50 ms de gravação foram empregados para obtenção dos parâmetros estatísticos relativos ao silêncio. Conforme foi visto, são calculados para estas janelas a taxa de cruzamento do zero e o valor das energias, utilizados na determinação dos referidos períodos de voz/não voz. Assim, o processo do cálculo das amplitudes médias das energias está embutido nessa primeira etapa. A taxa de cruzamento do zero não foi aproveitada como característica definidora da série: essa grandeza é útil para o processo de detecção do silêncio, mas é também ambígua no tocante à representação do sinal propriamente dito. O terceiro processo necessário, o cálculo dos tons, ou frequência fundamental, seguiu o algoritmo descrito na seção V.3 do capítulo anterior. Uma vez mais as janelas foram mantidas em 10 ms, com cada uma delas gerando um valor estimado de frequência fundamental.

Tanto o sinal de voz padrão como os sinais a serem verificados foram submetidos aos processos acima descritos. Deste ponto em diante o tratamento se diferencia : as duas séries que descrevem o sinal padrão, energia e tom, são utilizadas para o treinamento de duas redes neurais recorrentes no tempo que aprendem as seqüências correspondentes. Os demais pares de séries são empregados nas seções de validação do método. Esses últimos, antes de serem apresentados às redes já treinadas, passam por um processo de *time-warp*, um cisalhamento no tempo, de modo a apresentar o mesmo número de termos da série treinada. Esse ajuste foi linear, segundo a Equação VI.1.

$$\omega(t) = \left(1 - \frac{\text{TamSériePadrão} - \text{TamSérieTeste}}{\text{TamSériePadrão}}\right) \cdot t \quad (\text{VI.1})$$

Neste ponto os sinais estão prontos para serem utilizados. A seguir é descrito como essa operação foi realizada.

#### VI.4 - Rede Utilizada

Seguindo o princípio geral do paradigma do modelo de retropropagação recorrente, descrito no Capítulo III, foi implementada uma rede neuronal com os seguintes atributos:

- 3 camadas "diretas" e 2 camadas recursivas, sendo nas camadas diretas: 5 neurônios na primeira camada, 2 na segunda e 1 neurônio na saída. Nas camadas recursivas: 2 na intermediária e 1 na saída, completamente interligados, conforme o diagrama abaixo.

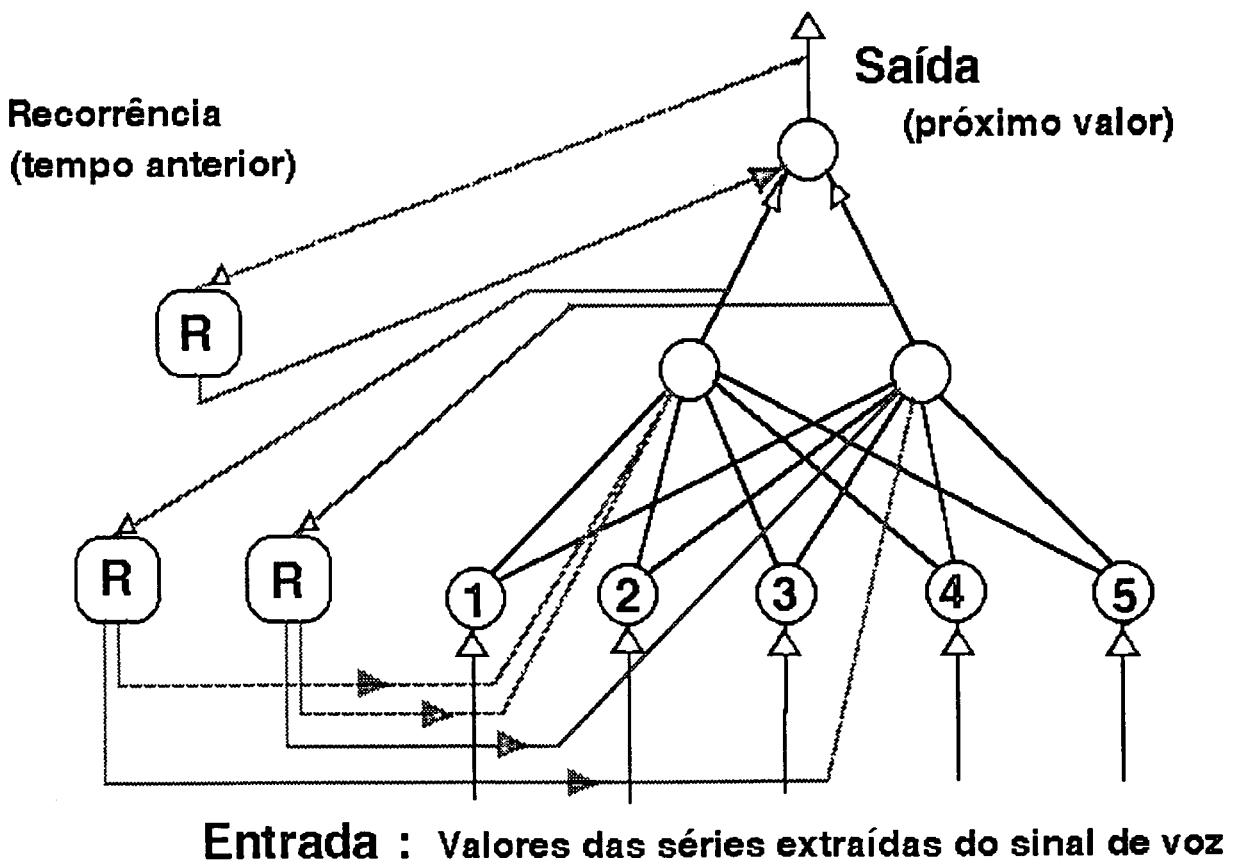


Figura VI.2 - Diagrama Esquemático da Rede Neuronal Utilizada

Dessa forma, uma amostra média de um orador, de agora em diante denominado orador A, foi considerada a voz padrão para fins de reconhecimento. Esse sinal de voz passou por todas as etapas de processamento descritos anteriormente, até ser transformada em duas séries temporais de valores de energia média e de estimativas de tom, ambas possuindo, ao fim da preparação, 91 termos cada.

Como na primeira camada da rede é composta de 5 unidades, as séries foram divididas em 18 grupos de 5. A cada grupo utilizado como entrada foi associada uma saída desejada cujo valor era o primeiro termo da "janela" seguinte, sendo o 91º termo utilizado no último grupo.

Para as sessões de treinamento, algumas otimizações foram empregadas. Em primeiro lugar, os valores tanto de energia como de tons foram normalizados segundo a fórmula da Equação (VI.2), a fim de evitar a saturação das sinapses. Essas, por sua vez, foram iniciadas com um valor fixo dado pela expressão (VI.3).

$$\begin{aligned} S'(Z) &= S(Z) \cdot Escala \\ Escala &= \frac{0,2}{S_{máx} - S_{mín}} \end{aligned} \quad (VI.2)$$

$$w_{ij} = \frac{1}{\sqrt{K_i}} \quad (VI.3)$$

onde  $K_i$  é o número de ligações aferentes no neurônio  $i$ , o que corresponde a  $1/\sqrt{5}$  e  $1/\sqrt{2}$  nas 2ª e 3ª camadas respectivamente. Essa iniciação baseia-se numa estratégia simples de tentar manter o valor inicial dos neurônios próximos de 1, de modo a evitar mínimos locais.

O processo de treinamento foi descrito no Capítulo III, acrescido da técnica de otimização do aprendizado delta-bar-delta estendido (EDBD), descrita no Capítulo II. Os parâmetros do EDBD foram os seguintes:

- $K_\alpha$  - fator linear da taxa de aprendizado = 0,7
- $K_\mu$  - fator linear da taxa de momentum = 0,3
- $\gamma_\alpha$  - fator exponencial da taxa de aprendizado = 0,05
- $\gamma_\mu$  - fator exponencial da taxa de momentum = 0,05
- $\phi_\alpha$  - fator de decremento linear da taxa de aprendizado = 0,5
- $\phi_\mu$  - fator de decremento linear da taxa de momentum = 0,2
- $\alpha_{max}$  - limite superior da taxa de aprendizado = 2,0



$\mu_{\min}$  - limite superior da taxa de momentum = 0,9

$\lambda$  - parâmetro de tolerância para atualização = 3,0

$\theta$  - fator de convexidade dos pesos = 0,3

A função de transferência empregada foi a sigmóide convencional.

As sessões de treinamento foram limitadas a um máximo de 500.000 iterações, onde o erro do aprendizado mostrou-se pequeno o suficiente para o correto reconhecimento do padrão, como pode ser visto nas saídas do programa no apêndice A.

Os demais sinais de voz, após passarem pela preparação, são apenas processados pelas redes, isto é, a camada de entrada de ambas as redes é alimentada com as séries correspondentes e a distância entre o valor obtido e o esperado, este último proveniente da previsão do primeiro valor de cada grupo de 5 neurônios é medido.

Dois ajustes são ainda efetuados de modo a que os padrões aproximem-se mais eficientemente. O primeiro é uma normalização referida à curva padrão, isto é, todos os valores dos sinais utilizados para teste são submetidos a uma transformação linear que os exprime em função dos valores máximo e mínimo da série considerada referência. Em outras palavras, é mantida uma proporcionalidade entre os padrões de modo a desprezar as variações devidas exclusivamente ao volume de gravação.

O segundo ajuste é conhecido como "torsão no tempo", *time warp*, e é simplesmente um processo que tenta minimizar as variações ocorridas na duração dos fonemas, ou seja, uma vez que o mesmo orador dificilmente manterá uma constância temporal precisa, um mesmo som pode demorar mais em uma tentativa de verificação do que na gravação original. O *time warp* empregado foi linear, segundo a equação (VI.1). O que acontece é uma escolha dinâmica do valor correspondente ao tempo a ser apresentado à rede, em função da diferença de comprimento das seqüências envolvidas.

Os resultados são mostrados a seguir.

## VI.5 - Análise do Desempenho

Os resultados obtidos são apresentados através de dois tipos de tabelas. Primeiramente, listou-se a distância apontada pela rede neuronal entre o padrão apresentado e o padrão que ela "conhece", energias médias ou estimativas de tom, conforme o caso. Essa distância mede o quanto a rede considera semelhantes a série aprendida e a série observada. Em seguida, os erros médios encontrados nas primeiras tabelas foram justapostos e combinados em um único índice, ao mesmo tempo sendo confrontados com o diagnóstico esperado, ou seja, a validação correta do orador em questão. Desses dados pode-se então inferir a eficácia do modelo na atividade de memorização e verificação de uma assinatura sonora.

A Figura VI.3 mostra o perfil no tempo do sinal de voz do orador A, o padrão de voz aprendido pela rede através de suas séries derivadas. Pode ser notado que a palavra escolhida imprime suas características no sinal : as sílabas aparecem bem definidas. Apenas se fosse desejada a identificação dos fonemas de cada uma delas entretanto, é que seria necessário levar em conta outras propriedades da onda.

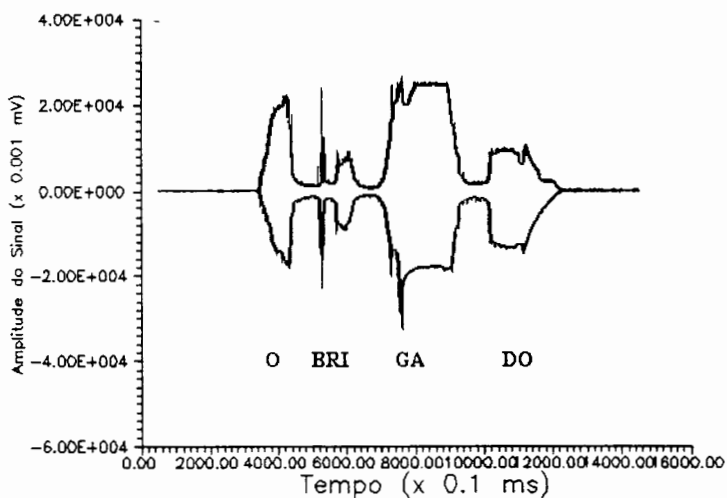


Figura VI.3 - Sinal de Voz no Tempo (Padrão de Teste)

Tabela VI.1 - Energias Médias (Erro em %)

Orador	Amostra	Sem <i>warp</i>	Com <i>warp</i>	p.u. sem <i>warp</i> <sup>2</sup>
Padrão <sup>1</sup>	-	17,3	17,3	1,0
A	1	32,1	36,4	
	2	31,9	40,1	
	3	39,0	23,6	
	4	50,7	38,3	
	5	40,8	32,8	
	6	83,2	64,8	
	7	62,0	25,2	
	8	76,7	52,4	
	9	24,1	16,4	
	média	48,94	36,67	2,83
B	1	129,2	38,4	
	2	137,5	44,7	
	3	138,4	41,6	
	média	135,05	41,57	7,80
C <sup>3</sup>	1	145,7	38,7	8,42
	média	145,7	38,7	8,42
D <sup>3</sup>	1	88,9	49,2	
	2	252,2	86,0	
	média	170,55	67,60	9,86
E	1	117,2	88,1	
	2	130,3	80,7	
	3	254,7	95,3	
	média	167,4	88,03	9,67

Nota:

- 1 - O padrão foi criado a partir das amostras 1 a 7 do orador 1.
- 2 - Valores em p.u. (*per unit*) em relação ao padrão.
- 3 - Não foi possível aproveitar todas as amostras destes oradores.

Nesta primeira tabela, o erro total indicado é a soma dos erros encontrados em cada uma das associações realizadas pela rede entre os termos da série sendo testada e a previsão do próximo termo na seqüência, calculada baseando-se no padrão original memorizado. O valor *média* refere-se à média aritmética das distâncias. Esses valores envolvem apenas as séries de amplitudes médias de energia. A soma dos erros foi ponderada levando-se em consideração os erros do processo de treinamento. Assim, em

uma série de 18 termos, desprezaram-se os erros encontrados nas previsões dos tempos 6 e 18, cujos valores eram bem menos acurados que os demais, sendo aproveitados os 16 tempos restantes. Foram comparados os resultados obtidos empregando-se os ajustes de *time warp* e escala mencionados anteriormente, sendo notado que, especificamente para o processo de **validação do orador** tal procedimento não parece ser acertado : ocorre uma aproximação de todos oradores ao orador padrão, uma vez que diminui-se drasticamente o peso de características vocais como volume de voz e velocidade de fala. Na verdade, o que aparentemente ocorre é que a rede torna-se capaz de reconhecer não as características do orador, mas sim da palavra proferida. Se por um lado isto é um indício de que o modelo pode efetivamente ser aproveitado no reconhecimento de comandos vocais, por outro desvia o processo de seu objetivo original, qual seja a identificação de assinaturas, as quais precisam preservar características de seus geradores. Para o estudo corrente e envolvendo as características temporais escolhidas, essa identificação dá-se mais adequadamente sem os ajustes de amplitude e tempo, uma vez que as condições de obtenção dos sinais de voz foram mantidas constantes para todos os oradores. O não aproveitamento de duas amostras do orador C e de uma do orador D deveu-se a distorções extremas no sinal obtido, ocasionadas por um volume insuficiente da fala e por um tempo gasto exagerado, o que causou o aparecimento de séries derivadas demasiadamente discrepantes. Uma vez que nenhum dos oradores envolvidos era o padrão, a inclusão dessas séries espúrias viciaria o teste. Assim sua eliminação foi decidida, sem prejuízo para o ensaio executado.

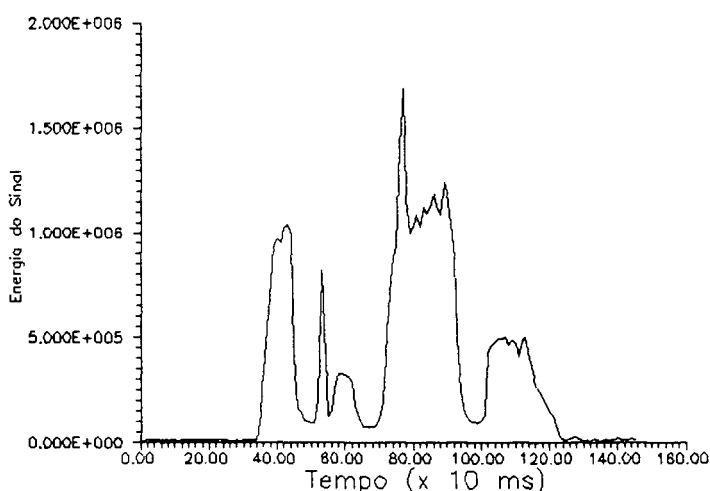


Figura VI.4 - Série de Amplitudes Médias de Energia no Tempo

O reconhecimento da série de amplitudes médias de energia mostrou-se o mais importante no ato de validação do orador. Note-se que mesmo havendo erros elevados em duas amostras do orador padrão, a média das distâncias encontradas pela rede para o

orador A é bastante inferior às dos demais oradores. Na Figura VI.4 é mostrado o perfil da energia média do sinal padrão. A curva apresenta-se como uma assinatura nítida, o que pode justificar esse papel decisivo que sua identificação tem no processo.

Tabela VI.2 - Estimativas de Tom (Erro em %)

Orador	Amostra	Sem <i>warp</i>		Com <i>warp</i>		p.u. sem <i>warp</i> <sup>1</sup> do último tempo
		Total	Último tempo	Total	Último tempo	
Padrão <sup>2</sup>	-	1,7	0,9	1,7	0,9	1,0
A	1	10	1,9	8,2	0,1	
	2	4,3	1,5	6,0	3,3	
	3	6,1	7,0	8,7	4,6	
	4	10,9	10,7	6,6	2,4	
	5	4,7	5,5	7,1	3,3	
	6	7,4	18,4	8,0	8,0	
	7	7,0	4,8	5,4	3,3	
	8	4,7	4,5	8,4	3,4	
	9	6,9	2,0	10,2	23,7	
	média	6,88	6,25	7,6	5,79	6,94
B	1	7,7	19,4	8,1	0,02	
	2	9,7	19,4	4,9	1,3	
	3	6,2	20,4	2,6	2,2	
	média	7,8	19,7	5,2	1,17	21,89
C <sup>3</sup>	1	6,9	18,8	4,0	2,1	
	média	6,9	18,8	4,0	2,1	20,89
D <sup>3</sup>	1	8,5	2,2	13,0	5,6	
	2	9,3	2,2	18,7	21,3	
	média	8,9	2,2	15,9	13,45	2,44
E	1	7,1	24,0	6,3	1,2	
	2	9,1	19,0	10,1	6,3	
	3	9,8	19,5	13,9	13,2	
	média	8,67	20,83	10,1	6,9	23,1

Nota:

- 1 - Valores em p.u. (*per unit*) em relação ao padrão.
- 2 - O padrão foi criado a partir das amostras 1 a 7 do orador 1.
- 3 - Não foi possível aproveitar todas as amostras destes oradores.

Já na segunda tabela, onde os valores referem-se ao reconhecimento das séries de estimativas de frequência fundamental, o erro apontado não é o erro global e sim o erro apresentado pela previsão do último termo da série. A razão dessa escolha é a origem dos termos da série em questão. Os valores são, conforme foi dito, **estimativas sucessivas** da frequência fundamental da voz do orador. As assinaturas assim obtidas possuem um comportamento peculiar, assumindo valores parciais convergentes, aproximando-se cada vez mais de um valor constante, a estimativa final de tom. Dessa forma, a séries em sua maioria apresentam valores iniciais extremamente próximos, como visto na Figura VI.6, afastando-se à medida que avançam no tempo. Logo, o erro de previsão do último tempo possui um significado muito maior que um erro médio que levasse em consideração os pequenos erros do início das séries. Nesses valores de tons, mais ainda que na tabela anterior, os ajustes de tempo e escala mostram-se desastrosos, uma vez que confundiriam totalmente curvas que, pelo próprio método de obtenção, já encontram-se bastante próximas. Aqui, também, para efeito de treinamento e levando-se em conta que a rede é recorrente, o cálculo do erro global de aprendizado foi ponderado, desprezando-se os 5 primeiros tempos da seqüência. Privilegiou-se assim, uma vez mais, o trecho mais significativo do final da série de tons.

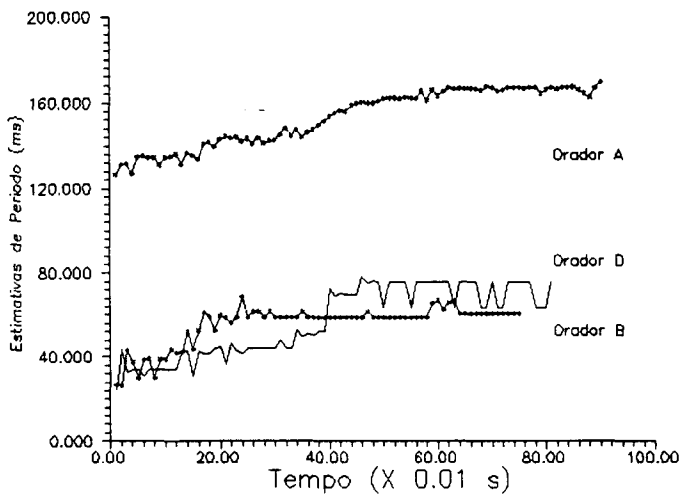


Figura VI.6 - Estimativa de Frequência Fundamental - Oradores Justapostos

Conforme mencionado, a resposta da rede às séries de estimativas de tom não teve o peso esperado no resultado final. Apesar dos oradores B, C e D apresentarem valores bastante afastados do padrão, houve uma ocorrência de distância inferior ao padrão. Esse desempenho deve-se sobretudo ao fato de a frequência fundamental de um sinal de voz ser uma característica mais ligada ao modo como determinados fonemas são expressos de uma forma geral do que por um orador em particular. É uma grandeza bastante significativa em estudos ligados à prosódia e análise de pronúncias regionais,

mas aparece geralmente como coadjuvante em sistemas de identificação de voz. O presente estudo, pelo que foi visto, não constituiu exceção. As respostas às séries de tom contribuíram para a confirmação daquelas encontradas pela rede de energias associada. A Figura VI.5 mostra o perfil da curva decorrente das estimativas de período e, conseqüentemente da freqüência fundamental.

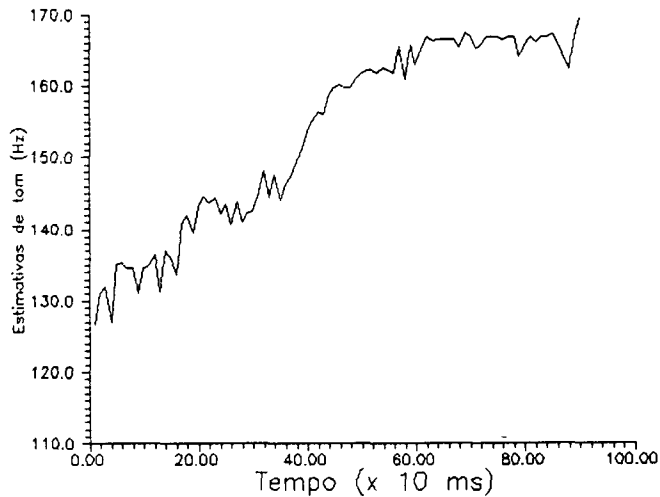


Figura VI.5 - Série de Estimativas de Freqüência Fundamental no Tempo

Tabela VI.3 - Resumo dos Resultados (Valores em p.u. do Padrão)

Orador	Energia	Tom	Total	Esperado
A	2,83	6,94	4,885	SIM
B	7,80	21,89	14,85	NÃO
C	8,42	20,89	14,65	NÃO
D	9,86	2,44	6,150	NÃO
E	9,67	23,10	16,38	NÃO

Finalmente, a média das distâncias apresentadas para cada orador são relacionadas ao diagnóstico esperado, ou seja, se o orador é o padrão ou não. Em uma população mais extensa, o valor final de distância deveria ser ponderado levando-se em consideração a distribuição estatística dos valores utilizados. Aqui, porém, a fim de mostrar-se a mecânica do processo de assimilação por parte do modelo conexionista, simplesmente as distâncias foram expressas em função da distância encontrada no reconhecimento do próprio padrão (erro de treinamento), para cada uma dos dois tipos de séries, e calculada sua média aritmética. Pode-se notar a coerência dos resultados. Na

verdade, das 18 séries de cada tipo apresentadas às redes, as 9 séries de energia não originadas pelo orador padrão encontram-se acima da média das demais, enquanto que, nas seqüências de estimativas de tom, apenas 2 encontram-se no mesmo nível daquelas do orador A.

O desempenho da rede neuronal, como pode ser verificado pelo valor dos erros relativos ao reconhecimento do próprio padrão treinado, é muito bom mesmo envolvendo séries temporais que apresentam uma auto-correlação relativamente baixa, como é o caso dos sinais de voz. Uma vez que se permaneça dentro do trecho empregado no aprendizado, pequenas variações na série de entrada levam a uma discrepância grande nos valores encontrados na camada de saída, variações essas que caem bruscamente à medida que os termos apresentados aproximam-se dos padrões da série aprendida. O modelo efetivamente assimila o sinal sonoro desejado.



## CAPÍTULO VII

### CONCLUSÕES E RECOMENDAÇÕES

A partir dos resultados obtidos nos testes realizados ao longo deste trabalho, algumas conclusões podem ser tiradas, no que diz respeito à utilização de uma abordagem conexionista do problema de reconhecimento de assinaturas sonoras, utilizando-se um modelo recorrente no tempo.

Em primeiro lugar, ainda que as técnicas convencionais de *backpropagation* apresentem um resultado satisfatório na memorização do padrão sonoro em si, o processo de reconhecimento envolve mais alguns detalhes que são mais eficientemente resolvidos por um processo que acrescente recorrência no tempo. Uma série de valores que varie a frequência com que um subconjunto de termos ocorre é melhor prevista utilizando-se recorrência temporal, já que certos grupos de padrões causam uma mudança específica no estado geral da rede, criando um contexto, uma ligação entre sucessões de valores. Assim, em vez de necessitar de um controle rígido sobre a ligação entre um tempo e um valor, o reconhecimento acontece também guiado por "gatilhos" ou "deixas", ocorrências na série que disparam um certo estado geral. Quando do processo de previsão em uma série temporal, essa capacidade é responsável pelo bom desempenho em séries que variam a frequência e amplitude de seus valores. No aproveitamento do modelo para uma atividade de identificação, essa característica confere à rede um perfil de memória associativa, isto é, de um sistema que representa um certo grupo de padrões através de outro, mais convenientemente armazenado.

Além disso, mesmo que as assinaturas sonoras possam, em alguns casos, apresentar um grau de coerência mínimo, ou seja, tornarem-se quase caóticas, a memorização de tais seqüências é bem executada pelo modelo. Os resultados dos treinamentos da rede para as séries representativas dos sinais de voz mostram uma taxa de erro pequena, à custa, entretanto, de sessões de treinamento bastante extensas. Esse treinamento contudo, consegue uma boa capacidade de associatividade mesmo em sinais ruidosos. Porém, se for desejada a absorção de seqüências longas como, por exemplo, de extensão da ordem de grandeza das ondas sonoras originais apresentadas no Capítulo V, torna-se imperiosa a necessidade de utilizar-se um *hardware* mais potente, estações de trabalho certamente, e mesmo assim deve-se esperar treinamentos que durem dias.

Os resultados obtidos mostraram uma capacidade de previsão superior ao modelo *backpropagation* convencional, mesmo com um período de treinamento limitado a 250.000 iterações. O erro de estimativa mantém-se menor que o melhor resultado não recorrente mesmo estendendo-se a previsão além do primeiro termo seguinte ao treinamento, até um máximo de 10 valores, sem necessidade de novo aprendizado.

Como consequência desta habilidade em assimilar uma estrutura básica de uma série de valores, a identificação de um orador conhecido pode ser facilmente realizada, já que o modelo apresenta índices de similaridade quatro vezes maiores ao ser confrontado com a mesma fonte de um sinal de voz aprendido.

Assinale-se uma vez mais o fato de não ser necessário saber com antecedência o comprimento total de uma série a ser analisada, o que é uma vantagem interessante sobre o método de retropropagação convencional. Para sucessões de pontos advindos de digitalizações realizadas por osciloscópios, ainda o método mais comum para obtenção de assinaturas industriais, a ordem de grandeza encontrada condiz com a das séries de energias e tons analisadas neste trabalho.

A continuação natural deste trabalho é finalmente conseguir-se a montagem da bancada em laboratório utilizando um dispositivo de geração de assinaturas mecânicas, de impacto. O processamento necessário para a análise desse tipo de sinal é o mesmo realizado aqui para as ondas de voz, ou seja, da digitalização em diante, o experimento não sofre alterações. O esperado é que o modelo comporte-se muito bem frente a esse tipo de assinaturas, uma vez que são mais coerentes e bem definidas. Talvez seja possível algum tipo de montagem em que, após as sessões de treinamento, as ondas de teste possam ser transmitidas diretamente à entrada da rede, causando uma identificação em tempo real. Esse tipo de montagem seria interessante para aplicações diferentes em que o processo de reconhecimento dar-se-ia, por exemplo, no campo e a obtenção da resposta em um tempo curto fosse um ponto importante.

Outra possibilidade seria a aplicação de uma segunda rede neuronal BP convencional para analisar os erros encontrados pela rede recorrente, adicionando um poder maior de generalização, além de constituir-se num método não-linear substituto ao cálculo da função distância entre a série esperada e a obtida. Assim, todo o processo de classificação da onda processada poderia ser realizada através de modelos conexionistas rápidos e com capacidade de aprendizado progressiva.

Se for desejado prosseguir examinando-se o desempenho no tocante aos sinais de voz mesmo, além de um ambiente de processamento mais poderoso, deveriam ser incluídas as análises das séries de LPC e de formantes da onda. Essa atitude poderia permitir o reconhecimento de fonemas e, mediante um treinamento de diversas séries de menor comprimento, em vez de uma só longa e complexa seqüência como foi implementado neste caso, ensinar à rede um pequeno vocabulário útil. Seria interessante também uma comparação com o desempenho utilizando-se séries no domínio da frequência, isto é, sucessões de valores provenientes de transformadas de Fourier. Essa análise de espectro combinada com a rede traria novas possibilidades de aplicações ao modelo.

Em resumo, a aplicação de um modelo classicamente relacionado como previsor, mas que apresenta características associativas, ao problema de identificação de padrões no tempo mostrou-se viável, contanto que se leve em consideração o tipo específico de padrão temporal em questão e que pague-se o preço de um treinamento exaustivo. Mesmo assim, a familiaridade dos algoritmos empregados e as vantagens de implementação credenciam essa abordagem a ser aproveitada em termos práticos e desenvolvida em futuras investigações.

## REFERÊNCIAS BIBLIOGRÁFICAS

- [1] - Lapedes, A.; Faber, R., *Non-Linear Signal Processing using Neural Networks : Prediction and System Modeling*, Los Alamos National Laboratory Report LA-UR-87-2662, 1987.
- [2] - Jones, W. P.; Hoskins, J., "Back-Propagation, A Generalized Delta Learning Rule", *BYTE Magazine*, Out 1987.
- [3] - Levine, D. S., *Introduction to Neural & Cognitive Modeling*, 2nd Ed, Lawrence Erlbaum Associates, New Jersey, 1990.
- [4] - McClelland, J. L.; Rumelhart, D. E., *Parallel Distributed Processing : Exploration in the Microstructure of Cognition*, MIT, Vol. 1, Foundations, Cambridge, Mass., 1988.
- [5] - Jacobs, R. A., "Increased Rates of Convergence Through Learning Rate Adaptation", *Neural Networks*, Vol. 1, pg. 295-307, 1988.
- [6] - Minai, A. A.; Williams, R. D., "Acceleration of Back-Propagation through Learning Rate and Momentum Adaptation", *International Joint Conference on Neural Networks*, Vol. 1, pg. 676-679, Jan 1990.
- [7] - Waibel, Alex, "Modular Construction of Time-Delay Neural Networks for Speech Recognition", *Neural Computation 1*, No 1, pg. 39-46, 1989.
- [8] - Hecht-Nielsen, Robert, *Neurocomputing*. Reading, Mass.: Addison-Wesley, 1990
- [9] - Pineda, Fernando J., *Generalizations of Backpropagation to Recurrent and Higher Order Neural Networks*, Neural Information Processing Systems, Denver, 1987.
- [11] - Hertz, J., Krogh, A., Palmer, R.G., *Introduction to the Theory of Neural Computation*, Addison-Wesley Publishing Company, Redwood City, 1991.
- [10] - Werbos, P. J., "Backpropagation Through Time: What it does and how to do it", *Proceedings of IEEE*, Vol 78, No. 10, pg. 1550-1560, Out. 1990.
- [12] - Hopfield, J. J., "Neural Networks and Physical Systems with Emergent Collective Computational Abilities", *Proceedings of the National Academy of Sciences*, Vol. 79, pg. 2554-2558, Abril 1982.
- [13] - Kosko, B., "Bidirectional Associative Memories", *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 18, No 1, pg. 49-60, Jan/Fev 1988.
- [14] - Raposo, Carlos M., *Redes Neurais na Previsão de Séries Temporais*, Dissertação de Mestrado, COPPE/UFRJ, Rio de Janeiro, 1992.
- [15] - Mabert, V. A., *An Introduction to Short Term Forecasting using the Box-Jenkins Methodology*, Purdue University, 1975.

- [16] - Rosemberg, A.E. e Sambur, M.R., "New Techniques for Automatic Speaker Verification", *IEEE Trans. Acoust., Speech, and Signal Proc.*, Vol. ASSP-23, pg. 169-176, Abr 1975.
- [17] - Rabiner, L.R., M.J. Cheng, A.E. Rosenberg, e C.A. McGonegal, "A Comparative Performance Study of Several Pitch Detection Algorithms", *IEEE Trans. Acoust., Speech, and Signal Proc.*, Vol. ASSP-24, No. 5, pg. 399-418, Out 1976.
- [18] - Rabiner, L.R., Schafer, R.W. *Digital Processing of Speech Signals*, Prentice-Hall Inc., New Jersey, 1978.
- [19] - Robinson, T. and Fallside, F., "A Recurrent Error Propagation Network Speech Recognition System", *Computer Speech and Language*, Vol. 5, pg. 259-274, 1991.
- [20] - Hunt, A., "Recurrent Neural Networks for Syllabification", *Speech Communication*, Vol. 13, No 3-4, pg. 323-332, Dez, 1993.
- [21] - Lippmann, R. P., "Review of Neural Networks for Speech Recognition", *Neural Computation*, No 1, pg. 1-38, 1989.
- [22] - Moore, Martin L., *The Ultimate Sound Blaster Book*, 1st Ed., Que Corp., Indianapolis, 1993.

## APÊNDICE A

### 1 - Programa de Conversão do Formato VOC para formato ASCII.

```

program conversor;

uses
  Crt;

const
  Ganho = 1.0;

type

  TipoTam = array [1..3] of byte;

  TipoHeader = record
    Id      : array [1..20] of byte;
    DadosPtr : integer;
    Versao  : array [1..2] of byte;
    Codigo  : array [1..2] of byte;
  end;

  HeaderDados = record
  {      Tipo : byte;   }
    Tam : TipoTam ;
  end;

  TipoSubBloco0 = byte;

  TipoSubBloco1 = record
    Header : HeaderDados;
    TC     : byte;
    Pack   : byte;
  {      Dados : byte;   }
  end;

  TipoSubBloco2 = record
    Header : HeaderDados;
  {      Dados : byte;   }
  end;

  TipoSubBloco3 = record
    Header : HeaderDados;
    Período : integer;
    TC     : byte;
  end;

  TipoSubBloco4 = record
    Header : HeaderDados;
  
```

```

    Marca : word;
end;
```

```

TipoSubBloco5 = record
    Header : HeaderDados;
{
    Texto : byte;
}
end;
```

```

TipoSubBloco6 = record
    Header : HeaderDados;
    Contador: integer;
end;
```

```

TipoSubBloco7 = record
    Header : HeaderDados;
    Contador: integer;
end;
```

```

TipoSubBloco8 = record
    Header : HeaderDados;
    TC : byte;
    Pack : byte;
    Modo : byte;
end;
```

```

var
    ArqIn : file;
    ArqOut : text;
    Cabeca : TipoHeader;
    NomeArq : string;
    SubBloco,
    p : byte;
    i, tempo : integer;
```

```

procedure TrataDados ( Tipo : byte );
```

```

var
    Dados1 : TipoSubBloco1;
    Dados2 : TipoSubBloco2;
    Tam : longint;
    Buffer : integer;
```

```

procedure CalculaTam (T1 : TipoTam; var T2 : longint);
begin
    T2 := 0;
    move (T1, T2, 3);
end;
```

```

begin
    case Tipo of
        1,9 : begin
            blockread (ArqIn, Dados1, sizeof (TipoSubBloco1));
            CalculaTam (Dados1.Header.Tam, Tam);
```

```

    dec (Tam, 2);
    for i := 1 to Tam div 2 do
        begin
            blockread (ArqIn, Buffer, 2);
            inc (tempo);
            writeln (ArqOut, tempo, ' ', Buffer);
        end;
    end;
2 : begin
    blockread (ArqIn, Dados2, sizeof (TipoSubBloco2));
    CalculaTam (Dados2.Header.Tam, Tam);
    for i := 1 to Tam div 2 do
        begin
            blockread (ArqIn, Buffer, 2);
            inc (tempo);
            writeln (ArqOut, tempo, ' ', Buffer);
        end;
    end;
end;
end;

procedure TrataSilencio;
begin
end;

procedure TrataMarca;
begin
end;

procedure TrataTexto;
begin
end;

procedure TrataLoop ( Tipo : byte );
begin
end;

procedure TrataExtensao;
begin
end;

begin
    ClrScr;
    gotoxy (1,5);
    NomeArq := ParamStr(1);
    p := pos ('.', NomeArq);
    if (p <> 0)
        then NomeArq := copy (NomeArq, 1, p);
    assign (ArqIn, NomeArq+'.VOC');
    assign (ArqOut, NomeArq+'.DAT');
    reset (ArqIn, 1);
    rewrite (ArqOut);

```



```

blockread (ArqIn, Cabeca, sizeof (TipoHeader));
with Cabeca do
begin
for i:= 1 to 19 do write (chr(Id[i]));
writeln;
writeln ('Versão ',versao[2],',',versao[1]);
end;
seek (ArqIn, Cabeca.DadosPtr);
blockread (ArqIn, SubBloco, 1);
tempo := 0;
while (SubBloco <> 0) do
begin
case SubBloco of
1, 2, 9 : TrataDados (SubBloco);
3 : TrataSilencio;
4 : TrataMarca;
5 : TrataTexto;
6, 7 : TrataLoop (SubBloco);
8 : TrataExtensao;
end;
blockread (ArqIn, SubBloco, 1);
end;
close (ArqIn);
close (ArqOut);
end.

```

## 2 - Programa de Detecção de Voz/Não Voz

```

program bordas;

```

```

{*****}
{

```

Este programa determina os pontos de início e fim do sinal de voz, isto é, a passagem do silêncio para a fala e vice-versa.

```

}
{*****}

```

```

uses

```

```

  Crt;

```

```

const

```

```

  TFixo = 25;

```

```

  TMax = 100;

```

```

  MaxImp = 300;

```

```

type

```

```
PtrPulso = ^TipoPulso;
```

```
TipoPulso = record
    t : integer;
    X, Z : real;
    Ant,
    Prox : PtrPulso;
end;
```

```
var
```

```
X, Y : array [1..MaxImp] of real;
N1, N2, NCand,
i, j, NumImp,
m, n, Tic : integer;
Silencio,
p1, p3 : PtrPulso;
MaxEn,
X1, X2,
I1, I2,
T1, T2, T3,
En, ESalva,
Z, ZSalva,
ZAux, EnAux,
ME, SE,
MZ, SZ, tempo,
IZC : real;
ArqDef,
ArqEGraf,
ArqGraf2,
Arq, Arq2 : text;
ArqMed,
ArqOut : file;
ch : char;
```

```
function Sinal (X : real) : integer;
begin
    if X>=0
    then Sinal := 1
    else Sinal := -1;
end;
```

```
function Jan (n : integer) : real;
begin
    if (N>=0) and (N<=TMax)
    then Jan := 1 / (2*TMax)
    else Jan := 0;
end;
```

```
procedure Media (p : PtrPulso; var ME, MZ : real);
var
    i : integer;
begin
```

```

ME := 0;
MZ := 0;
i := 0;
while p <> nil do
  begin
    inc (i);
    MZ := MZ + p^.Z;
    ME := ME + p^.X;
    p := p^.prox;
  end;
ME := ME / (i+1);
MZ := MZ / (i+1);
end;

procedure DesvioPadrao (p : PtrPulso; var SE, SZ : real);
var
  i : integer;
  SQE, SQZ : real;
begin
  i := 0;
  SQE := 0;
  SQZ := 0;
  while p <> nil do
    begin
      inc (i);
      SQE := SQE + sqr(p^.X);
      SQZ := SQZ + sqr(p^.Z);
      p := p^.prox;
    end;

    SE := (SQE - (i+1)*sqr(ME)) / (i+1);
    SE := sqrt (SE);
    SZ := (SQZ - (i+1)*sqr(MZ)) / (i+1);
    SZ := sqrt (SZ);
  end;

begin

assign (Arq, ParamStr(1)+'DAT');
reset (Arq);
assign (ArqMed, copy (ParamStr(1), 1, 5)+'EN.IN');
rewrite (ArqMed, sizeof (real));
assign (ArqDef, copy (ParamStr(1), 1, 5)+'EN1.DEF');
rewrite (ArqDef);
assign (ArqEGraf, copy (ParamStr(1), 1, 6)+'EN.DAT');
rewrite (ArqEGraf);
assign (ArqGraf2, copy (ParamStr(1), 1, 6)+'ENG.DAT');
rewrite (ArqGraf2);

getmem (Silencio, sizeof (TipoPulso));

gotoxy (5, 10);

```

```

writeln ('Lidos    tempos');
gotoxy (5, 12);
writeln ('Gerados    impulsos');

m := 0; n := TMax; Tic := 0;
En := 0; Z := 0;
ESalva := 0; ZSalva := 0;
MaxEn := 0; NumImp := 0;

readln (Arq, tempo, X1);
p1 := nil;
p3 := Silencio;

while not EOF (Arq) do
begin
  readln (Arq, tempo, X2);
  inc (m); inc (Tic);
  gotoxy (11, 10);
  write (Tic:5);

  EnAux := abs (X1);
  En := En + EnAux;
  ESalva := ESalva+EnAux;

  ZAux := abs (Sinal(X2)-Sinal(X1)) * Jan(n-m);
  Z := Z + ZAux;
  ZSalva := ZSalva+ZAux;

  if Tic <= 500
  then
  begin
    if p1 <> nil
    then p1^.prox := p3;
    p3^.Ant := p1;
    p1 := p3;
    p1^.Z := ZSalva;
    p1^.X := ESalva;
    p1^.t := Tic;
    getmem (p3, sizeof (TipoPulso));
    p1^.prox := nil;
  end;

  if Tic = 500
  then
  begin
    freemem (p3, sizeof(TipoPulso));
    Media (Silencio, ME, MZ);
    DesvioPadrao (Silencio, SE, SZ);
    p1 := Silencio;
    while p1 <> nil do
    begin
      p3 := p1^.prox;
      freemem (p1, sizeof(TipoPulso));
    end;
  end;
end;

```

```

        p1 := p3;
    end;
end;
if m = n
then
begin
    inc (NumImp);
    X[NumImp] := En;
    Y[NumImp] := Z;

    if En > MaxEn then MaxEn := En;

    writeln (ArqEGraf, NumImp, ' ',En);

    Z := 0; En := 0; m := 0;
    gotoxy (13, 12);
    write (Tic div n);
end;
X1 := X2;
end;

close (ArqEGraf);

{ Thresholds de energia }
{ ME = a media dos primeiros 100 ms}

I1 := 0.0005 * (MaxEn-ME) + ME;
I2 := 4 * ME;

if I1 <= I2
then T2 := I1
else T2 := I2;

T1 := 4.0 * T2;

{ Threshold de cruzamento do zero no silencio}

IZC := MZ + 2 * SZ;
if (TFixo <= IZC)
then T3 := TFixo
else T3 := IZC;

m := 0;
repeat
    inc (m);
until (X[m] > T1) or (m = NumImp);
N1 := m;

repeat

```

```

    inc (m);
until (X[m] < T1) or (m = NumImp);
N2 := m;

```

```

m := N1;
repeat
    dec (m);
until (X[m] < T2) or (m = 0);
N1 := m;

```

```

m := N2;
repeat
    inc (m);
until (X[m] < T2) or (m = NumImp);
N2 := m;

```

```

m := N1;
j := 0;
i := 0;

```

```

repeat
    inc (j);
    dec(m);
    if Y[m] < T3
        then
            begin
                if i = 0
                    then NCand := m;
                inc (i);
            end;
until (j = 25) or (i > 3);

```

```

if (i = 3)
    then N1 := NCand;

```

```

m := N2;
j := 0;
i := 0;

```

```

repeat
    inc (j);
    inc (m);
    if Y[m] < T3
        then
            begin
                if i = 0
                    then NCand := m;
                inc (i);
            end;
until (j = 25) or (i > 3);

```

```

if (i = 3)
    then N2 := NCand;

```

```

if N1 < 0 then N1 := 0;
if N2 > NumImp then N2 := NumImp;

close (Arq);

gotoxy (5, 14);
writeln (N1, ' <---> ', N2);
if ((N2 - N1 + 1) mod 5) = 0
  then inc (N2);

for m:= N1 to N2 do
  begin
    blockwrite (ArqMed, X[m], 1);
    gotoxy (5, 15);
    writeln (ArqGraf2, m-N1+1, ' ', X[m]);
    write (m-N1+1);
  end;
close (ArqMed);
close (ArqGraf2);

writeln (ArqDef, 3);
writeln (ArqDef, 5);
writeln (ArqDef, 2);
writeln (ArqDef, 1);
writeln (ArqDef, N2-N1+1);
writeln (ArqDef, 1);

close (ArqDef);

assign (ArqOut, ParamStr(1)+'PIT');
assign (Arq2, copy (ParamStr(1),1,7)+'G.DAT');
rewrite (ArqOut, sizeof (real));
rewrite (Arq2);
reset (Arq);
for m := 0 to (N1-1)*100 do readln (Arq);

while (m < (N2+1)*100) and (not Eof (Arq)) do
  begin
    readln (Arq, tempo, X1);
    blockwrite (ArqOut, X1, 1);
    writeln (Arq2, tempo, ' ', X1);
    inc (m);
  end;

close (Arq);
close (Arq2);
close (ArqOut);

end.

```

### 3 - Programa para Análise de Frequência Fundamental

```

program Pitch;

{*****}
{
  Este programa analisa uma sērie de sinais no tempo e gera, a cada 10 ms
  uma estimativa da frequencia fundamental da onda.

}
{*****}

uses
  Crt;

const
  TMax = 100;

type
  PtrPulso = ^TipoPulso;

  TipoPulso = record
    t, X : real;
    Prox : PtrPulso;
  end;

var
  X      : array [1..TMax] of real;
  Trem,
  Lista  : array [1..6] of PtrPulso;
  Picos,
  Vales,
  p, pt,
  paux   : PtrPulso;
  Perodo : array [1..6] of real;
  PerodoAnt : array [1..6, 1..2] of real;
  PM, PMAnt : array [1..6] of real;
  PerodoEst,
  Soma,
  Taxa,
  Intervalo,
  Saida   : real;
  TotalPulsos : longint;
  TT,
  NumPulsos : integer;
  k, n     : byte;
  ArqPicos,
  ArqOut,
  ArqAux,
  ArqVales : text;

```



```

ArqPit,
ArqIn   : file;
ListaArq : array [1..6] of text;
Crescendo : boolean;
NS      : string[1];
NomeArquivo : string[40];
ch      : char;

```

```

procedure Inclui ( var p : PtrPulso; T ,X : real);
var
  p2, paux : PtrPulso;
begin
  paux := p;
  p2 := p;
  while (p2 <> nil) do
    begin
      paux := p2;
      p2 := p2^.prox;
    end;

  if (p = nil)
  then
    begin
      new(p);
      p2 := p;
      paux := nil;
    end
  else
    new (p2);

  if (paux <> nil)
  then paux^.Prox := p2;

  p2^.Prox := nil;
  p2^.X := X;
  p2^.T := T;
end;

```

```

procedure IncluiLista (N : byte; p1, p2 : PtrPulso);
var
  Amp : real;
begin
  case N of
    1 : Amp := p1^.X;
    2 : Amp := p1^.X - p2^.X;
    3 : begin
        Amp := p1^.X - p2^.X;
        if Amp < 0 then Amp := 0;
      end;
    4 : Amp := -p1^.X;
    5 : Amp := -p1^.X + p2^.X;
    6 : begin

```

```

    Amp := -p1^.X + p2^.X;
    if Amp < 0 then Amp := 0;
  end;
end;
Inclui (Lista[N], p1^.T, Amp);
writeln (ListaArq[N], p1^.T, Amp);
end;

procedure GeralListas;
var
  paux,
  p1, p2 : PtrPulso;
  tpo, i : integer;

begin
  Crescendo := true;
  for tpo := 2 to TMax do
    begin
      if (Crescendo) and (X[tpo] < X[tpo-1])
      then
        begin
          Inclui (Picos, tpo-1, X[tpo-1]);
          writeln (ArqPicos, (tpo-1):6, ' ', X[tpo-1]:8:3);
          Crescendo := false;
        end
      else if (not Crescendo) and (X[tpo] > X[tpo-1])
      then
        begin
          Inclui (Vales, tpo-1, X[tpo-1]);
          writeln (ArqVales, (tpo-1):6, ' ', X[tpo-1]:8:3);
          Crescendo := true;
        end;
    end;

  for i := 1 to 6 do Lista[i] := nil;

  p1 := Picos;

  while p1 <> nil do
    begin
      IncluiLista (1, p1, nil);
      p1 := p1^.Prox;
    end;

  p1 := Picos;
  p2 := Vales;

  while (p1 <> nil) do
    begin
      while (p2^.t < p1^.t) do
        begin
          paux := p2;
          p2 := p2^.Prox;

```

```

    end;
    IncluirLista (2, p1, p2);
    p1 := p1^.Prox;
end;

p1 := Picos;
p2 := p1;
p1 := p1^.Prox;

while (p1 <> nil) do
begin
    IncluirLista (3, p1, p2);
    p2 := p1;
    p1 := p1^.Prox;
end;

p1 := Vales;

while p1 <> nil do
begin
    IncluirLista (4, p1, nil);
    p1 := p1^.Prox;
end;

p1 := Vales;
p2 := Picos;

while (p1 <> nil) do
begin
    while (p2^.t < p1^.t) do
begin
    p aux := p2;
    p2 := p2^.Prox;
end;
    IncluirLista (5, p1, p2);
    p1 := p1^.Prox;
end;

p1 := Vales;
p2 := p1;
p1 := p1^.Prox;

while (p1 <> nil) do
begin
    IncluirLista (6, p1, p2);
    p2 := p1;
    p1 := p1^.Prox;
end;

end;

function MaisComum : real;

```

```

var
i, j, C, L : integer;
Bias      : byte;
MaxCo     : integer;
NumCo     : array [1..6] of integer;
Coincidencias : array [1..4] of integer;
Matriz    : array [1..6, 1..6] of real;
begin
writeln (' (' ,n:2,')');
for i:=1 to 6 do
begin
Matriz[1,i] := Periodo[i];
Matriz[2,i] := PeriodoAnt[i, 1];
Matriz[3,i] := PeriodoAnt[i, 2];
Matriz[4,i] := Matriz[1,i]+Matriz[2,i];
Matriz[5,i] := Matriz[2,i]+Matriz[3,i];
Matriz[6,i] := Matriz[1,i]+Matriz[2,i]+Matriz[3,i];
end;

for i:= 1 to 6 do
begin
if (Periodo[i] >= 16) and (Periodo[i] < 31)
then j := 1
else if (Periodo[i] >= 31) and (Periodo[i] < 63)
then j := 2
else if (Periodo[i] >= 63) and (Periodo[i] < 127)
then j := 4
else if (Periodo[i] >= 127) and (Periodo[i] < 255)
then j := 8;

for Bias := 1 to 4 do
begin
case Bias of
1 : begin
Coincidencias[1] := 0;
for C := 1 to 6 do
for L := 1 to 6 do
begin
if (L <> 1) or (C <> i)
then if (abs (Periodo[i] - Matriz[C, L]) < (j * Bias))
then
begin
inc (Coincidencias[1]);
end;
end;
dec (Coincidencias[1], 1);
end;
2 : begin
Coincidencias[2] := 0;
for C := 1 to 6 do
for L := 1 to 6 do

```

```

begin
  if (L <> 1) or (C <> i)
    then if (abs (Periodo[i] - Matriz[C, L]) < (j * Bias))
      then
        begin
          inc (Coincidencias[2]);
        end;
      end;
    dec (Coincidencias[2], 2);
  end;
3 : begin
  Coincidencias[3] := 0;
  for C := 1 to 6 do
    for L := 1 to 6 do
      begin
        if (L <> 1) or (C <> i)
          then if (abs (Periodo[i] - Matriz[C, L]) < (j * Bias))
            then
              begin
                inc (Coincidencias[3]);
              end;
            end;
          end;
        dec (Coincidencias[3], 5);
      end;
4 : begin
  Coincidencias[4] := 0;
  for C := 1 to 6 do
    for L := 1 to 6 do
      begin
        if (L <> 1) or (C <> i)
          then if (abs (Periodo[i] - Matriz[C, L]) < (j * Bias))
            then
              begin
                inc (Coincidencias[4]);
              end;
            end;
          end;
        dec (Coincidencias[4], 7);
      end;
    end;
  end;
end;
end;

MaxCo := 0;
for j := 1 to 4 do
  if Coincidencias[j] > MaxCo
    then MaxCo := Coincidencias[j];
Numco[k] := MaxCo;
end;

MaxCo := 0; j := 1;
for i:= 1 to 6 do
  if NumCo[i] >= MaxCo
    then
      begin

```

```

        MaxCo := NumCo[i];
        j := i;
    end;

    MaisComum := Perodo[j];
end;

begin
    ClrScr;
    gotoxy (5, 16);
    write ('Lidos    pulsos no frame');
    NomeArquivo := ParamStr(1)+'PIT';
    assign (ArqIn, NomeArquivo);
    reset (ArqIn, sizeof(real));
    assign (ArqPicos,'PICOS.DAT');
    assign (ArqVales,'VALES.DAT');
    assign (ArqAux, copy (ParamStr(1),1, 5)+'PIG.DAT');
    assign (ArqOut, copy (ParamStr(1),1, 5)+'PI2.DEF');
    assign (ArqPit, copy (ParamStr(1),1, 6)+'PI.IN');
    rewrite (ArqPicos);
    rewrite (ArqVales);
    rewrite (ArqAux);
    rewrite (ArqOut);
    rewrite (ArqPit, sizeof(real));
    for n := 1 to 6 do
        begin
            str (n, ns);
            assign (ListaArq[n], 'LISTA'+NS+'.DAT');
            rewrite (ListaArq[n]);
        end;

    TotalPulsos := filesize (ArqIn);

    for k := 1 to 6 do
        begin
            PerodoAnt[k,1] := 0;
            PerodoAnt[k,2] := 0;
        end;

    for n := 1 to (TotalPulsos div TMax ) do
        begin
            for NumPulsos := 1 to TMax do
                begin
                    blockread (ArqIn, X[NumPulsos], 1);
                    gotoxy (11, 16);
                    write (NumPulsos:3);
                    gotoxy (32, 16);
                    write (n:3);
                end;
            end;

            Picos := nil;
            Vales := nil;
        end;
    end;
end;

```

GeraListas;

TT := 0;

Saida := Picos<sup>X</sup>;

for k := 1 to 6 do

begin

Periodo[k] := 1;

PMAnt[k] := 1;

end;

for k:= 1 to 6 do

begin

Trem[k] := nil;

p := Lista[k];

PM[k] := (PMAnt[k] + PeriodoAnt[k,1]) / 2;

PMAnt[k] := PM[k];

while p <> nil do

begin

if TT = 0

then

begin

if p<sup>X</sup> > Saida

then

begin

Saida := p<sup>X</sup>;

Intervalo := 0.4 \* PM[k];

TT := round (Intervalo);

Inclui (Trem[k], p<sup>t</sup>, Saida);

end

else

begin

Taxa := PM[k] / 0.695;

Saida := Saida \* exp (-Taxa);

end

end

else dec (TT);

paux := p;

p := p<sup>Prox</sup>;

dispose (paux);

end;

NumPulsos := 0;

p := Trem[k];

paux := p;

if Paux <> nil then p := p<sup>Prox</sup>;

Soma := 0;

while (p <> nil) do

begin

Soma := Soma + (p<sup>t</sup> - paux<sup>t</sup>);

paux := p;

p := p<sup>Prox</sup>;

```

        inc (NumPulsos);
    end;
    if NumPulsos > 0
    then Período[k] := Soma / NumPulsos
    else Período[k] := PeríodoAnt[k,1];

    if (n > 2)
    then Período[k] := (Período[k]
        + PeríodoAnt[k,1]
        + PeríodoAnt[k,2]) / 3;

    PeríodoAnt[k,2] := PeríodoAnt[k,1];
    PeríodoAnt[k,1] := Período[k];
end;

if n > 2
then
begin
    PeríodoEst := MaisComum ;

    gotoxy (5,20);
    ClrEol;
    write (n , '--> ',(PeríodoEst*0.1):8:3, ' ms');
    blockwrite (ArqPit, PeríodoEst, 1);
    writeln (ArqAux, n-2, ' ',1/PeríodoEst);
    if PeríodoEst <> 0
    then
    begin
        gotoxy (5,21);
        ClrEol;
        write ((10 / PeríodoEst):8:3, ' kHz');
        end;
    end;

end;

end;

writeln;
writeln (ArqOut, 3);
writeln (ArqOut, 5);
writeln (ArqOut, 2);
writeln (ArqOut, 1);
writeln (ArqOut, n-2);
writeln (ArqOut, 1);

close (ArqIn);
close (ArqPit);
close (ArqPicos);
close (ArqVales);
close (ArqAux);
close (ArqOut);
for n := 1 to 6 do close (ListaArq[n]);

```



end.

#### 4 - Simulação de Redes Neurais com Retropropagação no Tempo

```
program tese;
```

```
{ $F+ }
```

```
{*****}
{
```

Este programa simula uma rede neuronal backpropagation recorrente com um nível de recorrência no tempo.

```
}
{*****}
```

```
uses
```

```
Dos,
```

```
Crt;
```

```
label Executa, Fim;
```

```
const
```

```
TamMaxSeq = 50;
```

```
NumMaxSeq = 1;
```

```
NumMaxNeuronios = 5;
```

```
NumMaxCamadas = 3;
```

```
ErroMaximo = 0.1;
```

```
MaxIteracoes = 750000;
```

```
FreqSalva = 1000;
```

```
FreqSalvaMin = 0;
```

```
ContMin : longint = 0;
```

```
ContMin2 : longint = 0;
```

```
ValorReset = 0.25;
```

```
Teta = 0.3;
```

```
Lambda = 3.0;
```

```
AlfaMax = 2.0;
```

```
MiMax = 0.9;
```

```
Beta = 1.0;
```

```
NumInicio = 2;
```

```
NumFim = 2;
```

```
TamSerie = 156;
```

```
type
```

```
TipoVetorPesos = array [1..NumMaxNeuronios,1..NumMaxNeuronios] of real;
```

```
MatrizDeNiveis = array [0..TamMaxSeq, 1..NumMaxNeuronios] of real;
```

```

TipoCamada = record
    NumNeuronios : byte;
    X, Y, Erro : MatrizDeNiveis;
    Alfa, Mi,
    AlfaRec, MiRec,
    Delta, DAnt,
    DeltaAnt,
    DeltaRec, DRecAnt,
    DeltaAntRec,
    DBarra,
    DBarraRec,
    W, WR : TipoVetorPesos;
end;

TipoRede = array [1..NumMaxCamadas] of TipoCamada;

TipoInput = record
    X,
    XDesejado : MatrizDeNiveis;
end;

TipoSinapses = array [2..NumMaxCamadas] of TipoVetorPesos;

FuncaoDeTransferencia = function (l : real) : real;

TipoMemErros = array [1..TamMaxSeq] of real;
var
    Camada : TipoRede;
    Input : array [1..NumMaxSeq] of TipoInput;
    Sinapse,
    SinapseRec,
    SW, SWR : TipoSinapses;
    YFinal : array [1..NumMaxSeq, 1..TamMaxSeq] of real;
    YDesejado : MatrizDeNiveis;

{ Arquivos de entrada e saída }
    ArqSin,
    ArqTmp,
    ArqPnd,
    ArqIn : file;
    Arq2,
    Arq : text;

    FTransf,
    DFTransfDX : FuncaoDeTransferencia;

    YD, E,
    EPondMinimo,
    EMinimo : real;
    XAux : array [1..5] of real;
    EMin,
    EAnt : TipoMemErros ;

```

```

Atualiza   : array [1..TamMaxSeq] of boolean;
NumArqStr  : string[2];
NomeArq    : string[80];

```

```

Capa, Fi,
Gama      : real;

```

```

XMax, XMin,
Escala,
OffSet    : real;
Contador  : longint;
NumArq,
NumCamadas,
NumSeq,
TamSeq, t,
TamseqTotal,
Salvamentos,
i, j, k, S : integer;
ch        : char;

```

```

Ho,Min,Se,Hu : word;

```

```

function Sigmoide ( l : real ) : real;
begin
  Sigmoide := 1 / ( 1 + exp ( -{Beta}*l) );
end;

```

```

function DSigmoideDX ( l : real ) : real;
begin
  DSigmoideDX := Sigmoide (l) * ( 1 - Sigmoide (l) );
end;

```

```

function TangHiper ( l : real ) : real;
var
  Aux1, Aux2 : real;
begin
  Aux1 := (exp (l) - exp (-l));
  Aux2 := (exp (l) + exp (-l));
  TangHiper := Aux1 / Aux2 ;
end;

```

```

function DTangHiperDX ( l : real ) : real;
begin
  DTangHiperDX := 1 - sqr (TangHiper (l));
end;

```

```

function Normaliza ( X : real ) : real;
begin
  Normaliza := X * Escala + OffSet;
end;

```

```

function Denormaliza ( X : real ) : real;
begin
  Denormaliza := (X - OffSet) / Escala ;
end;

procedure ProcessaRede;
var
  Soma1, Soma2 : real;
  i, j, k      : byte;
begin
  for k := 2 to NumCamadas do
    with Camada[k] do
      begin
        for j := 1 to NumNeuronios do
          begin
            Soma1 := 0; Soma2 := 0;
            for i := 1 to Camada[k-1].NumNeuronios do
              begin
                Soma1 := Soma1 + Sinapse[k,j,i] * Camada[k-1].Y[t,i];
              end;
            for i := 1 to NumNeuronios do
              begin
                Soma2 := Soma2 + SinapseRec[k,j,i] * Y[t-1,i];
              end;
            X[t,j] := Soma1 + Soma2;
            Y[t,j] := FTtransf (X[t,j]);
          end;
        end;
      end;
    end;

  procedure CalculaErros;
  var
    Soma1, Soma2,
    Somatorio,
    Distancia    : extended;
    i, j, l      : byte;
  begin
    if t = TamSeq
    then
      begin
        with Camada[3] do
          begin
            for j:= 1 to NumNeuronios do
              begin
                Distancia := YDesejado[TamSeq,j] - Y[TamSeq,j];
                Erro[TamSeq, j] := Distancia * DFTransfDX (X[TamSeq, j]);
              end;
            if (abs(Erro[TamSeq, j]) < abs((EMin[TamSeq] * Lambda)))
            then
              begin
                EMin[TamSeq] := Erro[TamSeq, j];
                Atualiza[TamSeq] := true;
              end
            end;
          end;
        end;
      end;
    end;
  end;

```

```

else
    Atualiza[TamSeq] := false;
end;
with Camada[2] do
begin
    for j := 1 to NumNeuronios do
        begin
            Soma1 := 0;
            for i:= 1 to Camada[3].NumNeuronios do
                Soma1 := Soma1 + Sinapse[3,i,j] * Camada[3].Erro[TamSeq, i];
                Erro[TamSeq, j] := Soma1 * DFTransfDx (X[TamSeq, j]);
            end;
        end;
    end;
end
else
begin
with Camada[3] do
begin
    for j := 1 to NumNeuronios do
        begin
            Somatorio := 0; Soma1 := 0;
            Distancia := YDesejado[t,j] - Y[t,j];
            for i := 1 to NumNeuronios do
                Somatorio := Somatorio + SinapseRec[3,j,i] * Erro[t+1,i];
                Soma1 := Distancia * DFTransfDX (X[t,j]) + Somatorio;
                Erro[t, j] := Soma1 * DFTransfDX (X[t,j]);
                if (abs(Erro[t, j]) < abs((EMin[t] * Lambda)))
                    then
                        begin
                            EMin[t] := Erro[t, j];
                            Atualiza[t] := true;
                        end
                    else
                        Atualiza[t] := false;
                    end;
            end;
        end;
    end;
with Camada[2] do
begin
    for j := 1 to NumNeuronios do
        begin
            Somatorio := 0; Soma1 := 0; Soma2 := 0;
            for i := 1 to Camada[3].NumNeuronios do
                Soma1 := Soma1 + Sinapse[3,i,j] * Camada[3].Erro[t,i];
            for i := 1 to NumNeuronios do
                Somatorio := Somatorio + SinapseRec[2,j,i] * Erro[t+1,i];
                Soma2 := Soma1 + Somatorio;
                Erro[t, j] := Soma2 * DFTransfDX (X[t,j]);
            end;
        end;
    end;
end;
end;
end;

```

```
{
```

```

procedure AlteraPesos;
var
  DeltaW : real;
  i, j, k : byte;
begin
  for k := 2 to NumCamadas do
    with Camada[k] do
      begin
        for j:= 1 to NumNeuronios do
          for i := 1 to Camada[k-1].NumNeuronios do
            begin
              DeltaW := Beta*Erro[t,j]*Camada[k-1].Y[t,i] ;
              W[j,i] := W[j,i] + DeltaW + Momento * DeltaWAntDireta;
              DeltaWAntDiKhreta := DeltaW + Momento * DeltaWAntDireta;
            end;
          for j:= 1 to NumNeuronios do
            for i := 1 to NumNeuronios do
              begin
                DeltaW := Beta*Erro[t,j]*Y[t-1,i];
                WR[j,i] := WR[j,i] + DeltaW + Momento * DeltaWAntRecursiva;
                DeltaWAntRecursiva := DeltaW + Momento * DeltaWAntRecursiva;
              end;
            end;
          end;
        end;
      end;
    end;
  }

```

```

procedure AlteraPesos;
var
  DBarraAnt,
  D, DRec : real;
  i, j, k : byte;

```

```

function DBD (D, A : real) : real;
begin
  if (D > 0)
  then DBD := Capa
  else if (D < 0)
  then DBD := - Fi * A
  else DBD := 0;
end;

```

```

function EDBD (D1, D2, D3, A : real; Tipo : byte) : real;
var
  Prod : real;
begin
  if Tipo = 0
  then
    begin
      Capa := 0.2;
      Gama := 0.01;
      Fi := 0.2;
    end

```

```

else
  begin
    Capa := 0.2;
    Gama := 0.01;
    Fi := 0.2;
  end;

Prod := D1 * D3;

if (Prod > 0)
  then EDBD := Capa * exp(-Gama*abs(D2))
  else if (Prod < 0)
    then EDBD := - Fi * A
    else EDBD := 0;
end;

begin
  for k := 2 to NumCamadas do
    with Camada[k] do
      begin
        for j:= 1 to NumNeuronios do
          for i := 1 to Camada[k-1].NumNeuronios do
            begin
              Delta[j,i] := Erro[t,i]*Camada[k-1].Y[t,i] ;
              DBarraAnt := DBarra[j,i];
              DBarra[j,i] := (1-Teta)*Delta[j,i]+Teta*DeltaAnt[j,i];
              Alfa[j,i] := Alfa[j,i] + EDBD (Delta[j,i], DBarra[j,i],
                DBarraAnt, Alfa[j,i], 0);
              if Alfa[j,i] > AlfaMax
                then Alfa[j,i] := AlfaMax;
              Mi[j,i] := Mi[j,i] + EDBD (Delta[j,i],DBarra[j,i],
                DBarraAnt, Mi[j,i], 1);
              if Mi[j,i] > MiMax
                then Mi[j,i] := MiMax;
              if (Atualiza[t])
                then
                  begin
                    D := Alfa[j,i]*Delta[j,i] + Mi[j,i]*DAnt[j,i];
                    W[j,i] := W[j,i] + D;
                    DAnt[j,i] := D;
                  end;
            {
              else W[j,i] := 0;
              else W[j,i] := W[j,i] * (1+((random (1) / 5) - 1/10));}
              DeltaAnt[j,i] := Delta[j,i];
            end;

          for i:= 1 to NumNeuronios do
            for j := 1 to NumNeuronios do
              begin
                DeltaRec[j,i] := Erro[t,i]*Y[t-1,i] ;
                DBarraAnt := DBarraRec[j,i];
                DBarraRec[j,i] := (1-Teta)*DeltaRec[j,i]+Teta*DeltaAntRec[j,i];
              end;
            end;
          end;
        end;
      end;
    end;
  end;

```

```

        AlfaRec[j,i] := AlfaRec[j,i] + EDBD (DeltaRec[j,i], DBarraRec[j,i],
            DBarraAnt, AlfaRec[j,i], 0);
    if AlfaRec[j,i] > AlfaMax
        then AlfaRec[j,i] := AlfaMax;
    MiRec[j,i] := MiRec[j,i] + EDBD (DeltaRec[j,i],DBarraRec[j,i],
        DBarraAnt, MiRec[j,i], 1);
    if MiRec[j,i] > MiMax
        then MiRec[j,i] := MiMax;
    if (Atualiza[t]) and ( t > 0)
        then
            begin
                DRec := AlfaRec[j,i]*DeltaRec[j,i] + MiRec[j,i]*DRecAnt[j,i];
                WR[j,i] := WR[j,i] + DRec;
                DRecAnt[j,i] := DRec;
            end;
    {
        else WR[j,i] := 0;
        else WR[j,i] := WR[j,i] * (1+((random (1) / 5) - 1/10));}
    DeltaAntRec[j,i] := DeltaRec[j,i];
    end;
end;
end;

```

```

procedure AtualizaPesos;

```

```

var

```

```

    i, j, k : byte;

```

```

begin

```

```

    for k := 2 to NumCamadas do

```

```

        with Camada[k] do

```

```

            begin

```

```

                for j:= 1 to NumNeuronios do

```

```

                    for i := 1 to Camada[k-1].NumNeuronios do

```

```

                        begin

```

```

                            SW[k,j,i] := SW[k,j,i] + W[j,i];

```

```

                        end;

```

```

                    for j:= 1 to NumNeuronios do

```

```

                        for i := 1 to NumNeuronios do

```

```

                            begin

```

```

                                SWR[k,j,i] := SWR[k,j,i] + WR[j,i];

```

```

                            end;

```

```

                        end;

```

```

                    end;

```

```

procedure AlteraSinapses ;

```

```

var

```

```

    i, j, k : byte;

```

```

begin

```

```

    for k := 2 to NumCamadas do

```

```

        with Camada[k] do

```

```

            begin

```

```

                for i:= 1 to NumNeuronios do

```

```

                    for j := 1 to Camada[k-1].NumNeuronios do

```

```

                        begin

```

```

                            Sinapse[k,i,j] := Sinapse[k,i,j] + SW[k,i,j];

```



```

    end;
  for i:= 1 to NumNeuronios do
    for j:= 1 to NumNeuronios do
      begin
        SinapseRec[k,i,j] := SinapseRec[k,i,j] + SWR[k,i,j];
      end;
    end;
  end;
end;

procedure SalvaSinapses (Tipo : byte);
begin
  case Tipo of

    0 : begin
      rewrite (ArqSin, 1);
      blockwrite (ArqSin, Contador,  sizeof (longint));
      blockwrite (ArqSin, Sinapse,  sizeof (TipoSinapses));
      blockwrite (ArqSin, SinapseRec, sizeof (TipoSinapses));
      blockwrite (ArqSin, Camada,  sizeof (TipoRede));
      blockwrite (ArqSin, EAnt,  sizeof (TipoMemErros));
      blockwrite (ArqSin, EMin,  sizeof (TipoMemErros));
      blockwrite (ArqSin, EMinimo,  sizeof (real));
      close (ArqSin);
    end;

    1 : begin
      rewrite (ArqTmp, 1);
      blockwrite (ArqTmp, Contador,  sizeof (longint));
      blockwrite (ArqTmp, Sinapse,  sizeof (TipoSinapses));
      blockwrite (ArqTmp, SinapseRec, sizeof (TipoSinapses));
      blockwrite (ArqTmp, Camada,  sizeof (TipoRede));
      blockwrite (ArqTmp, EAnt,  sizeof (TipoMemErros));
      blockwrite (ArqTmp, EMin,  sizeof (TipoMemErros));
      blockwrite (ArqTmp, EMinimo,  sizeof (real));
      close (ArqTmp);
    end;

    2 : begin
      rewrite (ArqPnd, 1);
      blockwrite (Arqpnd, Contador,  sizeof (longint));
      blockwrite (Arqpnd, Sinapse,  sizeof (TipoSinapses));
      blockwrite (Arqpnd, SinapseRec, sizeof (TipoSinapses));
      blockwrite (Arqpnd, Camada,  sizeof (TipoRede));
      blockwrite (Arqpnd, EAnt,  sizeof (TipoMemErros));
      blockwrite (Arqpnd, EMin,  sizeof (TipoMemErros));
      blockwrite (Arqpnd, EMinimo,  sizeof (real));
      close (Arqpnd);
    end;
  end;
end;

inc (Salvamentos);
gotoxy (11, 5);
write (Salvamentos:3);
end;

```

```

procedure CarregaSinapses;
begin
{$I-}
  reset (ArqSin, 1);
{$I+}
  if IOResult = 2
  then
    begin
      writeln ('Arquivo das sinapses nÆo foi encontrado');
      Halt;
    end
  else
    begin
      blockread (ArqSin, Contador, sizeof (longint));
      blockread (ArqSin, Sinapse, sizeof (TipoSinapses));
      blockread (ArqSin, SinapseRec, sizeof (TipoSinapses));
      blockread (ArqSin, Camada, sizeof (TipoRede));
      blockread (ArqSin, EAnt, sizeof (TipoMemErros));
      blockread (ArqSin, EMin, sizeof (TipoMemErros));
      blockread (ArqSin, EMinimo, sizeof (real));
      close (ArqSin);
    end;
end;

procedure RodaRede;
var
  Soma1, Soma2 : real;
  i, j, k : byte;
begin
  with Camada[2] do
    begin
      for j := 1 to NumNeuronios do
        begin
          Soma1 := 0; Soma2 := 0;
          for i := 1 to Camada[1].NumNeuronios do
            begin
              Soma1 := Soma1 + Sinapse[2,j,i] * Camada[1].Y[t,i];
            end;
          for k := 1 to NumNeuronios do
            begin
              Soma2 := Soma2 + SinapseRec[2,j,k] * Y[t-1,k];
            end;
          X[t,j] := Soma1 + Soma2;
          Y[t,j] := FTransf (X[t,j]);
        end;
      end;
    end;
  with Camada[3] do
    begin
      for j := 1 to NumNeuronios do
        begin
          Soma1 := 0; Soma2 := 0;
          for i := 1 to Camada[2].NumNeuronios do
            Soma1 := Soma1 + Sinapse[3,j,i] * Camada[2].Y[t,i];
          end;
        end;
      end;
    end;
  end;
end;

```

```

    for k := 1 to NumNeuronios do
        begin
            Soma2 := Soma2 + SinapseRec[3,j,k] * Y[t-1,k];
        end;
        X[t,j] := Soma1 + Soma2;
        Y[t,j] := FTransf (X[t,j]);
    end;
end;
end;

function AprendeU : boolean;
var
    EPond,
    ESoma : real;
begin
    AprendeU := false {true};
    ESoma := 0;
    EPond := 0;
    for t := 1 to TamSeq do
        for j := 1 to Camada[3].NumNeuronios do
            begin
                E := ( abs (Normaliza(Input[S].XDesejado[t,j])
                    - YFinal[S,t]) / Normaliza(Input[S].XDesejado[t,j]) ) * 100;
                gotoxy (5+(t-1) div 18 * 40, 6+(t-1) mod 18);
                write ('Tempo : ', t:2, ' ', E:8:3, ' %');
                ESoma := ESoma + E;
                if (t > 5) {and (t <> 3)}
                    then EPond := EPond + E;
            {$IFDEF Setas}
                if E < EAnt[t]
                    then write (' '+#25)
                else if E > EAnt[t]
                    then write (' '+#24)
                    else write (' -');
                if E > ErroMaximo
                    then AprendeU := false;
                EAnt[t] := E;
            {$ENDIF}
        end;
        gotoxy (45, 25);
        write ('Atual : ', ESoma:8:3);
        gotoxy (45, 24);
        write ('Ponderado : ', EPond:8:3);

    {$IFDEF Minimos}
        inc (contMin);
        inc (contMin2);
        if (contMin2 >= FreqSalvaMin)
            then
                if EPond < EPondMinimo
                    then
                        begin
                            EPondMinimo := EPond;

```

```

    gotoxy (2, 24);   ClrEol;
    contmin2 := 0;
    write ('Mínimo P : ', EPondMinimo:8:3, ' Iterações : ', Contador);
    SalvaSinapses (2);
  end;
if (contMin >= FreqSalvaMin)
  then
  if ESoma < EMinimo
    then
    begin
      EMinimo := ESoma;
      contmin := 0;
      gotoxy (2, 25);   ClrEol;
      write ('Mínimo : ', EMinimo:8:3, ' Iterações : ', Contador);
      SalvaSinapses (0);
    end;
{$ENDIF}
  if Contador = MaxIteracoes then Aprende := true;
end;

function ValorInicial ( k : byte): real;
var
  Aux : real;
  Cte : word;
begin
{  randomize;
  Cte := round (1 / sqrt (Camada[k].NumNeuronios));
  Aux := random (Cte);
  ValorInicial := Aux - Cte / 2; }

  ValorInicial := 0.5 / sqrt (Camada[k-1].NumNeuronios);

end;

begin

XMax := -1E10;
XMin := 1E10;

  ClrScr;

for NumArq := NumInicio to NumFim do
  begin

    gotoxy (5, 3);
    write ('Nome do Arquivo de padrões : ');
    str (NumArq, NumArqStr);
    NomeArq := ParamStr(1)+NumArqStr;

    gotoxy (35, 3);

```

```
write (NomeArq, ' ', MaxIteracoes);
writeln;
```

```
{ Função de Transferência }
```

```
FTransf := Sigmoide;
DFTransfDX := DSigmoideDX;
```

```
{ FTransf := TangHiper;
DFTransfDX := DTangHiperDX; }
```

```
{ Arquivos }
```

```
assign (Arq, NomeArq+'.DEF'); { DEF - SERIE a ser aprendida }
reset (Arq);
```

```
assign (Arq2, NomeArq+'.OUT'); { OUT - log }
rewrite (Arq2);
```

```
assign (ArqIn, NomeArq+'.IN'); { IN - SERIE de teste }
reset (ArqIn, sizeof(real));
```

```
assign (ArqSin, NomeArq+'.SIN'); { SIN - Sinapses minimas }
assign (ArqTmp, NomeArq+'.TMP'); { SIN - Sinapses parciais }
assign (ArqPnd, NomeArq+'.PND'); { SIN - Sinapses parciais }
```

```
{ Parametros }
```

```
GetTime (Ho,Min,Se,Hu);
writeln (Arq2);
writeln (Arq2, NomeArq);
writeln (Arq2);
writeln (Arq2, 'Inicio : ',Ho:2,':',Min:2);
```

```
Contador := 0;
readln (Arq, NumCamadas);
```

```
writeln (Arq2, 'Qtd de camadas : ',NumCamadas);
```

```
for i:=1 to NumCamadas do
  readln (Arq, Camada[i].NumNeuronios);
for i:=1 to NumCamadas do
  writeln (Arq2, 'Qtd de neurônios na camada ', i, ': ',
          Camada[i].NumNeuronios);
readln (Arq, TamSeqTotal);
TamSeq := (TamSeqTotal-1) div Camada[1].NumNeuronios;
readln (Arq, NumSeq);
```

```
writeln (Arq2, 'Tamanho da Sequência : ', TamSeq);
writeln (Arq2, 'Número de Sequências : ', NumSeq);
writeln (Arq2);
```

```

close (Arq);

{ Leitura da S rie }

for S := 1 to NumSeq do
begin
  for t := 1 to TamSeq do
  begin
    for j:= 1 to Camada[1].NumNeuronios do
    begin
      blockread (ArqIn, Input[S].X[t,j], 1);
      if Input[S].X[t,j] > XMax
      then XMax := Input[S].X[t,j];
      if Input[S].X[t,j] < XMin
      then XMin := Input[S].X[t,j];
    end;
    for j:= 1 to Camada[3].NumNeuronios do
      blockread (ArqIn, Input[S].XDesejado[t,j], 1);
      seek (ArqIn, t*Camada[1].NumNeuronios);
    end;
  end;

close (ArqIn); { j leu a s rie }

{ Calculo da escala de normaliza o }

Escala := 0.2 / (XMax - XMin);
OffSet := {0.1 - Escala*XMin} 0;

for S := 1 to NumSeq do
begin
  writeln (Arq2, 'Seq ncia ', S);
  writeln (Arq2, 'Input : ');
  for t := 1 to TamSeq do
  begin
    for j:= 1 to Camada[1].NumNeuronios do
      write (Arq2, ' ', Input[S].X[t,j]:6:2);
    writeln (Arq2);
  end;
  writeln (Arq2);
  writeln (Arq2, 'Treinamento');
  writeln (Arq2, 'Saıda desejada : ');
  for t := 1 to TamSeq do
    for j:= 1 to Camada[3].NumNeuronios do
      writeln (Arq2, ' ', Input[S].XDesejado[t,j]:6:2);
    writeln (Arq2);
  end;

  writeln (Arq2);
  writeln (Arq2, 'Valor de reset : ', ValorReset:6:2);

{ writeln (Arq2, 'Coef. Aprend. : ', Beta:6:2);
  writeln (Arq2, 'Momentum : ', Momento:6:2); }

```

```

writeln (Arq2, 'Fator de Convexidade      : ', Teta:6:2);
writeln (Arq2, 'Incr. da Taxa de Aprendizado      : ', Capa:6:2);
writeln (Arq2, 'Fator de Decr. da Taxa de Aprendizado : ', Fi:6:2);

writeln (Arq2, 'Escala      : ', Escala:8:5);
if @@FTtransf = addr(Sigmoide)
  then writeln (Arq2, 'Funço de Transf. : Sigmoide')
  else writeln (Arq2, 'Funço de Transf. : Tangente Hiperblica');
writeln (Arq2);

{ Iniciaço das sinapses }

{$I-}
reset (ArqSin, 1);
{$I+}
if IOResult = 0
  then CarregaSinapses
  else
  begin
    for k := 2 to NumMaxCamadas do
      for i:= 1 to NumMaxNeuronios do
        begin
          for j:= 1 to NumMaxNeuronios do Sinapse[k,i,j] := ValorInicial(k);
          for j:= 1 to NumMaxNeuronios do SinapseRec[k,i,j] := ValorInicial(k+1);
          for j:= 1 to NumMaxNeuronios do Camada[k].DeltaAnt[i,j] := 0.25;
          for j:= 1 to NumMaxNeuronios do Camada[k].DeltaAntRec[i,j] := 0.25;
          for j:= 1 to NumMaxNeuronios do Camada[k].DAnt[i,j] := 0.25;
          for j:= 1 to NumMaxNeuronios do Camada[k].DRecAnt[i,j] := 0.25;
          for j:= 1 to NumMaxNeuronios do Camada[k].Alfa[i,j] := 0.01;
          for j:= 1 to NumMaxNeuronios do Camada[k].AlfaRec[i,j] := 0.01;
          for j:= 1 to NumMaxNeuronios do Camada[k].Mi[i,j] := 0.01;
          for j:= 1 to NumMaxNeuronios do Camada[k].MiRec[i,j] := 0.01;
          for j:= 1 to NumMaxNeuronios do Camada[k].DBarra[i,j] := 0.01;
          for j:= 1 to NumMaxNeuronios do Camada[k].DBarraRec[i,j] := 0.01;
        end;
        for i:= 1 to TamMaxSeq do EAnt[i] := 0;
        for i:= 1 to TamMaxSeq do EMin[i] := 100000;
        EMinimo := 10000;
      end;

EPondMinimo := 10000;

gotoxy (2, 25);  ClrEol;
write ('Mnimo : ', EMinimo:8:3, ' Iteraço : ', Contador);
gotoxy (2, 24);  ClrEol;
write ('Mnimo P: ', EPondMinimo:8:3, ' Iteraço : ', Contador);

ch := 'T';
if ParamStr(2) = ''
  then
  begin
    gotoxy (5,5);

```

```

write ('(T)reinamento ou (E)xecu#o ? ');
ch := ReadKey;
gotoxy (1, 5);
ClrEol;
end;
if (ch in ['E', 'e'])
then goto Executa;

{ Treinamento }

gotoxy (5, 4);
write ('TREINAMENTO');
gotoxy (5, 5);
write ('Salvo vezes');
gotoxy (11, 5);
Salvamentos := Contador div FreqSalva;
write (Salvamentos:3);

repeat
randomize;

{ Somatorio temporario das sinapses }

for k := 2 to NumMaxCamadas do
for i:= 1 to NumMaxNeuronios do
begin
for j:= 1 to NumMaxNeuronios do SW[k,i,j] := 0;
for j:= 1 to NumMaxNeuronios do SWR[k,i, j] := 0;
end;

{ Mudanca de seq#ncia }

for S := 1 to NumSeq do
begin

{ Pesos }

for k:= 2 to NumCamadas do
for i := 1 to Camada[k].NumNeuronios do
for j:= 1 to Camada[k-1].NumNeuronios do
begin
Camada[k].W[i,j] := 0;
Camada[k].WR[j,i] := 0;
end;

{ Valor de Reset }

for k := 1 to NumCamadas do
with Camada[k] do
begin
for t := 0 to TamSeq do
for j := 1 to NumNeuronios do
begin

```



```

        X[t, j] := ValorReset;
        Y[t, j] := ValorReset;
    end;
end;

{ Execucao da rede }

for t:= 1 to TamSeq do
    begin
        for j := 1 to Camada[1].NumNeuronios do
            begin
                Camada[1].X[t,j] := Normaliza(Input[S].X[t,j]);
                Camada[1].Y[t,j] := Camada[1].X[t,j];
            end;
        ProcessaRede;
    end;

    { C lculo de erros }

    for t := TamSeq downto 1 do
        begin
            for j := 1 to Camada[3].NumNeuronios do
                YDesejado[t,j] := FTransf (Normaliza(Input[S].XDesejado[t,j]));
            CalculaErros;
        end;

        { Altera o somatçrio de sinapses }

        for t := 1 to TamSeq do AlteraPesos;

        { Altera os pesos }

        AtualizaPesos;

        for t := 1 to TamSeq do
            YFinal[S,t] := Camada[3].X[t,1];

        end;

    inc (Contador);
    gotoxy (45, 5);
    write (Contador);

    AlteraSinapses ;

    if (Contador mod FreqSalva = 0) then SalvaSinapses (1);

{   until Contador = MaxIteracoes; }
    until Aprendeu ;

{ fim do treinamento }

writeln (Arq2, Contador, ' iteraçoes');

```

```

writeln (Arq2);

for S := 1 to NumSeq do
begin
writeln (Arq2, 'Seq ncia ', S);
writeln (Arq2, ' Esperados Calculados');
for t := 1 to TamSeq do
begin
E := ( abs (Normaliza(Input[S].XDesejado[t,1]) - YFinal[S,t]) /Normaliza(Input[S].XDesejado[t,1]) ) *
100;
write (Arq2, ' ',{Normaliza}(Input[S].XDesejado[t,1]):10:4,
' ',DeNormaliza(YFinal[S,t]):10:4, ' ',E:6:3,'% de erro');
writeln (Arq2);
end;
writeln (Arq2);
end;

{$IFDEF Minimos}
rename (ArqTmp, NomeArq+'.SIN');
{$ENDIF}

if not (ch in ['E','e'])
then goto Fim;

Executa :

assign (ArqIn, ParamStr(1)+'.IN');
reset (ArqIn, sizeof(real));

CarregaSinapses;

XAux[1] := 0;
XAux[2] := 0;
XAux[3] := 0;

gotoxy (5, 6);
write (' resultados escritos em '+NomeArq+'.OUT');

{ for i:= 0 to TamSerie-(TamSeq*Camada[1].NumNeuronios)-1 do }

for k:= 1 to NumCamadas do
for j:= 1 to Camada[k].NumNeuronios do
begin
Camada[k].X[0, j] := ValorReset;
Camada[k].Y[0, j] := ValorReset ;
end;

for i:= 1 to TamSeq do
begin

{ for t := 1 to TamSeq do
begin }

```

```

for j:= 1 to Camada[1].NumNeuronios do
  begin
    blockread (ArqIn, Camada[1].X[t,j], 1);
    Camada[1].X[t,j] := Normaliza(Camada[1].X[t,j]);
    Camada[1].Y[t,j] := Camada[1].X[t,j];
  end;
{ end; }

blockread (ArqIn, YD, 1);

for t := 1 to TamSeq do ProcessaRede;

if ch in ['e','E']
  then
    for j := 1 to Camada[3].NumNeuronios do
      begin
        gotoxy (1, 6+i);
        write (i:2);
        write ('| Valor esperado : ', YD:11:3);
        write ('| Valor calculado ', Denormaliza(Camada[3].X[TamSeq,j]):11:3);
        E := ( abs (YD - Denormaliza(Camada[3].X[TamSeq,j])) / YD) * 100;
        writeln ('| ',E:7:3, '% de erro');
        writeln ;
      end;
    for j := 1 to Camada[3].NumNeuronios do
      begin
        writeln (Arq2, i);
        writeln (Arq2,'Valor esperado : ', YD:8:3);
        writeln (Arq2,'Valor calculado ', Denormaliza(Camada[3].X[TamSeq,j]):8:3);
        E := ( abs (YD - Denormaliza(Camada[3].X[TamSeq,j])) / YD) * 100;
        writeln (Arq2, ' ',E:6:3, '% de erro');
        writeln (Arq2);
      end;

    seek (ArqIn, filepos(ArqIn)-1);
    gotoxy (5, 6);
    write (i:3);
  end;

close (ArqIn);

Fim:

GetTime (Ho,Min,Se,Hu);
writeln (Arq2, 'Fim : ',Ho:2:', ',Min:2);
writeln (Arq2);

readln;
ClrScr;

close (Arq2);
end;
end.

```

## APÊNDICE B

## B.1 - SÉRIES UTILIZADAS

## 1 - Consumo Mensal de Energia (MWh)

Ano	Janeiro	Fevereiro	Março	Abril	Maiο	Junho
1954	226.705	206.998	232.202	218.260	222.989	223.850
1955	269.679	241.588	285.170	271.828	283.323	275.758
1956	373.499	342.117	364.302	343.677	348.255	342.601
1957	382.966	337.531	368.615	350.038	353.502	336.454
1958	377.395	334.631	364.840	339.673	354.958	342.003
1959	419.325	379.322	413.931	382.444	389.134	403.207
1960	448.025	428.626	446.652	399.422	402.203	395.718
1961	409.234	361.229	400.510	388.159	392.716	379.395
1962	465.063	407.779	440.558	418.149	425.788	417.154
1963	473.874	428.516	455.636	447.072	449.772	437.582
1964	487.303	440.832	468.165	442.546	457.999	462.126
1965	500.828	450.747	497.278	457.820	460.504	462.007
1966	501.665	468.958	519.319	484.456	481.817	497.593
1967	560.572	522.916	545.742	516.294	536.430	541.045
1968	617.747	587.046	586.097	540.829	567.503	603.869
1969	650.387	613.735	634.894	621.336	632.423	646.633
1970	689.626	616.488	676.767	612.048	624.052	651.663

Ano	Julho	Agosto	Setembro	Outubro	Novembr o	Dezembro
1954	215.966	231.139	227.034	243.624	255.703	274.755
1955	291.888	313.815	306.543	337.649	342.701	366.698
1956	318.590	349.833	336.575	371.337	371.025	376.397
1957	346.738	365.611	349.416	373.400	362.110	370.140
1958	351.529	361.280	358.349	385.891	376.232	429.165
1959	390.373	396.642	386.770	389.306	402.078	448.236
1960	393.719	418.406	398.758	431.902	292.340	422.172
1961	397.140	411.101	425.239	422.996	405.604	454.618
1962	400.512	433.831	412.572	432.379	439.149	451.799
1963	445.563	451.678	433.395	461.069	462.409	497.878
1964	460.533	450.313	445.164	458.087	461.694	461.664
1965	458.618	480.103	473.550	484.278	483.624	511.386
1966	503.168	515.626	489.971	521.223	535.843	556.692
1967	539.120	566.032	539.487	569.215	570.595	579.917
1968	611.445	639.375	562.923	634.066	622.171	648.651
1969	717.611	699.046	638.600	642.998	632.139	709.500
1970	698.192	715.772	651.048	652.476	-	-

## 2 - Número Mensal de Passageiros de Vôos Internacionais

Ano	Janeiro	Fevereiro	Março	Abril	Maió	Junho
1949	112	118	132	129	121	135
1950	115	126	141	135	125	149
1951	145	150	178	163	172	178
1952	171	180	193	181	183	218
1953	196	196	236	235	229	243
1954	204	188	235	227	234	264
1955	242	233	267	269	270	278
1956	284	277	317	313	318	306
1957	315	301	356	348	355	336
1958	340	318	362	348	363	337
1959	360	342	406	396	420	405
1960	417	391	419	461	472	535

Ano	Julho	Agosto	Setembro	Outubro	Novembr o	Dezembro
1949	148	148	136	119	104	118
1950	170	170	158	133	114	140
1951	199	199	184	162	146	166
1952	230	242	209	191	172	194
1953	264	272	237	211	180	201
1954	302	293	259	229	203	229
1955	315	364	347	312	274	237
1956	374	413	405	355	306	271
1957	422	465	467	404	347	305
1958	435	491	505	404	359	310
1959	472	548	559	463	407	362
1960	622	606	508	461	390	432

### 3 - Vendas de Equipamento Elétrico

Período	1	2	3	4	5	6
1961	26.411	22.899	23.145	24.000	22.658	29.400
1962	25.566	33.551	24.506	22.776	25.925	31.479
1963	23.239	22.862	23.677	18.874	26.562	25.044
1964	23.196	20.829	24.895	30.195	32.095	27.836
1965	29.165	20.829	20.620	22.170	33.997	26.486
1966	28.418	20.424	24.794	30.729	27.093	32.150
1967	25.019	16.736	24.860	28.431	27.313	30.065
1968	21.471	39.538	20.403	25.078	30.701	41.026
1969	34.962	25.846	30.675	28.020	30.685	37.270
1970	30.963	32.660	25.312	34.424	27.477	41.257
1971	39.746	31.358	34.316	24.315	44.774	36.533
1972	41.890	35.201	43.779	36.674	47.803	56.422

Período	7	8	9	10	11	12	13
1961	32.527	31.992	26.592	32.121	34.068	21.587	21.566
1962	27.641	32.620	32.772	31.900	27.291	21.968	21.188
1963	26.565	25.176	29.006	32.783	38.834	28.863	24.203
1964	30.684	31.414	30.577	32.047	25.599	31.813	18.254
1965	33.289	32.003	31.482	32.891	33.483	23.534	20.662
1966	33.289	29.457	32.486	30.304	26.566	22.205	23.649
1967	24.068	39.488	31.702	31.483	32.629	24.207	30.826
1968	36.203	37.980	28.332	31.377	23.466	31.414	18.642
1969	32.620	36.254	34.181	39.680	36.536	36.828	32.252
1970	34.577	35.140	38.909	43.314	36.099	38.794	29.705
1971	42.131	50.904	40.289	36.485	42.923	38.697	44.237
1972	55.269	39.087	55.121	57.747	47.948	53.798	52.449

## B.2 - TABELAS DE RESULTADOS DOS TESTES DAS SÉRIES

### 1 - Consumo Mensal de Energia (MWh)

#### 1.1 - OHIO 1

Início : 13:32

Qtd de camadas : 3

Qtd de neurônios na camada 1: 3

Qtd de neurônios na camada 2: 2

Qtd de neurônios na camada 3: 1

Tamanho da Seqüência (n): 8

Número de Seqüências : 1

#### *Seqüência 1*

Input :

226.705,00	206.998,00	232.202,00
218.260,00	222.989,00	223.850,00
215.966,00	231.139,00	227.034,00
243.624,00	255.703,00	274.755,00
269.679,00	241.588,00	285.170,00
271.828,00	283.323,00	275.758,00
291.888,00	313.815,00	306.543,00
337.649,00	342.701,00	366.698,00

#### *Treinamento*

Saída desejada :

218.260,00
215.966,00
243.624,00
269.679,00
271.828,00
291.888,00
337.649,00
373.499,00

Valor de reset : 0,25

Coef. Aprend. : 0,10

Momentum : 0,10

Escala : 0,00001

150.000 iterações

#### *Seqüência 1*

Esperados	Calculados	
0,1564	0,0517	66,942% de erro
0,1449	0,0608	58,027% de erro
0,2835	0,3519	24,123% de erro
0,4140	0,4600	11,106% de erro
0,4248	0,5281	24,333% de erro
0,5252	0,5984	13,921% de erro
0,7545	0,6932	8,126% de erro
0,9341	0,7932	15,080% de erro

*Execução*

## Previsão do tempo

1	Valor esperado : 343677,000 Valor calculado 352849,288	2,669% de erro
2	Valor esperado : 348255,000 Valor calculado 351424,104	0,910% de erro
3	Valor esperado : 342601,000 Valor calculado 352586,941	2,915% de erro
4	Valor esperado : 318590,000 Valor calculado 350582,726	10,042% de erro
5	Valor esperado : 349833,000 Valor calculado 348600,527	0,352% de erro
6	Valor esperado : 336575,000 Valor calculado 347289,365	3,183% de erro
7	Valor esperado : 371337,000 Valor calculado 346810,736	6,605% de erro
8	Valor esperado : 371025,000 Valor calculado 351655,633	5,221% de erro
9	Valor esperado : 376397,000 Valor calculado 353512,641	6,080% de erro
10	Valor esperado : 382966,000 Valor calculado 356505,511	6,909% de erro

Fim : 14:18

**1.2 - OHIO 2**

Início : 14:18

Qtd de camadas : 3

Qtd de neurônios na camada 1: 3

Qtd de neurônios na camada 2: 2

Qtd de neurônios na camada 3: 1

Tamanho da Seqüência (n) : 10

Número de Seqüências : 1

*Seqüência 1*

Input :

226705,00 206998,00 232202,00  
 218260,00 222989,00 223850,00  
 215966,00 231139,00 227034,00  
 243624,00 255703,00 274755,00  
 269679,00 241588,00 285170,00  
 271828,00 283323,00 275758,00  
 291888,00 313815,00 306543,00



337649,00 342701,00 366698,00  
 373499,00 342117,00 364302,00  
 343677,00 348255,00 342601,00

### Treinamento

Saída desejada :

218260,00  
 215966,00  
 243624,00  
 269679,00  
 271828,00  
 291888,00  
 337649,00  
 373499,00  
 343677,00  
 318590,00

Valor de reset : 0,25

Coef. Aprend. : 0,10

Momentum : 0,10

Escala : 0,00000

150000 iterações

### Seqüência 1

Esperados	Calculados	
0,1541	0,0586	62,000% de erro
0,1431	0,2300	60,706% de erro
0,2760	0,2707	1,920% de erro
0,4012	0,3613	9,936% de erro
0,4115	0,4359	5,941% de erro
0,5079	0,5366	5,662% de erro
0,7277	0,6726	7,574% de erro
0,9000	0,7490	16,778% de erro
0,7567	0,7712	1,911% de erro
0,6362	0,7705	21,118% de erro

### Execução

Previsão do tempo

1	Valor esperado : 371337,000	
	Valor calculado 345405,879	6,983% de erro
2	Valor esperado : 371025,000	
	Valor calculado 347185,966	6,425% de erro
3	Valor esperado : 376397,000	
	Valor calculado 347508,586	7,675% de erro
4	Valor esperado : 382966,000	
	Valor calculado 348044,826	9,119% de erro
5	Valor esperado : 337531,000	
	Valor calculado 348586,619	3,275% de erro
6	Valor esperado : 368615,000	
	Valor calculado 348062,930	5,575% de erro

7	Valor esperado : 350038,000 Valor calculado 348357,710	0,480% de erro
8	Valor esperado : 353502,000 Valor calculado 347447,753	1,713% de erro
9	Valor esperado : 336454,000 Valor calculado 347930,959	3,411% de erro
10	Valor esperado : 346738,000 Valor calculado 347072,391	0,096% de erro

Fim : 15:10

### 1.3 - OHIO 3

Início : 22:32

Qtd de camadas : 3

Qtd de neurônios na camada 1: 3

Qtd de neurônios na camada 2: 2

Qtd de neurônios na camada 3: 1

Tamanho da Seqüência (n) : 12

Número de Seqüências : 1

#### *Seqüência 1*

Input :

226705,00 206998,00 232202,00  
 218260,00 222989,00 223850,00  
 215966,00 231139,00 227034,00  
 243624,00 255703,00 274755,00  
 269679,00 241588,00 285170,00  
 271828,00 283323,00 275758,00  
 291888,00 313815,00 306543,00  
 337649,00 342701,00 366698,00  
 373499,00 342117,00 364302,00  
 343677,00 348255,00 342601,00  
 318590,00 349833,00 336575,00  
 371337,00 371025,00 376397,00

#### *Treinamento*

Saída desejada :

218260,00  
 215966,00  
 243624,00  
 269679,00  
 271828,00  
 291888,00  
 337649,00  
 373499,00  
 343677,00  
 318590,00  
 371337,00  
 382966,00

Valor de reset : 0,25  
 Coef. Aprend. : 0,10  
 Momentum : 0,10  
 Escala : 0,00000

150000 iterações

*Seqüência 1*

Esperados	Calculados	
0,1532	0,0492	67,858% de erro
0,1424	0,0565	60,285% de erro
0,2730	0,3488	27,772% de erro
0,3960	0,4420	11,609% de erro
0,4062	0,4951	21,901% de erro
0,5009	0,5695	13,705% de erro
0,7170	0,6627	7,579% de erro
0,8863	0,7631	13,897% de erro
0,7455	0,7947	6,605% de erro
0,6270	0,7823	24,774% de erro
0,8761	0,7665	12,509% de erro
0,9310	0,8182	12,123% de erro

*Execução*

Previsão do tempo

1	Valor esperado : 350038,000 Valor calculado 357090,587	2,015% de erro
2	Valor esperado : 353502,000 Valor calculado 355556,581	0,581% de erro
3	Valor esperado : 336454,000 Valor calculado 356438,276	5,940% de erro
4	Valor esperado : 346738,000 Valor calculado 353298,672	1,892% de erro
5	Valor esperado : 365611,000 Valor calculado 351979,781	3,728% de erro
6	Valor esperado : 349416,000 Valor calculado 353578,354	1,191% de erro
7	Valor esperado : 373400,000 Valor calculado 355456,749	4,805% de erro
8	Valor esperado : 362110,000 Valor calculado 356852,083	1,452% de erro
9	Valor esperado : 370140,000 Valor calculado 357689,537	3,364% de erro
10	Valor esperado : 377395,000 Valor calculado 359100,226	4,848% de erro

Fim : 23:34

## 1.4 - OHIO 4

Início : 15:49

Qtd de camadas : 3

Qtd de neurônios na camada 1: 3

Qtd de neurônios na camada 2: 2

Qtd de neurônios na camada 3: 1

Tamanho da Seqüência (n) : 13

Número de Seqüências : 1

### *Seqüência 1*

Input :

226705,00	206998,00	232202,00
218260,00	222989,00	223850,00
215966,00	231139,00	227034,00
243624,00	255703,00	274755,00
269679,00	241588,00	285170,00
271828,00	283323,00	275758,00
291888,00	313815,00	306543,00
337649,00	342701,00	366698,00
373499,00	342117,00	364302,00
343677,00	348255,00	342601,00
318590,00	349833,00	336575,00
371337,00	371025,00	376397,00
382966,00	337531,00	368615,00

### *Treinamento*

Saída desejada :

218260,00
215966,00
243624,00
269679,00
271828,00
291888,00
337649,00
373499,00
343677,00
318590,00
371337,00
382966,00
350038,00

Valor de reset : 0,25

Coef. Aprend. : 0,10

Momentum : 0,10

Escala : 0,00000

50000 iterações

## Sequência 1

Esperados	Calculados	
0,1512	-0,0238	115,732% de erro
0,1408	0,1389	1,317% de erro
0,2665	0,2801	5,112% de erro
0,3850	0,4579	18,958% de erro
0,3947	0,4998	26,618% de erro
0,4859	0,5533	13,858% de erro
0,6940	0,6386	7,986% de erro
0,8570	0,7296	14,860% de erro
0,7214	0,7606	5,432% de erro
0,6073	0,7487	23,274% de erro
0,8471	0,7319	13,597% de erro
0,9000	0,7731	14,095% de erro
0,7503	0,7739	3,147% de erro

## Execução

## Previsão do tempo

1	Valor esperado : 346738,000 Valor calculado 350889,748	1,197% de erro
2	Valor esperado : 365611,000 Valor calculado 350474,415	4,140% de erro
3	Valor esperado : 349416,000 Valor calculado 351613,275	0,629% de erro
4	Valor esperado : 373400,000 Valor calculado 352057,240	5,716% de erro
5	Valor esperado : 362110,000 Valor calculado 354280,090	2,162% de erro
6	Valor esperado : 370140,000 Valor calculado 353921,756	4,382% de erro
7	Valor esperado : 377395,000 Valor calculado 355579,552	5,781% de erro
8	Valor esperado : 334631,000 Valor calculado 356140,009	6,428% de erro
9	Valor esperado : 364840,000 Valor calculado 354069,846	2,952% de erro
10	Valor esperado : 339673,000 Valor calculado 354556,313	4,382% de erro

Fim : 16:14

**1.5 - OHIO 5**

Início : 13:58

Qtd de camadas : 3

Qtd de neurônios na camada 1: 3

Qtd de neurônios na camada 2: 2

Qtd de neurônios na camada 3: 1

Tamanho da Sequência (n) : 15

Número de Sequências : 1

*Sequência 1*

Input :

226705,00 206998,00 232202,00  
 218260,00 222989,00 223850,00  
 215966,00 231139,00 227034,00  
 243624,00 255703,00 274755,00  
 269679,00 241588,00 285170,00  
 271828,00 283323,00 275758,00  
 291888,00 313815,00 306543,00  
 337649,00 342701,00 366698,00  
 373499,00 342117,00 364302,00  
 343677,00 348255,00 342601,00  
 318590,00 349833,00 336575,00  
 371337,00 371025,00 376397,00  
 382966,00 337531,00 368615,00  
 350038,00 353502,00 336454,00  
 346738,00 365611,00 349416,00

*Treinamento*

Saída desejada :

218260,00  
 215966,00  
 243624,00  
 269679,00  
 271828,00  
 291888,00  
 337649,00  
 373499,00  
 343677,00  
 318590,00  
 371337,00  
 382966,00  
 350038,00  
 346738,00  
 373400,00

Valor de reset : 0,25

Coef. Aprend. : 0,10

Momentum : 0,10

Escala : 0,00000

150.000 iterações

*Seqüência 1*

Esperados	Calculados	
0,1512	0,0481	68,199% de erro
0,1408	0,0568	59,619% de erro
0,2665	0,3416	28,175% de erro
0,3850	0,4305	11,822% de erro
0,3947	0,4785	21,224% de erro
0,4859	0,5525	13,695% de erro
0,6940	0,6433	7,299% de erro
0,8570	0,7418	13,438% de erro
0,7214	0,7712	6,901% de erro
0,6073	0,7586	24,901% de erro
0,8471	0,7434	12,247% de erro
0,9000	0,7949	11,682% de erro
0,7503	0,7842	4,519% de erro
0,7353	0,7668	4,279% de erro
0,8565	0,7775	9,227% de erro

*Execução*

## Previsão do tempo

1	Valor esperado : 377395,000 Valor calculado 359732,389	4,680% de erro
2	Valor esperado : 334631,000 Valor calculado 360432,193	7,710% de erro
3	Valor esperado : 364840,000 Valor calculado 358826,347	1,648% de erro
4	Valor esperado : 339673,000 Valor calculado 356513,656	4,958% de erro
5	Valor esperado : 354958,000 Valor calculado 354187,873	0,217% de erro
6	Valor esperado : 342003,000 Valor calculado 355026,114	3,808% de erro
7	Valor esperado : 351529,000 Valor calculado 353240,599	0,487% de erro
8	Valor esperado : 361280,000 Valor calculado 353723,428	2,092% de erro
9	Valor esperado : 358349,000 Valor calculado 354786,979	0,994% de erro
10	Valor esperado : 385891,000 Valor calculado 356580,729	7,595% de erro

Fim : 15:16

## 2 - Número Mensal de Passageiros de Vôos Internacionais

### 2.1 - Vôos 1

Início : 14:26

Qtd de camadas : 3

Qtd de neurônios na camada 1: 3

Qtd de neurônios na camada 2: 2

Qtd de neurônios na camada 3: 1

Tamanho da Seqüência (n) : 8

Número de Seqüências : 1

#### *Seqüência 1*

Input :

112,00 118,00 132,00

129,00 121,00 135,00

148,00 148,00 136,00

119,00 104,00 118,00

115,00 126,00 141,00

135,00 125,00 149,00

170,00 170,00 158,00

133,00 114,00 140,00

#### *Treinamento*

Saída desejada :

129,00

148,00

119,00

115,00

135,00

170,00

133,00

145,00

Valor de reset : 0,25

Coef. Aprend. : 0,10

Momentum : 0,10

Escala : 0,01212

150000 iterações

#### *Seqüência 1*

Esperados	Calculados	
0,4030	0,3901	3,216% de erro
0,6333	0,5460	13,790% de erro
0,2818	0,2668	5,346% de erro
0,2333	0,2283	2,171% de erro
0,4758	0,5741	20,671% de erro
0,9000	0,8432	6,310% de erro
0,4515	0,4429	1,898% de erro
0,5970	0,6726	12,674% de erro



*Execução*

## Previsão do tempo

1	Valor esperado : 163,000 Valor calculado 171,487	5,207% de erro
2	Valor esperado : 172,000 Valor calculado 135,981	20,941% de erro
3	Valor esperado : 178,000 Valor calculado 143,701	19,269% de erro
4	Valor esperado : 199,000 Valor calculado 162,148	18,519% de erro
5	Valor esperado : 199,000 Valor calculado 174,808	12,157% de erro
6	Valor esperado : 184,000 Valor calculado 147,185	20,008% de erro
7	Valor esperado : 162,000 Valor calculado 136,974	15,448% de erro
8	Valor esperado : 146,000 Valor calculado 123,390	15,487% de erro
9	Valor esperado : 166,000 Valor calculado 106,183	36,034% de erro
10	Valor esperado : 171,000 Valor calculado 162,418	5,018% de erro

Fim : 15: 8

**2.2 Vãos 2**

Início : 12:57

Qtd de camadas : 3

Qtd de neurônios na camada 1: 3

Qtd de neurônios na camada 2: 2

Qtd de neurônios na camada 3: 1

Tamanho da Seqüência (n) : 10

Número de Seqüências : 1

*Seqüência 1*

Input :

112,00 118,00 132,00  
 129,00 121,00 135,00  
 148,00 148,00 136,00  
 119,00 104,00 118,00  
 115,00 126,00 141,00  
 135,00 125,00 149,00  
 170,00 170,00 158,00  
 133,00 114,00 140,00  
 145,00 150,00 178,00  
 163,00 172,00 178,00

*Treinamento*

Saída desejada :

129,00  
 148,00  
 119,00  
 115,00  
 135,00  
 170,00  
 133,00  
 145,00  
 163,00  
 199,00

Valor de reset : 0,25

Coef. Aprend. : 0,10

Momentum : 0,10

Escala : 0,01081

150.000 iterações

*Seqüência 1*

Esperados	Calculados	
0,3703	0,1884	49,108% de erro
0,5757	0,3590	37,647% de erro
0,2622	0,2509	4,297% de erro
0,2189	0,2289	4,550% de erro
0,4351	0,5774	32,698% de erro
0,8135	0,7158	12,008% de erro
0,4135	0,5300	28,178% de erro
0,5432	0,6572	20,983% de erro
0,7378	0,8982	21,731% de erro
1,1270	0,8716	22,660% de erro

*Execução*

Previsão do tempo

1	Valor esperado : 162,000	
	Valor calculado 161,332	0,412% de erro
2	Valor esperado : 146,000	
	Valor calculado 131,494	9,935% de erro
3	Valor esperado : 166,000	
	Valor calculado 122,350	26,295% de erro

4	Valor esperado : 171,000 Valor calculado 170,316	0,400% de erro
5	Valor esperado : 180,000 Valor calculado 162,109	9,940% de erro
6	Valor esperado : 193,000 Valor calculado 161,507	16,317% de erro
7	Valor esperado : 181,000 Valor calculado 181,124	0,068% de erro
8	Valor esperado : 183,000 Valor calculado 161,143	11,943% de erro
9	Valor esperado : 218,000 Valor calculado 165,628	24,024% de erro
10	Valor esperado : 230,000 Valor calculado 192,290	16,396% de erro

Fim : 13:50

### 2.3 - Vãos 3

Início : 11:52

Qtd de camadas : 3

Qtd de neurônios na camada 1: 3

Qtd de neurônios na camada 2: 2

Qtd de neurônios na camada 3: 1

Tamanho da Seqüência (n) : 12

Número de Seqüências : 1

*Seqüência 1*

Input :

112,00 118,00 132,00  
 129,00 121,00 135,00  
 148,00 148,00 136,00  
 119,00 104,00 118,00  
 115,00 126,00 141,00  
 135,00 125,00 149,00  
 170,00 170,00 158,00  
 133,00 114,00 140,00  
 145,00 150,00 178,00  
 163,00 172,00 178,00  
 199,00 199,00 184,00  
 162,00 146,00 166,00

*Treinamento*

Saída desejada :

129,00  
 148,00  
 119,00  
 115,00  
 135,00  
 170,00  
 133,00  
 145,00  
 163,00  
 199,00  
 162,00  
 171,00

Valor de reset : 0,25

Coef. Aprend. : 0,10

Momentum : 0,10

Escala : 0,00842

150.000 iterações

*Seqüência 1*

Esperados	Calculados	
0,3105	0,2384	23,243% de erro
0,4705	0,3433	27,048% de erro
0,2263	0,2087	7,790% de erro
0,1926	0,1839	4,509% de erro
0,3611	0,5517	52,815% de erro
0,6558	0,5860	10,644% de erro
0,3442	0,4612	33,989% de erro
0,4453	0,4815	8,147% de erro
0,5968	0,6933	16,164% de erro
0,9000	0,6673	25,852% de erro
0,5884	0,6311	7,250% de erro
0,6642	0,6268	5,634% de erro

*Execução*

Previsão do tempo

1	Valor esperado : 181,000 Valor calculado 177,891	1,718% de erro
2	Valor esperado : 183,000 Valor calculado 164,572	10,070% de erro
3	Valor esperado : 218,000 Valor calculado 167,351	23,233% de erro
4	Valor esperado : 230,000 Valor calculado 186,479	18,922% de erro
5	Valor esperado : 242,000 Valor calculado 183,203	24,296% de erro
6	Valor esperado : 209,000 Valor calculado 184,537	11,705% de erro

7	Valor esperado : 191,000 Valor calculado 178,375	6,610% de erro
8	Valor esperado : 172,000 Valor calculado 169,445	1,486% de erro
9	Valor esperado : 194,000 Valor calculado 152,215	21,539% de erro
10	Valor esperado : 196,000 Valor calculado 181,114	7,595% de erro

Fim : 12:55

## 2.4 - Vôos 4

Início : 17: 5

Qtd de camadas : 3

Qtd de neurônios na camada 1: 3

Qtd de neurônios na camada 2:

Qtd de neurônios na camada 3: 1

Tamanho da Sequência (n) : 13

Número de Sequências : 1

### *Sequência 1*

Input :

112,00 118,00 132,00  
 129,00 121,00 135,00  
 148,00 148,00 136,00  
 119,00 104,00 118,00  
 115,00 126,00 141,00  
 135,00 125,00 149,00  
 170,00 170,00 158,00  
 133,00 114,00 140,00  
 145,00 150,00 178,00  
 163,00 172,00 178,00  
 199,00 199,00 184,00  
 162,00 146,00 166,00  
 171,00 180,00 193,00

### *Treinamento*

Saída desejada :

129,00  
 148,00  
 119,00  
 115,00  
 135,00  
 170,00  
 133,00  
 145,00  
 163,00  
 199,00  
 162,00  
 171,00  
 181,00

Valor de reset : 0,25  
 Coef. Aprend. : 0,10  
 Momentum : 0,10  
 Escala : 0,00842

150.000 iterações

*Seqüência 1*

Esperados	Calculados	
0,3105	0,2360	24,006% de erro
0,4705	0,3561	24,314% de erro
0,2263	0,1935	14,484% de erro
0,1926	0,1765	8,351% de erro
0,3611	0,5526	53,046% de erro
0,6558	0,5966	9,019% de erro
0,3442	0,4555	32,336% de erro
0,4453	0,4871	9,391% de erro
0,5968	0,7000	17,282% de erro
0,9000	0,6763	24,850% de erro
0,5884	0,6296	6,997% de erro
0,6642	0,6335	4,628% de erro
0,7484	0,7291	2,576% de erro

*Execução*

Previsão do tempo

1	Valor esperado : 230,000 Valor calculado : 187,175	18,620% de erro
2	Valor esperado : 242,000 Valor calculado : 183,643	24,114% de erro
3	Valor esperado : 209,000 Valor calculado : 184,833	11,563% de erro
4	Valor esperado : 191,000 Valor calculado : 178,099	6,755% de erro
5	Valor esperado : 172,000 Valor calculado : 168,055	2,294% de erro
6	Valor esperado : 194,000 Valor calculado : 150,273	22,539% de erro
7	Valor esperado : 196,000 Valor calculado : 181,596	7,349% de erro
8	Valor esperado : 196,000 Valor calculado : 178,081	9,142% de erro
9	Valor esperado : 236,000 Valor calculado : 172,846	26,760% de erro
10	Valor esperado : 235,000 Valor calculado : 189,087	19,537% de erro

Fim : 18:13

## 2.5 - Vãos 5

Início : 16:43

Qtd de camadas : 3

Qtd de neurônios na camada 1: 3

Qtd de neurônios na camada 2: 2

Qtd de neurônios na camada 3: 1

Tamanho da Sequência (n) : 15

Número de Sequências : 1

### *Sequência 1*

Input :

112,00 118,00 132,00  
 129,00 121,00 135,00  
 148,00 148,00 136,00  
 119,00 104,00 118,00  
 115,00 126,00 141,00  
 135,00 125,00 149,00  
 170,00 170,00 158,00  
 133,00 114,00 140,00  
 145,00 150,00 178,00  
 163,00 172,00 178,00  
 199,00 199,00 184,00  
 162,00 146,00 166,00  
 171,00 180,00 193,00  
 181,00 183,00 218,00  
 230,00 242,00 209,00

### *Treinamento*

Saida desejada :

129,00  
 148,00  
 119,00  
 115,00  
 135,00  
 170,00  
 133,00  
 145,00  
 163,00  
 199,00  
 162,00  
 171,00  
 181,00  
 230,00  
 191,00

Valor de reset : 0,25

Coef. Aprend. : 0,10

Momentum : 0,10

Escala : 0,00580

150.000 iterações

## Sequência 1

Esperados	Calculados	
0,2449	0,1835	25,090% de erro
0,3551	0,4116	15,922% de erro
0,1870	0,2040	9,130% de erro
0,1638	0,1551	5,316% de erro
0,2797	0,3160	12,973% de erro
0,4826	0,4187	13,239% de erro
0,2681	0,2730	1,812% de erro
0,3377	0,3046	9,788% de erro
0,4420	0,5478	23,934% de erro
0,6507	0,5728	11,977% de erro
0,4362	0,4484	2,791% de erro
0,4884	0,4445	8,988% de erro
0,5464	0,5906	8,094% de erro
0,8304	0,7935	4,447% de erro
0,6043	0,6258	3,547% de erro

## Execução

## Previsão do tempo

1	Valor esperado : 196,000 Valor calculado 197,084	0,553% de erro
2	Valor esperado : 196,000 Valor calculado 184,989	5,618% de erro
3	Valor esperado : 236,000 Valor calculado 168,535	28,587% de erro
4	Valor esperado : 235,000 Valor calculado 233,567	0,610% de erro
5	Valor esperado : 229,000 Valor calculado 221,159	3,424% de erro
6	Valor esperado : 243,000 Valor calculado 196,594	19,097% de erro
7	Valor esperado : 264,000 Valor calculado 240,425	8,930% de erro
8	Valor esperado : 272,000 Valor calculado 247,386	9,049% de erro
9	Valor esperado : 237,000 Valor calculado 237,545	0,230% de erro
10	Valor esperado : 211,000 Valor calculado 217,135	2,907% de erro

Fim : 18: 1



### 3 - Vendas de Equipamento Elétrico

#### 3.1 - Vendas 1

Início : 21:13

Qtd de camadas : 3

Qtd de neurônios na camada 1: 3

Qtd de neurônios na camada 2: 2

Qtd de neurônios na camada 3: 1

Tamanho da Seqüência : 8

Número de Seqüências : 1

##### *Seqüência 1*

Input :

26411,00 22899,00 23145,00  
 24000,00 22658,00 29400,00  
 32527,00 31992,00 26592,00  
 32121,00 34068,00 21587,00  
 21566,00 25566,00 33551,00  
 24506,00 22776,00 25925,00  
 31479,00 27641,00 32620,00  
 32772,00 31900,00 27291,00

##### *Treinamento*

Saída desejada :

24000,00  
 32527,00  
 32121,00  
 21566,00  
 24506,00  
 31479,00  
 32772,00  
 21968,00

Valor de reset : 0,25

Coef. Aprend. : 0,20

Momentum : 0,10

Escala : 0,00006

150.000 iterações

##### *Seqüência 1*

Esperados	Calculados	
0,2558	0,2073	18,937% de erro
0,8014	0,9539	19,029% de erro
0,7754	0,7192	7,245% de erro
0,1000	0,1215	21,489% de erro
0,2881	0,3158	9,600% de erro
0,7343	0,7780	5,946% de erro
0,8171	0,7417	9,221% de erro
0,1257	0,2520	100,418% de erro

*Execução*

## Previsão do tempo

1 Valor esperado : 22862,000  
 Valor calculado 22668,817 0,845% de erro

Fim : 21:50

**3.1 - Vendas 2**

Início : 23:34

Qtd de camadas : 3

Qtd de neurônios na camada 1: 3

Qtd de neurônios na camada 2: 2

Qtd de neurônios na camada 3: 1

Tamanho da Seqüência (n) : 10

Número de Seqüências : 1

*Seqüência 1*

Input :

26411,00 22899,00 23145,00  
 24000,00 22658,00 29400,00  
 32527,00 31992,00 26592,00  
 32121,00 34068,00 21587,00  
 21566,00 25566,00 33551,00  
 24506,00 22776,00 25925,00  
 31479,00 27641,00 32620,00  
 32772,00 31900,00 27291,00  
 21968,00 21188,00 23239,00  
 22862,00 23677,00 18874,00

*Treinamento*

Saída desejada :

24000,00  
 32527,00  
 32121,00  
 21566,00  
 24506,00  
 31479,00  
 32772,00  
 21968,00  
 22862,00  
 26562,00

Valor de reset : 0,25

Fator de Convexidade : 0,30

Incr. da Taxa de Aprendizado : 0,00

Fator de Decr. da Taxa de Aprendizado : 0,00

Escala : 0,00007

*Execução*

## Previsão do tempo

1 Valor esperado : 25176,000  
 Valor calculado 27617,448 9,698% de erro

2 Valor esperado : 29006,000

	Valor calculado	28131,457	3,015% de erro
3	Valor esperado :	32783,000	
	Valor calculado	28308,953	13,647% de erro
4	Valor esperado :	38834,000	
	Valor calculado	27632,616	28,844% de erro
5	Valor esperado :	28863,000	
	Valor calculado	29171,820	1,070% de erro
6	Valor esperado :	24203,000	
	Valor calculado	24801,409	2,472% de erro
7	Valor esperado :	23196,000	
	Valor calculado	30854,395	33,016% de erro
8	Valor esperado :	20829,000	
	Valor calculado	30590,138	46,863% de erro
9	Valor esperado :	24895,000	
	Valor calculado	24741,935	0,615% de erro
10	Valor esperado :	30195,000	
	Valor calculado	31777,888	5,242% de erro
Fim : 23:34			

### 3.3 - Vendas 3

Início : 22:27

Qtd de camadas : 3

Qtd de neurônios na camada 1: 3

Qtd de neurônios na camada 2: 2

Qtd de neurônios na camada 3: 1

Tamanho da Seqüência (n) : 12

Número de Seqüências : 1

#### Seqüência 1

Input :

26411,00 22899,00 23145,00  
 24000,00 22658,00 29400,00  
 32527,00 31992,00 26592,00  
 32121,00 34068,00 21587,00  
 21566,00 25566,00 33551,00  
 24506,00 22776,00 25925,00  
 31479,00 27641,00 32620,00  
 32772,00 31900,00 27291,00  
 21968,00 21188,00 23239,00  
 22862,00 23677,00 18874,00  
 26562,00 25044,00 26565,00  
 25176,00 29006,00 32783,00

#### Treinamento

Saída desejada :

24000,00  
 32527,00  
 32121,00

21566,00  
 24506,00  
 31479,00  
 32772,00  
 21968,00  
 22862,00  
 26562,00  
 25176,00  
 38834,00

Valor de reset : 0,25  
 Fator de Convexidade : 0,30  
 Escala : 0,00007

180.000 iterações

#### *Seqüência 1*

Esperados	Calculados	
0,4374	0,8221	87,968% de erro
0,9986	0,6907	30,831% de erro
0,9719	0,6992	28,056% de erro
0,2772	0,6986	152,055% de erro
0,4707	0,6987	48,440% de erro
0,9296	0,6982	24,895% de erro
1,0147	0,6987	31,142% de erro
0,3036	0,6987	130,102% de erro
0,3625	0,5936	63,765% de erro
0,6060	0,6290	3,805% de erro
0,5148	0,7033	36,620% de erro
1,4137	0,6984	50,599% de erro

#### *Execução*

##### Previsão do tempo

1	Valor esperado : 23196,000 Valor calculado 27964,996	20,560% de erro
2	Valor esperado : 20829,000 Valor calculado 27964,097	34,256% de erro
3	Valor esperado : 24895,000 Valor calculado 27575,376	10,767% de erro
4	Valor esperado : 30195,000 Valor calculado 27733,012	8,154% de erro
5	Valor esperado : 32095,000 Valor calculado 27963,864	12,872% de erro
6	Valor esperado : 27836,000 Valor calculado 27990,453	0,555% de erro
7	Valor esperado : 30684,000 Valor calculado 27979,996	8,812% de erro
8	Valor esperado : 31414,000 Valor calculado 27964,745	10,980% de erro

9	Valor esperado : 30577,000	
	Valor calculado 27962,994	8,549% de erro
10	Valor esperado : 32047,000	
	Valor calculado 27963,683	12,742% de erro

Fim : 22:40

## 2.4 - Vendas 4

Início : 0:29

Qtd de camadas : 3

Qtd de neurônios na camada 1: 3

Qtd de neurônios na camada 2: 2

Qtd de neurônios na camada 3: 1

Tamanho da Seqüência : 13

Número de Seqüências : 1

### Seqüência 1

Input :

26411,00 22899,00 23145,00  
 24000,00 22658,00 29400,00  
 32527,00 31992,00 26592,00  
 32121,00 34068,00 21587,00  
 21566,00 25566,00 33551,00  
 24506,00 22776,00 25925,00  
 31479,00 27641,00 32620,00  
 32772,00 31900,00 27291,00  
 21968,00 21188,00 23239,00  
 22862,00 23677,00 18874,00  
 26562,00 25044,00 26565,00  
 25176,00 29006,00 32783,00  
 38834,00 28863,00 24203,00

### Treinamento

Saída desejada :

24000,00  
 32527,00  
 32121,00  
 21566,00  
 24506,00  
 31479,00  
 32772,00  
 21968,00  
 22862,00  
 26562,00  
 25176,00  
 38834,00  
 23196,00

Valor de reset : 0,25

Fator de Convexidade : 0,30

Incr. da Taxa de Aprendizado : 0,00

Fator de Decr. da Taxa de Aprendizado : 0,00

Escala : 0,00005

50.000 iterações

*Seqüência 1*

Esperados	Calculados	
0,3568	0,2009	43,693% de erro
0,7840	0,4424	43,570% de erro
0,7637	0,4899	35,856% de erro
0,2349	0,4989	112,424% de erro
0,3822	0,5005	30,955% de erro
0,7315	0,5008	31,535% de erro
0,7963	0,5009	37,099% de erro
0,2550	0,5010	96,452% de erro
0,2998	0,5009	67,077% de erro
0,4852	0,5010	3,255% de erro
0,4157	0,5009	20,489% de erro
1,1000	0,5009	54,463% de erro
0,3165	0,5010	58,268% de erro

*Execução*

Previsão do tempo

1	Valor esperado : 30195,000 Valor calculado 26872,288	11,004% de erro
2	Valor esperado : 32095,000 Valor calculado 26872,295	16,273% de erro
3	Valor esperado : 27836,000 Valor calculado 26873,880	3,456% de erro
4	Valor esperado : 30684,000 Valor calculado 26875,620	12,412% de erro
5	Valor esperado : 31414,000 Valor calculado 26873,457	14,454% de erro
6	Valor esperado : 30577,000 Valor calculado 26874,111	12,110% de erro
7	Valor esperado : 32047,000 Valor calculado 26874,414	16,141% de erro
8	Valor esperado : 25599,000 Valor calculado 26873,866	4,980% de erro
9	Valor esperado : 31813,000 Valor calculado 26876,021	15,519% de erro
10	Valor esperado : 18254,000 Valor calculado 26872,291	47,213% de erro

Fim : 1:34

## 2.5 - Vendas 5

Início : 13:47

Qtd de camadas : 3

Qtd de neurônios na camada 1: 3

Qtd de neurônios na camada 2: 2

Qtd de neurônios na camada 3: 1

Tamanho da Sequência : 15

Número de Sequências : 1

### *Sequência 1*

Input :

26411,00	22899,00	23145,00
24000,00	22658,00	29400,00
32527,00	31992,00	26592,00
32121,00	34068,00	21587,00
21566,00	25566,00	33551,00
24506,00	22776,00	25925,00
31479,00	27641,00	32620,00
32772,00	31900,00	27291,00
21968,00	21188,00	23239,00
22862,00	23677,00	18874,00
26562,00	25044,00	26565,00
25176,00	29006,00	32783,00
38834,00	28863,00	24203,00
23196,00	20829,00	24895,00
30195,00	32095,00	27836,00

### *Treinamento*

Saída desejada :

24000,00
32527,00
32121,00
21566,00
24506,00
31479,00
32772,00
21968,00
22862,00
26562,00
25176,00
38834,00
23196,00
30195,00
30684,00

Valor de reset : 0,25

Coef. Aprend. : 0,20

Momentum : 0,10

Escala : 0,00004

150.000 iterações

## Sequência 1

Esperados	Calculados	
0,3055	0,2689	11,963% de erro
0,6472	0,6989	7,979% de erro
0,6309	0,5917	6,225% de erro
0,2079	0,0801	61,465% de erro
0,3257	0,3924	20,467% de erro
0,6052	0,6534	7,964% de erro
0,6570	0,6530	0,610% de erro
0,2240	0,4299	91,898% de erro
0,2598	0,3437	32,261% de erro
0,4081	0,3940	3,475% de erro
0,3526	0,4167	18,179% de erro
0,9000	0,6819	24,232% de erro
0,2732	0,2773	1,475% de erro
0,5537	0,4163	24,823% de erro
0,5733	0,5754	0,362% de erro

*Execução*

## Previsão do tempo

1	Valor esperado : 32047,000	
	Valor calculado 26723,486	16,612% de erro

Fim : 15: 5



## B.3 - RESULTADO DA IDENTIFICAÇÃO DO ORADOR

### 1 - Energia

#### 1.1 - Treinamento do Padrão

Início : 15: 3  
 Qtd de camadas : 3  
 Qtd de neurônios na camada 1: 5  
 Qtd de neurônios na camada 2: 2  
 Qtd de neurônios na camada 3: 1  
 Tamanho da Sequencia : 18  
 Numero de Sequencias : 1

##### *Sequencia 1*

##### *Input :*

12825,50 101436,17 258073,33 395743,50 550031,83  
 667929,00 719198,17 752337,67 772684,67 791825,50  
 717258,83 547214,17 278427,83 146497,50 119398,00  
 102373,33 113219,83 206523,50 305710,33 425345,83  
 457427,50 299301,17 157623,83 217681,17 310302,33  
 309751,50 306268,67 293631,67 282317,17 182861,17  
 105429,17 74027,33 59699,50 57794,50 58750,17  
 75298,83 124551,17 246361,50 379875,00 631919,67  
 733404,67 775437,33 940977,17 1107535,67 999327,17  
 848694,50 827028,17 820084,50 787678,83 812685,83  
 822602,33 814980,17 834132,00 843032,33 835533,67  
 875732,33 837697,33 764532,33 673351,83 520887,50  
 387442,83 209576,33 130770,50 101455,33 93535,67  
 80136,00 79362,00 96930,00 242876,17 316973,17  
 336687,17 330398,33 324282,83 329834,67 317625,50  
 314299,00 318276,83 304581,17 322281,00 323141,50  
 296745,50 263998,67 234060,33 217225,17 210828,50  
 183961,17 159938,17 135986,83 94529,00 38977,00

##### *Treinamento*

##### *Saída desejada :*

667929,00  
 717258,83  
 102373,33  
 457427,50  
 309751,50  
 105429,17  
 75298,83  
 733404,67  
 848694,50  
 822602,33  
 875732,33  
 387442,83  
 80136,00  
 336687,17

314299,00  
 296745,50  
 183961,17  
 20946,83

Valor de reset : 0,25  
 Fator de Convexidade : 0,30  
 Incr, da Taxa de Aprendizado : 0,20  
 Fator de Decr, da Taxa de Aprendizado : 0,20  
 Escala : 0,00000  
 Funcao de Transf, : Sigmoide

405874 iteracoes

### Sequencia 1

Esperados	Calculados	
667929,0000	857366,5070	28,362% de erro
717258,8333	700268,7625	2,369% de erro
102373,3333	106790,3886	4,315% de erro
457427,5000	425731,6638	6,929% de erro
309751,5000	269712,4477	12,926% de erro
105429,1667	287684,2415	172,870% de erro
75298,8333	85312,2902	13,298% de erro
733404,6667	563229,6849	23,203% de erro
848694,5000	855095,4672	0,754% de erro
822602,3333	671951,7028	18,314% de erro
875732,3333	734231,3620	16,158% de erro
387442,8333	555153,1954	43,286% de erro
80136,0000	57604,9110	28,116% de erro
336687,1667	347691,8416	3,269% de erro
314299,0000	357789,8174	13,837% de erro
296745,5000	344463,3856	16,080% de erro
183961,1667	244228,8339	32,761% de erro
20946,8333	124781,6681	495,707% de erro

Fim : 15: 3

## 1.2 - Verificação do orador

### Caso 1

#### Execução

#### Previsão do tempo

- Valor esperado : 932881,000  
 Valor calculado 1003925,595 7,616% de erro
- Valor esperado : 992983,000  
 Valor calculado 840314,017 15,375% de erro
- Valor esperado : 102839,000  
 Valor calculado 44869,037 56,370% de erro
- Valor esperado : 539077,000

	Valor calculado	604904,502	12,211% de erro
5	Valor esperado	: 324791,000	
	Valor calculado	222891,968	31,374% de erro
6	Valor esperado	: 118238,000	
	Valor calculado	315115,756	166,510% de erro
7	Valor esperado	: 85593,000	
	Valor calculado	99504,564	16,253% de erro
8	Valor esperado	: 885727,000	
	Valor calculado	605106,852	31,682% de erro
9	Valor esperado	: 998806,000	
	Valor calculado	982919,074	1,591% de erro
10	Valor esperado	: 1088284,000	
	Valor calculado	841079,127	22,715% de erro
11	Valor esperado	: 1237937,000	
	Valor calculado	881668,725	28,779% de erro
12	Valor esperado	: 246762,000	
	Valor calculado	649263,488	163,113% de erro
13	Valor esperado	: 98459,000	
	Valor calculado	20092,628	79,593% de erro
14	Valor esperado	: 485021,000	
	Valor calculado	499928,435	3,074% de erro
15	Valor esperado	: 489564,000	
	Valor calculado	488454,793	0,227% de erro
16	Valor esperado	: 405029,000	
	Valor calculado	484363,572	19,587% de erro
17	Valor esperado	: 186201,000	
	Valor calculado	229959,398	23,501% de erro
18	Valor esperado	: 16292,000	
	Valor calculado	123692,801	659,224% de erro

Erro total : 513,0597

Media : 32,0662

Fim : 12:26

## Caso 2

### Execução

Previsão do tempo

1	Valor esperado	: 704755,000	
	Valor calculado	953455,165	35,289% de erro

2	Valor esperado : 728550,000		
	Valor calculado 665730,117	8,623%	de erro
3	Valor esperado : 98120,000		
	Valor calculado 126200,920	28,619%	de erro
4	Valor esperado : 860820,000		
	Valor calculado 394542,515	54,167%	de erro
5	Valor esperado : 277326,000		
	Valor calculado 332974,317	20,066%	de erro
6	Valor esperado : 82712,000		
	Valor calculado 226438,042	173,767%	de erro
7	Valor esperado : 58909,000		
	Valor calculado 100963,687	71,389%	de erro
8	Valor esperado : 698763,000		
	Valor calculado 624883,645	10,573%	de erro
9	Valor esperado : 1168833,000		
	Valor calculado 1005768,635	13,951%	de erro
10	Valor esperado : 1213859,000		
	Valor calculado 872893,326	28,089%	de erro
11	Valor esperado : 1268712,000		
	Valor calculado 921701,820	27,351%	de erro
12	Valor esperado : 480208,000		
	Valor calculado 705644,433	46,946%	de erro
13	Valor esperado : 87826,000		
	Valor calculado 5465,724	93,777%	de erro
14	Valor esperado : 347441,000		
	Valor calculado 316439,795	8,923%	de erro
15	Valor esperado : 303961,000		
	Valor calculado 385184,731	26,722%	de erro
16	Valor esperado : 348462,000		
	Valor calculado 364326,025	4,553%	de erro
17	Valor esperado : 183793,000		
	Valor calculado 242302,930	31,835%	de erro
18	Valor esperado : 15008,000		
	Valor calculado 134915,645	798,958%	de erro

Erro total : 510,8710

Media : 31,9294

Fim : 12:21

**Caso 3***Execução*

## Previsão do tempo

1	Valor esperado : 403807,000 Valor calculado 689142,416	70,661% de erro
2	Valor esperado : 513498,000 Valor calculado 527574,975	2,741% de erro
3	Valor esperado : 71896,000 Valor calculado 155266,911	115,960% de erro
4	Valor esperado : 220762,000 Valor calculado 140267,525	36,462% de erro
5	Valor esperado : 199030,000 Valor calculado 224535,624	12,815% de erro
6	Valor esperado : 152281,000 Valor calculado 324581,746	113,147% de erro
7	Valor esperado : 40276,000 Valor calculado 74746,578	85,586% de erro
8	Valor esperado : 222011,000 Valor calculado 267335,280	20,415% de erro
9	Valor esperado : 424045,000 Valor calculado 516760,121	21,864% de erro
10	Valor esperado : 384351,000 Valor calculado 422506,344	9,927% de erro
11	Valor esperado : 450436,000 Valor calculado 436816,385	3,024% de erro
12	Valor esperado : 518779,000 Valor calculado 519038,329	0,050% de erro
13	Valor esperado : 104389,000 Valor calculado 227526,426	117,960% de erro
14	Valor esperado : 84174,000 Valor calculado 114987,328	36,607% de erro
15	Valor esperado : 211182,000 Valor calculado 317149,178	50,178% de erro
16	Valor esperado : 212451,000 Valor calculado 239402,499	12,686% de erro
17	Valor esperado : 204310,000 Valor calculado 260970,264	27,732% de erro

18 Valor esperado : 106656,000  
 Valor calculado 202844,221 90,185% de erro

Erro total : 624,6702

Media : 39,0419

Fim : 12:21

## Caso 4

### Execução

#### Previsão do tempo

1	Valor esperado : 402735,000	
	Valor calculado 713060,748	77,055% de erro
2	Valor esperado : 642634,000	
	Valor calculado 586865,920	8,678% de erro
3	Valor esperado : 109509,000	
	Valor calculado 312393,265	185,267% de erro
4	Valor esperado : 127005,000	
	Valor calculado 97139,341	23,515% de erro
5	Valor esperado : 178328,000	
	Valor calculado 288283,740	61,659% de erro
6	Valor esperado : 227067,000	
	Valor calculado 360477,870	58,754% de erro
7	Valor esperado : 43813,000	
	Valor calculado 78198,433	78,482% de erro
8	Valor esperado : 236039,000	
	Valor calculado 251892,136	6,716% de erro
9	Valor esperado : 620064,000	
	Valor calculado 641342,934	3,432% de erro
10	Valor esperado : 701584,000	
	Valor calculado 664396,709	5,300% de erro
11	Valor esperado : 700303,000	
	Valor calculado 644561,207	7,960% de erro
12	Valor esperado : 686411,000	
	Valor calculado 636475,888	7,275% de erro
13	Valor esperado : 131984,000	
	Valor calculado 454090,156	244,049% de erro
14	Valor esperado : 73339,000	
	Valor calculado 48016,691	34,528% de erro
15	Valor esperado : 256230,000	

	Valor calculado	318549,623	24,322% de erro
16	Valor esperado	: 223859,000	
	Valor calculado	288711,961	28,970% de erro
17	Valor esperado	: 246509,000	
	Valor calculado	281400,094	14,154% de erro
18	Valor esperado	: 235805,000	
	Valor calculado	309603,259	31,296% de erro

Erro total : 811,3631

Media : 50,7102

Fim : 12:22

## Caso 5

### Execução

#### Previsão do tempo

1	Valor esperado	: 445115,000	
	Valor calculado	706061,770	58,625% de erro
2	Valor esperado	: 793092,000	
	Valor calculado	667658,107	15,816% de erro
3	Valor esperado	: 97299,000	
	Valor calculado	295186,306	203,381% de erro
4	Valor esperado	: 227515,000	
	Valor calculado	104732,708	53,967% de erro
5	Valor esperado	: 280068,000	
	Valor calculado	386619,690	38,045% de erro
6	Valor esperado	: 345495,000	
	Valor calculado	353613,396	2,350% de erro
7	Valor esperado	: 66169,000	
	Valor calculado	168385,191	154,477% de erro
8	Valor esperado	: 250776,000	
	Valor calculado	179135,820	28,567% de erro
9	Valor esperado	: 696963,000	
	Valor calculado	772236,644	10,800% de erro
10	Valor esperado	: 1067772,000	
	Valor calculado	939821,792	11,983% de erro
11	Valor esperado	: 917599,000	
	Valor calculado	776039,279	15,427% de erro
12	Valor esperado	: 798679,000	
	Valor calculado	730310,823	8,560% de erro

13	Valor esperado : 584269,000		
	Valor calculado 576875,466	1,265%	de erro
14	Valor esperado : 93679,000		
	Valor calculado 72383,629	22,732%	de erro
15	Valor esperado : 370701,000		
	Valor calculado 295868,944	20,187%	de erro
16	Valor esperado : 438499,000		
	Valor calculado 459672,808	4,829%	de erro
17	Valor esperado : 609504,000		
	Valor calculado 582900,627	4,365%	de erro
18	Valor esperado : 465993,000		
	Valor calculado 469541,356	0,761%	de erro

Erro total : 653,0258

Media : 40,8141

Fim : 12:22

## Caso 6

### Execução

#### Previsão do tempo

1	Valor esperado : 469680,000		
	Valor calculado 804444,451	71,275%	de erro
2	Valor esperado : 125028,000		
	Valor calculado 294024,581	135,167%	de erro
3	Valor esperado : 313699,000		
	Valor calculado 132072,261	57,898%	de erro
4	Valor esperado : 83431,000		
	Valor calculado 446235,602	434,856%	de erro
5	Valor esperado : 236308,000		
	Valor calculado 271685,353	14,971%	de erro
6	Valor esperado : 70337,000		
	Valor calculado 134748,336	91,575%	de erro
7	Valor esperado : 277213,000		
	Valor calculado 202635,746	26,903%	de erro
8	Valor esperado : 1053123,000		
	Valor calculado 757588,922	28,063%	de erro
9	Valor esperado : 1028070,000		
	Valor calculado 922475,842	10,271%	de erro



10	Valor esperado : 1055002,000		
	Valor calculado	842696,558	20,124% de erro
11	Valor esperado : 978900,000		
	Valor calculado	874980,587	10,616% de erro
12	Valor esperado : 93526,000		
	Valor calculado	119133,816	27,380% de erro
13	Valor esperado : 382927,000		
	Valor calculado	178363,479	53,421% de erro
14	Valor esperado : 396953,000		
	Valor calculado	526970,855	32,754% de erro
15	Valor esperado : 427354,000		
	Valor calculado	363458,816	14,951% de erro
16	Valor esperado : 145067,000		
	Valor calculado	253694,560	74,881% de erro
17	Valor esperado : 19833,000		
	Valor calculado	83068,297	318,839% de erro
18	Valor esperado : 465993,000		
	Valor calculado	599273,774	28,601% de erro

Erro total : 1332,3695

Media : 83,2731

Fim : 12:22

## Caso 7

### Execução

#### Previsão do tempo

1	Valor esperado : 469411,000		
	Valor calculado	730550,322	55,631% de erro
2	Valor esperado : 736221,000		
	Valor calculado	675784,275	8,209% de erro
3	Valor esperado : 125434,000		
	Valor calculado	402254,987	220,691% de erro
4	Valor esperado : 59809,000		
	Valor calculado	66423,183	11,059% de erro
5	Valor esperado : 94200,000		
	Valor calculado	409041,717	334,227% de erro
6	Valor esperado : 344292,000		
	Valor calculado	291885,650	15,221% de erro
7	Valor esperado : 70907,000		

	Valor calculado	210221,947	196,476% de erro
8	Valor esperado :	234213,000	
	Valor calculado	150906,074	35,569% de erro
9	Valor esperado :	548586,000	
	Valor calculado	639703,439	16,610% de erro
10	Valor esperado :	529317,000	
	Valor calculado	510410,453	3,572% de erro
11	Valor esperado :	553105,000	
	Valor calculado	520837,808	5,834% de erro
12	Valor esperado :	510398,000	
	Valor calculado	514287,961	0,762% de erro
13	Valor esperado :	426087,000	
	Valor calculado	457197,372	7,301% de erro
14	Valor esperado :	69446,000	
	Valor calculado	102332,570	47,356% de erro
15	Valor esperado :	241834,000	
	Valor calculado	219237,019	9,344% de erro
16	Valor esperado :	263293,000	
	Valor calculado	339326,620	28,878% de erro
17	Valor esperado :	271909,000	
	Valor calculado	300613,955	10,557% de erro
18	Valor esperado :	262450,000	
	Valor calculado	327295,642	24,708% de erro

Erro total : 992,0742

Media : 62,0046

Fim : 12:22

## Caso 8

### Execução

#### Previsão do tempo

1	Valor esperado :	278029,000	
	Valor calculado	601295,480	116,271% de erro
2	Valor esperado :	637853,000	
	Valor calculado	557891,578	12,536% de erro
3	Valor esperado :	113131,000	
	Valor calculado	384486,740	239,860% de erro
4	Valor esperado :	85273,000	
	Valor calculado	70919,926	16,832% de erro

5	Valor esperado : 110756,000		
	Valor calculado 339641,220	206,657%	de erro
6	Valor esperado : 314215,000		
	Valor calculado 303285,624	3,478%	de erro
7	Valor esperado : 43161,000		
	Valor calculado 113347,904	162,616%	de erro
8	Valor esperado : 131964,000		
	Valor calculado 138486,515	4,943%	de erro
9	Valor esperado : 433927,000		
	Valor calculado 629407,770	45,049%	de erro
10	Valor esperado : 772597,000		
	Valor calculado 613916,959	20,539%	de erro
11	Valor esperado : 996147,000		
	Valor calculado 822880,798	17,394%	de erro
12	Valor esperado : 742046,000		
	Valor calculado 829555,593	11,793%	de erro
13	Valor esperado : 90181,000		
	Valor calculado 356032,283	294,797%	de erro
14	Valor esperado : 66916,000		
	Valor calculado 75406,137	12,688%	de erro
15	Valor esperado : 273002,000		
	Valor calculado 325907,267	19,379%	de erro
16	Valor esperado : 251095,000		
	Valor calculado 342911,433	36,566%	de erro
17	Valor esperado : 247741,000		
	Valor calculado 272790,079	10,111%	de erro
18	Valor esperado : 248227,000		
	Valor calculado 315631,996	27,155%	de erro

Erro total : 1228,0309

Media : 76,7519

Fim : 12:22

## Caso 9

### Execução

#### Previsão do tempo

1	Valor esperado : 614269,000		
	Valor calculado 841414,814	36,978%	de erro

2	Valor esperado : 1002074,000		
	Valor calculado 796392,206	20,526%	de erro
3	Valor esperado : 192710,000		
	Valor calculado 304463,335	57,990%	de erro
4	Valor esperado : 472029,000		
	Valor calculado 177951,209	62,301%	de erro
5	Valor esperado : 369008,000		
	Valor calculado 378634,112	2,609%	de erro
6	Valor esperado : 523703,000		
	Valor calculado 547838,016	4,609%	de erro
7	Valor esperado : 83873,000		
	Valor calculado 162292,957	93,498%	de erro
8	Valor esperado : 272390,000		
	Valor calculado 195467,007	28,240%	de erro
9	Valor esperado : 872200,000		
	Valor calculado 872344,973	0,017%	de erro
10	Valor esperado : 760372,000		
	Valor calculado 739431,410	2,754%	de erro
11	Valor esperado : 662634,000		
	Valor calculado 623662,915	5,881%	de erro
12	Valor esperado : 683231,000		
	Valor calculado 622612,127	8,872%	de erro
13	Valor esperado : 550290,000		
	Valor calculado 589054,974	7,044%	de erro
14	Valor esperado : 100167,000		
	Valor calculado 74613,643	25,511%	de erro
15	Valor esperado : 407080,000		
	Valor calculado 345460,759	15,137%	de erro
16	Valor esperado : 401562,000		
	Valor calculado 427759,426	6,524%	de erro
17	Valor esperado : 321794,000		
	Valor calculado 357489,149	11,093%	de erro
18	Valor esperado : 234522,000		
	Valor calculado 329828,169	40,638%	de erro

Erro total : 384,9749

Media : 24,0609

Fim : 12:22

**Caso 10***Execução*

## Previsão do tempo

1	Valor esperado : 456488,000		
	Valor calculado 685438,811	50,155%	de erro
2	Valor esperado : 214493,000		
	Valor calculado 481185,397	124,336%	de erro
3	Valor esperado : 322105,000		
	Valor calculado 296014,685	8,100%	de erro
4	Valor esperado : 390028,000		
	Valor calculado 752435,688	92,918%	de erro
5	Valor esperado : 181295,000		
	Valor calculado 33874,417	81,315%	de erro
6	Valor esperado : 1057860,000		
	Valor calculado 821522,530	22,341%	de erro
7	Valor esperado : 912739,000		
	Valor calculado 797881,887	12,584%	de erro
8	Valor esperado : 887033,000		
	Valor calculado 721044,327	18,713%	de erro
9	Valor esperado : 957113,000		
	Valor calculado 757175,343	20,890%	de erro
10	Valor esperado : 561960,000		
	Valor calculado 648899,649	15,471%	de erro
11	Valor esperado : 107856,000		
	Valor calculado 47622,259	55,846%	de erro
12	Valor esperado : 535292,000		
	Valor calculado 512767,211	4,208%	de erro
13	Valor esperado : 295509,000		
	Valor calculado 371239,976	25,627%	de erro
14	Valor esperado : 210086,000		
	Valor calculado 291871,679	38,930%	de erro
15	Valor esperado : 14021,000		
	Valor calculado 87957,746	527,329%	de erro
16	Valor esperado : 18462,000		
	Valor calculado 112461,589	509,152%	de erro
17	Valor esperado : 18462,000		
	Valor calculado 107275,491	481,061%	de erro
18	Valor esperado : 18462,000		

Valor calculado 108702,784 488,792% de erro

Erro total : 2066,6344

Media : 129,1646

Fim : 14:18

## Caso 11

### Execução

#### Previsão do tempo

1	Valor esperado : 380805,000		
	Valor calculado 790086,108	107,478%	de erro
2	Valor esperado : 365441,000		
	Valor calculado 163292,498	55,316%	de erro
3	Valor esperado : 473719,000		
	Valor calculado 431533,433	8,905%	de erro
4	Valor esperado : 134370,000		
	Valor calculado 201933,571	50,282%	de erro
5	Valor esperado : 765654,000		
	Valor calculado 492230,631	35,711%	de erro
6	Valor esperado : 858689,000		
	Valor calculado 860714,738	0,236%	de erro
7	Valor esperado : 963201,000		
	Valor calculado 757149,592	21,392%	de erro
8	Valor esperado : 987848,000		
	Valor calculado 775468,784	21,499%	de erro
9	Valor esperado : 952565,000		
	Valor calculado 815361,978	14,404%	de erro
10	Valor esperado : 196116,000		
	Valor calculado 599118,512	205,492%	de erro
11	Valor esperado : 243773,000		
	Valor calculado 32055,085	86,850%	de erro
12	Valor esperado : 296830,000		
	Valor calculado 586831,808	97,700%	de erro
13	Valor esperado : 36362,000		
	Valor calculado 17763,051	51,149%	de erro
14	Valor esperado : 25937,000		
	Valor calculado 141167,401	444,270%	de erro
15	Valor esperado : 25937,000		

	Valor calculado	106573,726	310,895% de erro
16	Valor esperado :	25937,000	
	Valor calculado	116691,780	349,905% de erro
17	Valor esperado :	25937,000	
	Valor calculado	113706,794	338,396% de erro
18	Valor esperado :	25937,000	
	Valor calculado	114588,343	341,795% de erro

Erro total : 2199,6444

Media : 137,4778

Fim : 14:21

## Caso 12

### Execução

#### Previsão do tempo

1	Valor esperado :	753096,000	
	Valor calculado	931252,523	23,657% de erro
2	Valor esperado :	183922,000	
	Valor calculado	305911,043	66,327% de erro
3	Valor esperado :	388902,000	
	Valor calculado	551830,887	41,895% de erro
4	Valor esperado :	276578,000	
	Valor calculado	596590,570	115,704% de erro
5	Valor esperado :	365208,000	
	Valor calculado	233128,599	36,166% de erro
6	Valor esperado :	1046517,000	
	Valor calculado	922602,105	11,841% de erro
7	Valor esperado :	913118,000	
	Valor calculado	785631,683	13,962% de erro
8	Valor esperado :	972743,000	
	Valor calculado	819267,560	15,778% de erro
9	Valor esperado :	1018021,000	
	Valor calculado	846876,103	16,812% de erro
10	Valor esperado :	872770,000	
	Valor calculado	824347,770	5,548% de erro
11	Valor esperado :	116923,000	
	Valor calculado	209717,658	79,364% de erro
12	Valor esperado :	313720,000	
	Valor calculado	159373,109	49,199% de erro

13	Valor esperado : 475063,000		
	Valor calculado 668833,843	40,788%	de erro
14	Valor esperado : 283344,000		
	Valor calculado 249339,211	12,001%	de erro
15	Valor esperado : 20247,000		
	Valor calculado 186867,065	822,937%	de erro
16	Valor esperado : 18300,000		
	Valor calculado 79520,252	334,537%	de erro
17	Valor esperado : 18300,000		
	Valor calculado 116999,010	539,339%	de erro
18	Valor esperado : 18300,000		
	Valor calculado 105663,568	477,397%	de erro

Erro total : 2214,0117

Media : 138,3757

Fim : 14:22

### Caso 13

#### Execução

#### Previsão do tempo

1	Valor esperado : 282625,000		
	Valor calculado 639131,339	126,141%	de erro
2	Valor esperado : 304950,000		
	Valor calculado 350515,847	14,942%	de erro
3	Valor esperado : 67499,000		
	Valor calculado 108287,981	60,429%	de erro
4	Valor esperado : 142740,000		
	Valor calculado 335765,088	135,228%	de erro
5	Valor esperado : 166510,000		
	Valor calculado 274519,539	64,867%	de erro
6	Valor esperado : 95795,000		
	Valor calculado 119545,824	24,793%	de erro
7	Valor esperado : 263946,000		
	Valor calculado 353903,629	34,082%	de erro
8	Valor esperado : 273815,000		
	Valor calculado 330066,550	20,544%	de erro
9	Valor esperado : 60721,000		
	Valor calculado 193877,554	219,292%	de erro



10	Valor esperado : 49906,000		
	Valor calculado	98454,269	97,279% de erro
11	Valor esperado : 91429,000		
	Valor calculado	190507,524	108,367% de erro
12	Valor esperado : 57429,000		
	Valor calculado	145862,142	153,987% de erro
13	Valor esperado : 33058,000		
	Valor calculado	115281,384	248,725% de erro
14	Valor esperado : 14789,000		
	Valor calculado	105141,460	610,944% de erro
15	Valor esperado : 58300,000		
	Valor calculado	150395,815	157,969% de erro
16	Valor esperado : 58300,000		
	Valor calculado	137651,286	136,109% de erro
17	Valor esperado : 58300,000		
	Valor calculado	141425,336	142,582% de erro
18	Valor esperado : 58300,000		
	Valor calculado	140297,631	140,648% de erro

Erro total : 2331,4859

Media : 145,7179

Fim : 14:26

## Caso 14

### Execução

#### Previsão do tempo

1	Valor esperado : 214196,000		
	Valor calculado	577919,493	169,809% de erro
2	Valor esperado : 64939,000		
	Valor calculado	212853,474	227,774% de erro
3	Valor esperado : 192585,000		
	Valor calculado	116678,715	39,414% de erro
4	Valor esperado : 215767,000		
	Valor calculado	265489,640	23,045% de erro
5	Valor esperado : 57034,000		
	Valor calculado	207322,987	263,508% de erro
6	Valor esperado : 21124,000		
	Valor calculado	90694,997	329,346% de erro
7	Valor esperado : 110630,000		

	Valor calculado	131162,204	18,559% de erro
8	Valor esperado :	340753,000	
	Valor calculado	411825,764	20,858% de erro
9	Valor esperado :	420519,000	
	Valor calculado	455605,660	8,344% de erro
10	Valor esperado :	374980,000	
	Valor calculado	443699,316	18,326% de erro
11	Valor esperado :	373582,000	
	Valor calculado	375475,507	0,507% de erro
12	Valor esperado :	201287,000	
	Valor calculado	328786,601	63,342% de erro
13	Valor esperado :	56509,000	
	Valor calculado	99092,664	75,357% de erro
14	Valor esperado :	42380,000	
	Valor calculado	137911,395	225,416% de erro
15	Valor esperado :	88878,000	
	Valor calculado	159432,106	79,383% de erro
16	Valor esperado :	98300,000	
	Valor calculado	170518,084	73,467% de erro
17	Valor esperado :	92250,000	
	Valor calculado	199130,311	115,859% de erro
18	Valor esperado :	37092,000	
	Valor calculado	88845,389	139,527% de erro

Erro total : 1422,9688

Media : 88,9355

Fim : 14:26

## Caso 15

### Execução

#### Previsão do tempo

1	Valor esperado :	65625,000	
	Valor calculado	494568,990	653,629% de erro
2	Valor esperado :	50593,000	
	Valor calculado	207602,742	310,339% de erro
3	Valor esperado :	98391,000	
	Valor calculado	224981,442	128,661% de erro
4	Valor esperado :	42306,000	
	Valor calculado	104819,808	147,766% de erro

5	Valor esperado : 56771,000		
	Valor calculado 122234,368	115,311%	de erro
6	Valor esperado : 320196,000		
	Valor calculado 326456,331	1,955%	de erro
7	Valor esperado : 366624,000		
	Valor calculado 383291,884	4,546%	de erro
8	Valor esperado : 347156,000		
	Valor calculado 376531,620	8,462%	de erro
9	Valor esperado : 371381,000		
	Valor calculado 389730,651	4,941%	de erro
10	Valor esperado : 284613,000		
	Valor calculado 349836,897	22,917%	de erro
11	Valor esperado : 57997,000		
	Valor calculado 155557,669	168,217%	de erro
12	Valor esperado : 52855,000		
	Valor calculado 128747,826	143,587%	de erro
13	Valor esperado : 104836,000		
	Valor calculado 186233,444	77,643%	de erro
14	Valor esperado : 117203,000		
	Valor calculado 182016,975	55,301%	de erro
15	Valor esperado : 79021,000		
	Valor calculado 182750,171	131,268%	de erro
16	Valor esperado : 15263,000		
	Valor calculado 99609,286	552,619%	de erro
17	Valor esperado : 6050,000		
	Valor calculado 97434,130	1510,481%	de erro
18	Valor esperado : 6050,000		
	Valor calculado 98773,443	1532,619%	de erro

Erro total : 4035,6867

Media : 252,2304

Fim : 14:28

## Caso 16

### Execução

#### Previsão do tempo

1	Valor esperado : 181896,000		
	Valor calculado 546911,006	200,672%	de erro

2	Valor esperado : 47994,000		
	Valor calculado 112551,016	134,511%	de erro
3	Valor esperado : 347969,000		
	Valor calculado 139292,947	59,970%	de erro
4	Valor esperado : 157488,000		
	Valor calculado 241104,916	53,094%	de erro
5	Valor esperado : 44764,000		
	Valor calculado 165354,992	269,393%	de erro
6	Valor esperado : 14860,000		
	Valor calculado 94199,013	533,910%	de erro
7	Valor esperado : 377313,000		
	Valor calculado 319891,463	15,219%	de erro
8	Valor esperado : 438833,000		
	Valor calculado 522481,140	19,061%	de erro
9	Valor esperado : 324561,000		
	Valor calculado 321674,466	0,889%	de erro
10	Valor esperado : 252718,000		
	Valor calculado 362521,481	43,449%	de erro
11	Valor esperado : 161416,000		
	Valor calculado 312890,941	93,841%	de erro
12	Valor esperado : 48811,000		
	Valor calculado 88535,987	81,385%	de erro
13	Valor esperado : 130866,000		
	Valor calculado 128300,931	1,960%	de erro
14	Valor esperado : 270098,000		
	Valor calculado 347055,252	28,492%	de erro
15	Valor esperado : 104864,000		
	Valor calculado 188340,540	79,605%	de erro
16	Valor esperado : 26680,000		
	Valor calculado 113432,370	325,159%	de erro
17	Valor esperado : 18850,000		
	Valor calculado 107234,310	468,882%	de erro
18	Valor esperado : 18850,000		
	Valor calculado 109125,769	478,917%	de erro

Erro total : 1875,5828

Media : 117,2239

Fim : 14:39

**Caso 16***Execução***Previsão do tempo**

1	Valor esperado : 228586,000		
	Valor calculado 631227,716	176,145%	de erro
2	Valor esperado : 53487,000		
	Valor calculado 124295,381	132,384%	de erro
3	Valor esperado : 113595,000		
	Valor calculado 124403,310	9,515%	de erro
4	Valor esperado : 204144,000		
	Valor calculado 252802,420	23,835%	de erro
5	Valor esperado : 39050,000		
	Valor calculado 166471,420	326,303%	de erro
6	Valor esperado : 25708,000		
	Valor calculado 97275,587	278,386%	de erro
7	Valor esperado : 413308,000		
	Valor calculado 323506,700	21,727%	de erro
8	Valor esperado : 471427,000		
	Valor calculado 643486,558	36,498%	de erro
9	Valor esperado : 470682,000		
	Valor calculado 434845,895	7,614%	de erro
10	Valor esperado : 541178,000		
	Valor calculado 491785,060	9,127%	de erro
11	Valor esperado : 87816,000		
	Valor calculado 88489,361	0,767%	de erro
12	Valor esperado : 39578,000		
	Valor calculado 150615,058	280,552%	de erro
13	Valor esperado : 269922,000		
	Valor calculado 363256,950	34,578%	de erro
14	Valor esperado : 262383,000		
	Valor calculado 297629,036	13,433%	de erro
15	Valor esperado : 82572,000		
	Valor calculado 232508,042	181,582%	de erro
16	Valor esperado : 19855,000		
	Valor calculado 82856,051	317,306%	de erro
17	Valor esperado : 18800,000		
	Valor calculado 115326,333	513,438%	de erro
18	Valor esperado : 18800,000		

Valor calculado 106733,719 467,733% de erro

Erro total : 2084,8045

Media : 130,3003

Fim : 14:40

## Caso 17

### Execução

#### Previsão do tempo

1	Valor esperado : 50735,000		
	Valor calculado 494088,770	873,862%	de erro
2	Valor esperado : 413735,000		
	Valor calculado 108497,357	73,776%	de erro
3	Valor esperado : 139507,000		
	Valor calculado 275530,916	97,503%	de erro
4	Valor esperado : 26019,000		
	Valor calculado 112540,951	332,534%	de erro
5	Valor esperado : 30934,000		
	Valor calculado 101575,944	228,363%	de erro
6	Valor esperado : 601934,000		
	Valor calculado 558149,626	7,274%	de erro
7	Valor esperado : 440553,000		
	Valor calculado 470588,097	6,818%	de erro
8	Valor esperado : 356674,000		
	Valor calculado 356426,704	0,069%	de erro
9	Valor esperado : 120290,000		
	Valor calculado 351337,535	192,075%	de erro
10	Valor esperado : 53966,000		
	Valor calculado 74504,330	38,058%	de erro
11	Valor esperado : 33927,000		
	Valor calculado 136439,691	302,157%	de erro
12	Valor esperado : 250828,000		
	Valor calculado 377201,737	50,383%	de erro
13	Valor esperado : 179853,000		
	Valor calculado 267392,660	48,673%	de erro
14	Valor esperado : 42211,000		
	Valor calculado 125042,197	196,231%	de erro
15	Valor esperado : 13561,000		
	Valor calculado 102050,223	652,527%	de erro

16	Valor esperado : 18410,000	
	Valor calculado 109943,585	497,195% de erro
17	Valor esperado : 18410,000	
	Valor calculado 107880,061	485,986% de erro
18	Valor esperado : 18410,000	
	Valor calculado 108474,811	489,217% de erro

Erro total : 4076,2105

Media : 254,7632

Fim : 14:41

## 2 - Estimativa da Frequência Fundamental

### 1 - Treinamento do Padrão

Início : 15: 0

Qtd de camadas : 3

Qtd de neurônios na camada 1: 5

Qtd de neurônios na camada 2: 2

Qtd de neurônios na camada 3: 1

Tamanho da Sequencia : 17

Numero de Sequencias : 1

#### *Sequencia 1*

Input :

26,48	31,17	32,05	26,86	35,17
35,48	34,66	34,72	30,99	34,71
35,09	36,56	31,33	37,00	35,70
33,73	40,84	41,92	39,46	43,14
44,61	43,67	44,32	42,09	43,50
40,75	43,87	41,12	42,41	42,52
45,08	48,25	44,46	47,62	44,45
46,32	47,30	49,55	51,37	53,65
55,11	56,48	56,02	58,66	59,75
60,28	59,78	59,79	60,85	61,95
62,19	62,29	61,80	62,63	62,27
61,97	65,66	60,95	65,90	63,06
65,16	66,95	66,35	66,55	66,48
66,51	66,50	65,52	67,48	67,00
65,24	65,82	67,00	66,98	66,99
66,54	66,99	66,99	64,15	65,91
66,99	66,15	66,99	66,99	67,29

#### *Treinamento*

Saida desejada :

35,48  
35,09  
33,73  
44,61

40,75  
 45,08  
 46,32  
 55,11  
 60,28  
 62,19  
 61,97  
 65,16  
 66,51  
 65,24  
 66,54  
 66,99  
 65,91

Valor de reset : 0,25  
 Fator de Convexidade : 0,30  
 Incr, da Taxa de Aprendizado : 0,20  
 Fator de Decr, da Taxa de Aprendizado : 0,20  
 Escala : 0,00488  
 Funcao de Transf, : Sigmoide

438176 iteracoes

*Sequencia 1*

Esperados	Calculados	
35,4761	110,6218	211,821% de erro
35,0943	15,9886	54,441% de erro
33,7282	44,6087	32,259% de erro
44,6127	44,4784	0,301% de erro
40,7459	47,2339	15,923% de erro
45,0790	44,3242	1,674% de erro
46,3212	49,1573	6,123% de erro
55,1084	55,9356	1,501% de erro
60,2822	61,0054	1,200% de erro
62,1932	62,1973	0,007% de erro
61,9676	62,7764	1,305% de erro
65,1637	63,3333	2,809% de erro
66,5054	66,3216	0,276% de erro
65,2362	65,8300	0,910% de erro
66,5391	65,9838	0,834% de erro
66,9873	64,9796	2,997% de erro
65,9096	66,5990	1,046% de erro

Fim : 15: 0



**APÊNDICE C****ESPECIFICAÇÃO DO EQUIPAMENTO ENVOLVIDO****A) Microcomputador :**

IBM-PC compatível AT-386,  
40 MHz,  
4 Mb de memória RAM  
128 Kb de memória cache

**B) Placa Digitalizadora :**

Sound Blaster 16  
Frequência de Amostragem : 11 kHz  
Resolução : 16 bits  
Gravação Mono

**C) Microfone :**

Le Son, MK2 - Plus,  
dinâmico,  
direcional,  
impedância de 300  $\Omega$

## APÊNDICE D

### FORMATO DOS ARQUIVOS DE DADOS

A seguir será fornecido o formato dos arquivos gerados pelo software de aquisição de sinal de voz VREC, de modo que sejam possíveis futuros desenvolvimentos compatíveis com o tipo de entrada utilizado neste trabalho.

Os arquivos de extensão .VOC são divididos em dois blocos principais : o bloco de cabeçalho e o bloco de dados. Este último é ainda constituído por oito tipos de sub-blocos.

#### Bloco de Cabeçalho

Bytes	Descrição
0 - 13	Descritor do tipo de arquivo. Contém a seguinte mensagem : "Creative Voice File", 1Ah.
14-15	Deslocamento do bloco de Dados, a partir do início do arquivo.
16-17	Número do Formato do Arquivo de Voz. Versão do formato, no caso, 1,20
18-19	Código de Identificação do Arquivo de Voz. Contém o complemento da versão mais o valor 1234h (hexadecimal).

#### Bloco de Dados

Bytes	Descrição
0	O primeiro byte de cada sub-bloco define o tipo de dado guardado ali.
1-3	Os próximos três bytes definem o comprimento do sub-bloco, excluindo os quatro que contêm o tipo e o comprimento.

#### Tipos de Dados

0	Bloco Terminador
1	Dados de Voz
2	Continuação
3	Silêncio

- 4 Marca
- 5 Texto ASCII
- 6 Início de "Loop" de repetição
- 7 Fim de "Loop" de repetição
- 8 Bloco Estendido (Informações Adicionais)

Os demais arquivos de entrada de dados utilizados neste trabalho possuem os seguintes formatos :

Valores das Séries Temporais e de Voz : cada valor ocupa um registro de 6 bytes, sequencialmente, para cada unidade de tempo.

Definição da Série para Treinamento : arquivo texto, ASCII.

- Linha 1 : Quantidade de camadas, N.
- Linha 2 até N+1 : Quantidade de neurônios na camada
- Linha N+2 : Quantidade de seqüências
- Linha N+3 : Número de termos na série