# Adaptive Pattern Classification

*by A. Teolis*
*Advisor: S. Shamma*

M.S. 89-6
*Formerly TR 89-107*

# Adaptive Pattern Classification

by

**Anthony Teolis**

Thesis submitted to the Faculty of The Graduate School
of The University Of Maryland in partial fulfillment
of the requirements for the degree of
Master of Science
1989

Advisory Committee:

| | |
|---|---|
| Associate Professor Shihab Shamma | Chairman/Advisor |
| Professor Martin C. Peckerar | |
| Professor Andre Tits | |

# ABSTRACT

| | |
|---|---|
| Title of Thesis: | Adaptive Pattern Classification |
| Name of Degree Canidate: | Anthony Teolis, Jr. |
| Degree and Year: | Master of Science, 1989 |
| Thesis directed by: | Associate Professor Shihab Shamma |
| | Systems Research Center |
| | Dept. of Electrical Engineering |

Up until the recent past, the power of multi layer feed forward artificial neural networks has been untapped mainly due to the lack of algorithms to train them. With the emergence of the backpropagation algorithm; however, this deficiency has been removed. Despite this innovation, the backpropagation method is still not without its drawbacks. Among these the most prominent are the facts that i) the learning is conducted in a supervised manner and ii) that learning and operation must occur in two distinct phases.

Because of these properties, the backpropagation algorithm falls short of solving a 'true' pattern classification problem. This is not to say that a network

could not be trained via backpropagation to mimic a previously solved pattern classification scheme; but that the backpropagation method is not capable of autonomously generating classification schemes.

A more realistic (and certainly more useful) learning scenario is that patterns would be presented without supervision to the system continuously; consequently, the system will begin to group like patterns into similar classes and continue to do so indefinitely; i.e. continuous learning. It is exactly this type of learning that is discussed here.

# DEDICATION

To my parents

# ACKNOWLEDGMENTS

I am most endebted to my advisor Dr. Shihab Shamma. I have benefited greatly from the many discussions on which most of the work presented in this thesis is based. His thoughtful insights and understanding have not gone unappreciated. For use of the resources of the intelligent servo-mechanisms labratory I am endebted to Dr. P.S. Krishnaprasad who has shown a constant interest in my progress.

In addition to sitting on my thesis committee, Dr. Martin Peckerar deserves my sincerest gratitude for his support and encouragment throughout the preparation of this thesis.

I would also like to thank Dr. Andre Tits for serving as a member of my thesis committe.

On a more personal note, I would like to thank my fiancé and colleague Carole Salter who provided a constant source of both technical and emotional support.

# TABLE OF CONTENTS

# LIST OF TABLES

# LIST OF FIGURES

# NOTATION

Some common notation found throughout this thesis is presented below for easy reference. Let $x, y \in \mathbb{R}^n$, $z \in \mathbb{R}^m$ and $A \in \mathbb{R}^{n \times n}$, where $A = (a_{ij})_{i,j=1}^n$, and $y = (y_1\ y_2\ \cdots,\ y_n\ )^T$. Also let $f(\cdot) : \mathbb{R}^n \mapsto \mathbb{R}$ and $g(\cdot,\cdot) : \mathbb{R}^n \times \mathbb{R}^m \mapsto \mathbb{R}$

| | | |
|---|---|---|
| $\mathrm{diag}(a_1,\ a_2,\ \cdots,\ a_n)$ | $\begin{pmatrix} a_1 & & & 0 \\ & a_2 & & \\ & & \ddots & \\ 0 & & & a_n \end{pmatrix}$ | diagonal matrix |
| $A^T$ | $A^T = (a_{ji})_{i,j=1}^n$ | transpose |
| $tr(A)$ | $tr(A) \triangleq \sum_{i=1}^n a_{ii}$ | trace of $A$ |
| $\sigma(A)$ | $\lambda \in \sigma(A) \Longleftrightarrow \exists\ x \in \mathbb{R}^n, x \neq 0 \ni Ax = \lambda x$ | spectrum of $A$ |
| $\mathcal{N}(A)$ | $\mathcal{N}(A) \triangleq \{x \in \mathbb{R}^n : Ax = 0\}$ | null space of $A$ |

| | | |
|---|---|---|
| $\|y\|_p$ | $\|y\|_p \triangleq (\sum_{i=1}^n \|y_i\|^p)^{\frac{1}{p}}$ | $\ell_p$ norm |
| $\bar{y}$ | $\bar{y} \triangleq \frac{y}{\|y\|_2}$ | normalization |
| $\nabla_x f$ | $\nabla_x f \triangleq \left( \frac{\partial f(x)}{\partial x_1} \cdots \frac{\partial f(x)}{\partial x_n} \right)^T$ | gradient with respect to $x$ |
| $\nabla_x g$ | $\nabla_x g \triangleq \left( \frac{\partial g(x,z)}{\partial x_1} \cdots \frac{\partial g(x,z)}{\partial x_n} \right)^T$ | partial gradient with respect to $x$ |
| $\langle x, y \rangle$ | $\langle x, y \rangle \triangleq \sum_{i=1}^n x_i y_i$ | inner product |
| $x \parallel y$ | $\exists\ \alpha \in \mathbb{R} \ni\ y = \alpha x$ | $x$ parallel to $y$ |
| $x \perp y$ | $\langle x, y \rangle = 0$ | $x$ orthogonal to $y$ |

CHAPTER
# ONE
## Introduction

There is little doubt that the ultimate goal of automated machine perception is an important one. Yet, feats of perception that are effortlessly carried out by biological means are not directly amenable to machine implementation. Already, the discipline of neural networks is striving to provide architectures for cognition that are biologically realistic. It is through the study of the biological nervous system that the most promising models of perception should emerge.

In short, the idea of the neural network discipline is to understand, and hence mimic, the processing done in biological nervous systems. One of the most prominent ways that processing in the nervous system differs from conventional sequential logic machines (computers) is by employing massively parallel computational processors (neurons) to perform its function. Since these neurons operate in a non-linear fashion, their complex behavior is not unexpected. Moreover, and certainly more significantly, the nervous system develops most of its processing strategies from observation of the environment, i.e. it learns, while a conventional computer requires direct and precise programming. Models of learning are realized in the form of a 'learning rule'.

Many practical signal processing tasks to date have been accomplished by digital implementation of some linear transformation process such as the (Fast) Fourier Transform. Although this type of processing has proven invaluable in low-level applications, it is inadequate for higher level tasks such as signal perception (or alternatively *pattern classification*). If access to such knowledge is ever to be granted to the machine world, stringent focus must be placed on (biologically motivated) adaptive learning methods. This thesis focuses on one such method.

Organized as follows, the thesis consists of four main chapters. In chapter two a brief description of artificial neural networks and in particular feed forward artificial neural networks is given. The inclusion of this chapter is mainly to accentuate pertinent areas of the subject as well as in the interest of making this thesis a self contained document. Chapter three introduces the pattern classification problem in a formal manner. It serves mainly to make precise the ideas and concepts associated with the pattern classification problem. In addition a general model for the solution to the pattern classification problem is described. Holding the main contribution of this exposition, chapter four details the design of a learning algorithm which possesses numerous desirable properties. In chapter five the derived algorithm/system is applied to several classification problems, including the perception of pitch, as well as the classification of phonemes.

# CHAPTER TWO

## Neural Networks

Sparked by the technological advances in massive parallel processing via VLSI [1], a resurgence of interest in the capabilities of 'neural networks' has surfaced in the past decade. Contrasted to sequential computational processors, the merits of neural computing are abundant. Because their processing strategy is a distributed one, neural networks provide a high degree of fault tolerance (or so called graceful degradation). Thus a failure in one component of the overall system need not be catastrophic. Because of the inherent massive parallelism of neural networks, their direct implementation in silicon follows immediately. Such an implementation offers a significant advantage in terms of processing speed. It is for this reason that massive parallelism is believed to be essential for real time processing of sensory data, e.g. speech or vision information. Adaptation or learning (through observation of the environment) is another significant feature of a neural system.

Neural systems offer more than just pure computational robustness and adaptation. It seems appropriate that a successful model of perception should take into account the biology of the nervous system. Solutions to many problems of perception such as the pitch perception problem, pattern association, associative memory, as well as pattern recognition should draw heavily from the discipline of neural networks.

This chapter provides the basic footing that is necessary to develop an adaptive pattern classification scheme which employs a neural network as a major component.

## 2.1 Artificial Neural Networks

A general description of an artificial neural network is presented in this section. An in-depth presentation of the discipline of neural networks is out of the scope of this thesis and the description is given in only as much detail as is pertinent. The description is begun with some definitions. An excellent introduction into the realm of neural computation may be found in [2], as well as [3].

The essential components of an artificial neural network are i) the neurons themselves, ii) the interconnections between the neurons (topology), and iii) the designation of input, output and so called hidden units. These components together with their mutual operation are described in detail below.

### 2.1.1 Neuronal Model

The basic computational processor in a neural network is the neuron. In the simple model which is employed here each neuron in the network operates on its input through a sigmoidal nonlinearity. In biological terms the value of the output represents the 'firing rate' of the neuron. Associated with each neuron is a bias which effectively shifts the sigmoidal along its domain axis. Hence, a particular neuron is said to 'fire' (attaining a value close to one) if its input is a sufficiently large positive value, and similarly it is said to be 'resting' (attaining a value close to zero) if its input is a sufficiently large negative value. Before a precise definition of the sigmoidal can be stated, the concept of an n-modal function must be introduced.

**Definition 2.1** *Let $f : I \mapsto \mathbb{R}$ be a continuous, monotone, and bounded function, defined on the interval $I = (a,b)$, $a,b \in \mathbb{R}$, and let $n$ be a positive integer. One says the function, $f(\cdot)$, is* **n-modal** *if there exists a partition, $\mathcal{P} \triangleq \{\ p_0,\ p_1,\ \ldots,\ p_{n+1}\ \}$ of the interval $I$ determined by*

$$a = p_0 < p_1 < \cdots < p_n < p_{n+1} = b \tag{2.1.1}$$

*so that*

*(i) on the intervals $I_1 = (p_0, p_1)$, $I_2 = (p_1, p_2)$, $\cdots$ $I_{n+1} = (p_n, p_{n+1})$ the function $f(\cdot)$ is either convex or concave, and*

*(ii) on $I_i \cup I_{i+1}$ the function $f(\cdot)$ is neither convex nor concave.*

Note that the definition restricts n-modal functions to be bounded and monotone. The special name given to the class of 1-modal or unimodal functions is 'sigmoid'. A formal definition and some specific examples of sigmoids are detailed below.

**Definition 2.2** *A function $f : I \mapsto \mathbb{R}$ is called* **sigmoidal** *if it is unimodal.*

**Example 2.1** *Let $h : (-\alpha, \alpha) \mapsto \mathbb{R}$ for some $\alpha \in \mathbb{R}$ be given as*

$$h(x) = sgn(x) \cdot \sqrt{|x|}, \quad x \in (-\alpha, \alpha), \tag{2.1.2}$$

*then the function $h(\cdot)$ is sigmoidal.*

**Example 2.2** *Let $h : \mathbb{R} \mapsto \mathbb{R}$ be given as*

$$h(x) = \frac{1}{1 + e^{-x}}, \tag{2.1.3}$$

*then the function $h(\cdot)$ is sigmoidal and has a bounded range in $(0, 1)$.*

The particular sigmoid employed in the neural model is termed the activation function and denoted as $h(\cdot)$. Throughout the remainder of this thesis is assumed that the activation function is given by the logistic function of example 2.2. The logistic function is plotted in figure 2.1.

It is the sigmoidal which is responsible for the nonlinear behavior (and the interesting properties) of the neural system. Some properties of the logistic function denoted $h(\cdot)$ are:

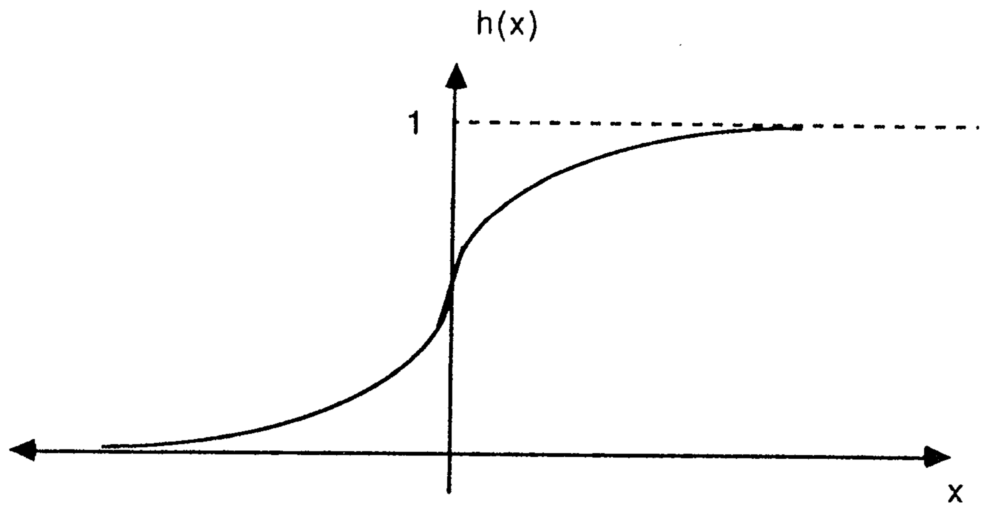


Figure 2.1: *The logistic function*

- bounded range.
- $\lim_{|x|\to\infty} \frac{\partial h(x)}{\partial x} = 0$.

As a convention the bounded range of the activation function, $h(\cdot)$ is taken to be $(0,1)$ and strictly monotonic increasing on $\mathbb{R} \Longrightarrow$

$$\frac{\partial h(x)}{\partial x} > 0, \ \forall x \in \mathbb{R}. \tag{2.1.4}$$

### 2.1.2 Interconnections

Associated with each connection is a strength or *connection weight* which quantifies the degree of coupling between two connected neurons. This strength takes on a value in the real numbers; a strength of zero indicating a null connection (unconnected), a large positive value indicating a strong excitatory connection, and a large negative value indicating a strong inhibitory connection.

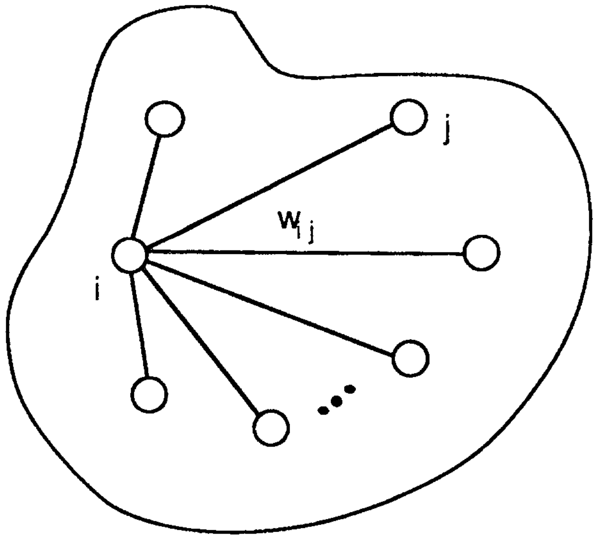


Figure 2.2: *An arbitrary artificial neural network (ANN)*

An arbitrary artificial neural network (ANN) with $n$ nodes may be represented by a weight matrix, $W \in \mathbb{R}^{n \times n}$ and a bias vector $b \in \mathbb{R}^n$. A schematic of an ANN is depicted in figure 2.2.

Given an $n$ node neural network, the set of nodes present in the network can be enumerated as

$$\mathcal{N} = \{\ 0,\ 1, \ldots,\ n-1\ \}. \tag{2.1.5}$$

Now the connection between two arbitrary nodes, $i$ connected to $j$, can be expressed as the scalar $w_{ji} \in \mathbb{R}, i, j \in \mathcal{N}$. Forming the weight matrix $W$ as $W = [w_{ij}]$ and the vector of biases as $b = [b_i]$. Any ANN then may be fully described by the parameters $(W, b)$, where $w_{ji} \in \mathbb{R}$ is the strength of the connection between nodes $i$ and $j$, and $b_i \in \mathbb{R}$ is the bias associated with neuron $i$.

### 2.1.3 Operation

Each neuron operates asynchronously and in parallel with all other neurons in the network; such systems have been referred to under the auspices of 'distributed parallel processing'. The following presents a mathematical description of the operation of an artificial neural system.

At the input of each neuron is the weighted sum of outputs from all other neurons. Denoting the input to neuron $i \in \mathcal{N}$ as $x_i$ and the output as $y_i$ then

$$x_i = \sum_{j \in \mathcal{N}} w_{ij} \cdot y_j \ + \ b_i, \quad \forall i \in \mathcal{N}. \tag{2.1.6}$$

Letting $x = [x_i] \in \mathbb{R}^n$ represent the vector of inputs, $y = [y_i] \in \mathbb{R}^n$ represent the vector of outputs, and $b = [b_i] \in \mathbb{R}^n$ represent the vector of biases, the neuronal input for every node can be concisely expressed in matrix notation as an affine operation

$$x = Wy + b. \tag{2.1.7}$$

The output of each neuron is simply its input mapped through the sigmoidal nonlinearity, $h(\cdot)$,

$$y_i = h(x_i), \quad \forall i \in \mathcal{N}. \tag{2.1.8}$$

If a nonlinear operator, $H : \mathbb{R}^n \mapsto [0,1]^n$, is defined as simply the vector determined by operating element-wise on an input vector then the output may be concisely written as

$$y = H(x). \tag{2.1.9}$$

As a consequence of this structure it is easy to establish the validity of the following fact.

**Fact 2.1** *The partial of the ith output function, $y_i$, with respect to any parameter in $(W, b)$ is positive, i.e.*

$$\frac{\partial y_i}{\partial w_{ij}} = \left.\frac{\partial h(x)}{\partial x}\right|_{x=x_i} \frac{\partial x_i}{\partial w_{ij}} > 0 \tag{2.1.10}$$

*and*

$$\frac{\partial y_i}{\partial b_i} = \left.\frac{\partial h(x)}{\partial x}\right|_{x=x_i} \frac{\partial x_i}{\partial b_i} > 0 \tag{2.1.11}$$

*for any $i, j \in \mathcal{N}$.*

It is clear that the fact is a consequence of the mononicity and boundedness properties of the sigmoid. This fact will prove useful in the latter portion of this thesis (refer to chapter four).

## 2.2 Feed Forward Networks

Here is described a very useful class of neural networks known as feed forward (or layered) networks. At least theoretically, restricting attention exclusively to feed forward networks represents no loss of generality. This is due to the fact that for any arbitrary (recurrent) network there exists an equivalent feed forward network [3].

A layered neural network consists of a serial linkage of layers, where each layer is connected only to the layer following it (hence signals flow in only one

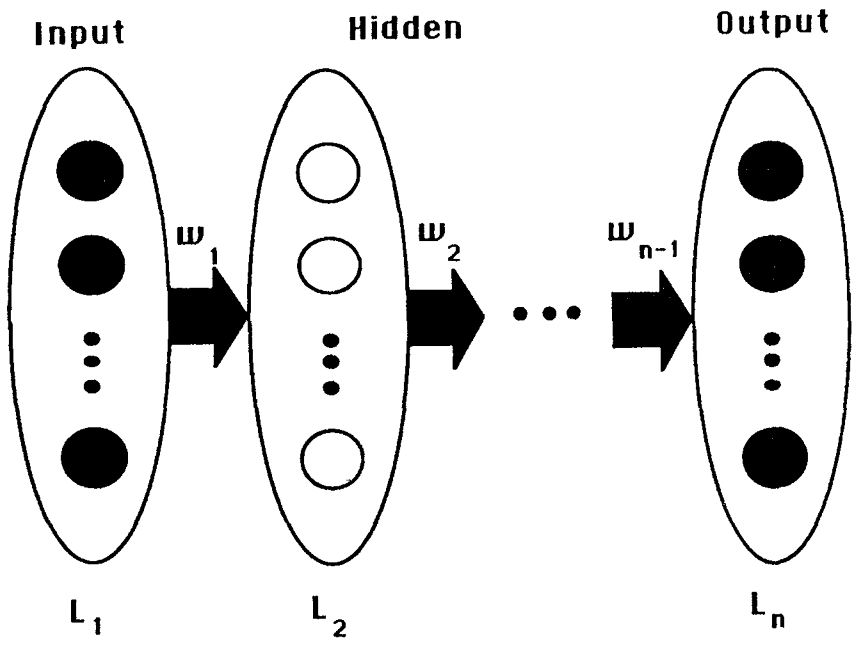


Figure 2.3: *The feed forward neural topology*

direction). The first layer is designated the input layer and the last is designated the output layer. A depiction of such a feed forward topology is given in figure 2.3.

It is clear that the constraint of a feed forward topology leads to a corresponding constraint on the weight matrix, $W$, of the network. Not surprisingly, the fixed feed forward structure forces most of the entries of $W$ to be null.

Developed now is some further notation regarding the special form of a feed forward network. A feed forward network topology can be parameterized by the number of neurons (or nodes) in each layer, and the number of layers. With the number of layers given as $n_l$, and the number of nodes in each layer as

$l_i,\ i = 1, 2, \ldots, n_l$ then the set of nodes present in each layer can be written as

$$\begin{aligned}
\mathcal{I} = L_1 &= \{\ 1,\ 2,\ \ldots,\ l_1\ \} \\
L_2 &= \{\ l_1 + 1,\ \ldots,\ l_1 + l_2\ \} \\
&\vdots \\
L_j &= \left\{ \sum_{i=1}^{j-1} l_i\ + 1,\ \ldots,\ \sum_{i=1}^{j} l_i \right\} \\
&\vdots \\
\mathcal{O} = L_{n_l} &= \left\{ \sum_{i=1}^{n_l - 1} l_i\ + 1,\ \ldots,\ \sum_{i=1}^{n_l} l_i \right\}.
\end{aligned}$$

The sets $L_1$ and $L_{n_l}$ are given the special names $\mathcal{I}$ and $\mathcal{O}$ to denote input and output sets respectively. The quantities $l_1$ and $l_{n_l}$ are also given the special names $n_i$ and $n_o$ denoting the number of input and output nodes respectively. Residing between the input and output layers are the so called 'hidden' layers $L_2$ through $L_{n_l - 1}$. This notation will be adhered to whenever the discussion involves a feed forward (FF) network.

## 2.3 Learning

One primitive form of learning that is especially amenable to neural architectures is that of pattern association. In this case it is desired to map a set of input patterns to a specific set of known output patterns. For example associating a face with a voice, or a jingle with a merchandiser's product.

Once knowledge of the weight matrix, $W$, and the bias vector, $b$, are given the dynamics of the ANN are completely determined. Conversely, if one of the parameters embedded in the bias vector or weight matrix is perturbed the dynamics of the ANN system will also be perturbed. From a mathematical viewpoint once the structure of the ANN has been fixed, it functions as nothing more than a continuous nonlinear mapping which is a function of the parameters making up the bias vector and weight matrix. As such, the problem of learning is then reduced to determining the parameters (weights and biases) which will force the ANN system to perform the desired mapping.

### 2.3.1 Numerical Optimization

Much of the types of learning that have been proposed for ANN's can be cast into the class of optimization problems. One popular method of learning through minimization of some objective function is the backpropagation algorithm. This method is described in detail below. Neural networks have been employed to solve a host of other computational problems through optimization techniques [4]; for example associative memory [5], the traveling salesman problem [6], analog decoding [7], etc.

### 2.3.2 The Backpropagation Method

For many years the benefits of multi-layered feed forward neural networks have been denied because of the lack of algorithms to train them. Recently. though, an optimization approach has been proposed which has met with great success. This so called backpropagation algorithm [3] provides a method for a feed forward neural architecture (layered network) to learn a set of network parameters (weights and biases) which map a known input pattern set to a corresponding known desired output set.

As before let $n_l$ denote the number of layers in the feed forward network. Further let $n_p$ be the number of patterns in the training set. Also let the input training set be given as $\mathcal{T}$ and the output due to input pattern $p$ as $y_p = (\ y_{p1}\ y_{p2} \ldots\ y_{pn_o}\ )^T$, and the target corresponding to input pattern $p$ as $t_p = (\ t_{p1}\ t_{p2} \ldots\ t_{pn_o}\ )^T$, $p \in \mathcal{P}$. Here $n_o \triangleq l_{n_l}$ is the number of output nodes and $\mathcal{P} \triangleq \{1, 2, \ldots, n_p\}$ the input training index set, i.e. $\mathcal{T} = \cup_{p \in \mathcal{P}}\ t_p$.

Because of the constrained form of a FF ANN the equations of operation (2.1.7, 2.1.8) can be written in the form

$$x_{pi} = \sum_{j \in L_{k-1}} w_{ij} y_{pj} + b_i \tag{2.3.1}$$

$$y_{pi} = h(x_{pi}), \quad i \in L_k,\ k = 2, 3, \ldots, n_l \tag{2.3.2}$$

It is desired to minimize the total squared error of the actual output, $y_p$ compared to the desired $t_p$. The objective energy function may then be written as

$$E = \sum_{p \in \mathcal{P}} E_p \tag{2.3.3}$$

$$E_p = \frac{1}{2} \sum_{k \in \mathcal{O}} (y_{pk} - t_{pk})^2 \tag{2.3.4}$$

$$= \frac{1}{2} \|y_p - t_p\|_2^2 \tag{2.3.5}$$

Following a descent type algorithm the parameter updates become

$$\Delta w_{ij} = -\gamma^2 \frac{\partial E_p}{\partial w_{ij}} \tag{2.3.6}$$

$$\Delta b_i = -\gamma^2 \frac{\partial E_p}{\partial b_i}, \tag{2.3.7}$$

where $\gamma^2$ is called the learning rate and represents the step size in the descent search. All that is necessary now is to compute the partials given in the update equations 2.3.7. Attention is restricted to the computation of the partial with respect to the weight, $\frac{\partial E_p}{\partial w_{ij}}$, since the other is analogous. Clearly this partial may be written by the chain rule as

$$\frac{\partial E_p}{\partial w_{ij}} = \frac{\partial E_p}{\partial y_{pi}} \frac{\partial y_{pi}}{\partial x_{pi}} \frac{\partial x_{pi}}{\partial w_{ij}}. \tag{2.3.8}$$

Computation of the latter two terms in equation 2.3.8 is direct and presents no problems. Computation of the first term, $\frac{\partial E_p}{\partial y_{pi}}$, however, is not direct for nodes outside the output layer, $\mathcal{O}$, and requires special attention. The essence of the backpropagation algorithm is to determine a recursive method for computing the quantity $\delta_{pi} \triangleq \frac{\partial E_p}{\partial y_{pi}}$.

For nodes in the output layer $\delta_{pi}$ is simply computed from the expression for the energy 2.3.5 as

$$\delta_{pi} = y_{pi} - t_{pi}, \quad i \in \mathcal{O}. \tag{2.3.9}$$

Now consider a node (say node $i$) in the $l$th layer, i.e. take $i \in L_l$, $l = 1, 2, \ldots, n_l - 1$. Again by the chain rule

$$\begin{aligned} \delta_{pi} &= \sum_{k \in L_{l+1}} \frac{\partial E_p}{\partial y_{pk}} \frac{\partial y_{pk}}{\partial x_{pk}} \frac{\partial x_{pk}}{\partial y_{pi}} && (2.3.10) \\ &= \sum_{k \in L_{l+1}} \delta_{pk} h'(x_{pk}) w_{ki}, \quad i \in L_l, \ l = 1, 2, \ldots, n_l - 1. && (2.3.11) \end{aligned}$$

Together equations 2.3.9 and 2.3.11 give a recursive scheme (starting from the output layer and *propagating backwards* to the input layer) for computing the quantity $\delta_{pi}$ for any node in the network.

Although the algorithm has met with great success, some of the drawbacks of the backpropagation method are listed below:

- requires distinct training and operating phases
- the learning requires apriori knowledge of the entire set of desired outputs (i.e., it is a supervised learning rule)
- convergence to a global minimum of the energy function is not guaranteed
- does not allow for the incorporation of new knowledge once the training has been completed

CHAPTER
# THREE
# Pattern Classification

In this chapter a formal statement of the pattern classification problem is presented along with a generalized approach to the solution of the pattern classification problem. A general system model, which includes a neural network as a key element, is developed which incorporates a wide class of so called pattern classifiers. The model also includes the backpropagation algorithm as a special case.

## 3.1 The Pattern Classification Problem

The essence of pattern classification lies in the partitioning of a set of patterns with respect to some measure of the likeness of each pattern. This notion of the likeness of two patterns as well as a precise description of a partition is formalized in the following definitions.

Let $\mathcal{X}$ denote the set of patterns, or pattern space, that the classifier is expected to segment. This space is assumed to be an inner product space (Hilbert space, e.g $L_2$ or $\mathbb{R}^n$) with the norm induced by the inner product

$$\|x\| = (\langle x, x\rangle)^{\frac{1}{2}}. \tag{3.1.1}$$

**Definition 3.1** *The finite collection of subsets, $S \triangleq \{s_k : s_k \subset \mathcal{X}\}_{k=1}^{n}$, forms a partition of the Hilbert space $\mathcal{X}$ if*

1. $\mathcal{X} = \bigcup_{i=1}^{n} s_i$ ,     *and*

2. $s_i \cap s_j = \emptyset, \quad \forall i \neq j = 1, 2 \ldots, n.$

Take note that the regions $s_k,\ k = 1, \ldots, n$ need not be connected.

**Definition 3.2** *A function $r : \mathcal{X} \times \mathcal{X} \mapsto [0, 1]$ is called a similarity measure defined on the pattern space $\mathcal{X}$ if the two following conditions are satisfied:*

1. $r(\cdot, \cdot)$ *is continuous*

2. $r(x, x) = 1, \quad \forall x \in \mathcal{X}.$

Condition 1 insures that small perturbations in a particular pattern when compared to the same unperturbed pattern will yield small changes in similarity. The second condition in the definition is somewhat arbitrary as is the restriction that the range of the similarity measure be $[0, 1]$. What is important, though, is that the range of $r(\cdot, \cdot)$ be compact with $r(x, x)$ attaining a value on the boundary of that range for all patterns, $x$, in the pattern space. Some examples of similarity functions are presented below.

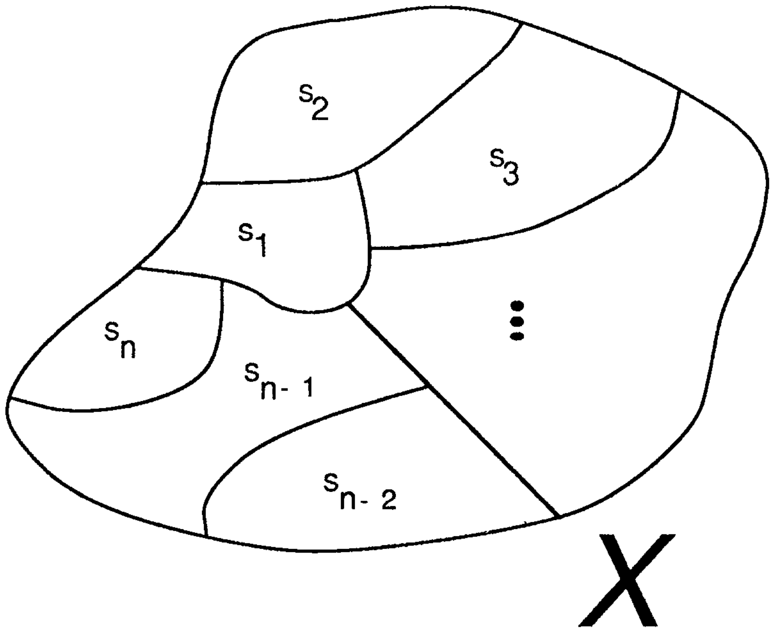


Figure 3.1: *Pattern classification viewed as a partition*

**Example 3.1** *With $x, y \in \mathcal{X}$ the following functions are similarity measures on $\mathcal{X}$:*

1. $r(x,y) = e^{-\frac{\|x-y\|_2^2}{\sigma^2}}, \quad \sigma \in \mathbb{R}$
2. $r(x,y) = \frac{|\langle x,y \rangle|}{\|x\|_2 \|y\|_2}$
3. $r(x,y) = 1.$

Case three, although a perfectly valid similarity measure, is somewhat useless in the practical sense since it says that any partition segments as well as any other. With the concept of a similarity measure established, a formal statement of the pattern classification problem can now be presented. It will be evident that the problem (and hence the solution) inherently has three major parameters: the first being the similarity measure; the second, a measure of the quality of the desired solution; and finally, a training subset of the pattern space $\mathcal{X}$.

**Problem 3.1** *The Pattern Classification Problem. Given an $\epsilon > 0$, a finite subset of an input space $\mathcal{X}$, denoted $\mathcal{T}$, and a similarity measure $r(\cdot,\cdot)$ defined on $\mathcal{X}$, determine a finite partition (or segmentation) $S$, of the space $\mathcal{X}$ so that*

1. *$S$ forms a partition of $\mathcal{X}$, and*
2. *$x, y \in s_k \cap \mathcal{T} \implies |1 - r(x,y)| < \epsilon$*

Since it is stated with respect to the three parameters, $r, \epsilon$, and $\mathcal{T}$, the problem is actually an infinite family of problems indexed by these three parameters. For notational completeness, then, the pattern classification problem should be specified with this dependence explicitly as PCP$(r, \epsilon, \mathcal{T})$. When the value of one or more of these parameters is irrelevant or can be determined by the context of the situation it will be dropped from the notation.

A simple example illustrating the need for a notion of minimality of the solution to the PCP is given below.

**Example 3.2** *Suppose the input space $\mathcal{X} = \mathbb{R}^2$, and the training set*

$$\mathcal{T} = \left\{ x_i : x_i = (\theta_1^i, \theta_2^i) \right\}_{i=1}^{6}, \; x_i \in \mathbb{R}^2, \; \theta_1^i, \theta_2^i \in \mathbb{R} \quad i = 1, \ldots 6 \tag{3.1.2}$$

*as shown in figure 3.2. Suppose further that the similarity measure is given as*

$$r(x_i, x_j) = e^{-|\theta_2^i - \theta_2^j|}. \tag{3.1.3}$$

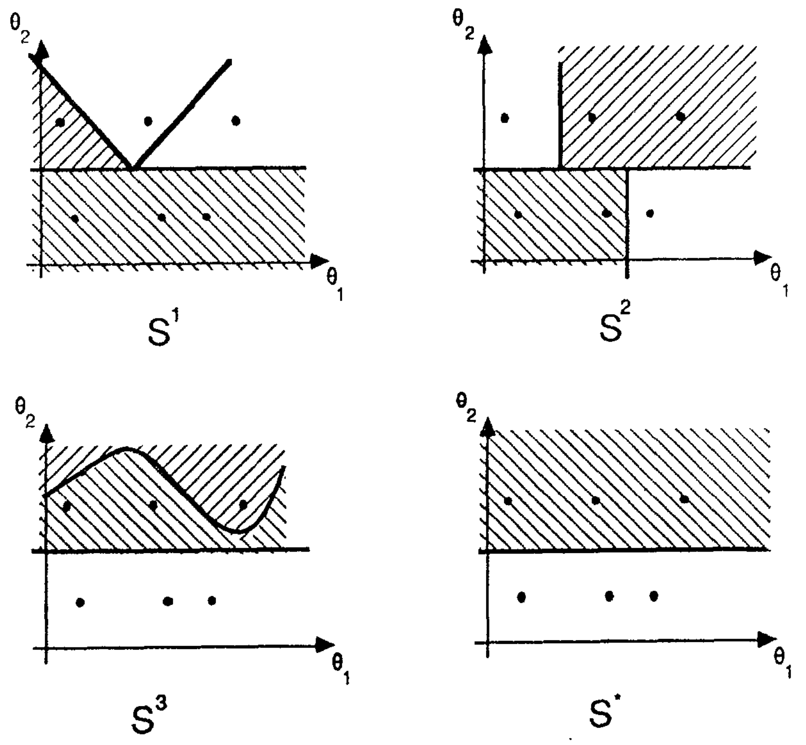


Figure 3.2: *A two dimensional PCP and some of its solutions*

Note that any two samples lying on the same horizontal line in the parameter space are considered identical with respect to this similarity measure. Because of this any of the partitions shown in figure 3.2 solves the PCP($\epsilon$), $\forall\epsilon > 0$. Clearly, though it is the partition $S^*$ which is the best solution. This example underscores the need for the notion of a minimal partition.

A minimal partition is best described as the "smallest" partition (i.e. a partition with the smallest number of distinct classes) that solves PCP($r, \epsilon, \mathcal{T}$). A formal definition is now given.

**Definition 3.3** *Let $F$ be the family of partitions of $\mathcal{X}$ such that condition two in the PCP holds, i.e.,*

$$F = \Big\{ S = \{s_k\}_{k=1}^{N} : x, y \in s_k \cap \mathcal{T} \Longrightarrow |1 - r(x,y)| < \epsilon \Big\}. \tag{3.1.4}$$

*The partition $S^*$ is said to be* **minimal** *if*

1. *$S^* \in F$, and*

2. *$\mathcal{C}(S^*) \leq \mathcal{C}(S), \quad \forall S \in F$.*

Here the mapping $\mathcal{C}$ denotes a type of cardinality and is given as $\mathcal{C}(S) = N$ where $S$ is given as in definition 3.3.

## 3.2 The Classifier Model

The goal of this section is to present a (neural) system which solves the PCP. Depicted in figure 3.3 is the generalized model of the classification process. The three major components of the model are the transformation, the pattern association, and the update.

**Transformation** The role of the transformation in the operation of the system is multifold. First, it is necessary to transform any input data into the domain space of the pattern associator. For instance, in the case of the pattern associator taken as a neural network, this means that the transformation should take inputs from the pattern space and transform them into a finite set of

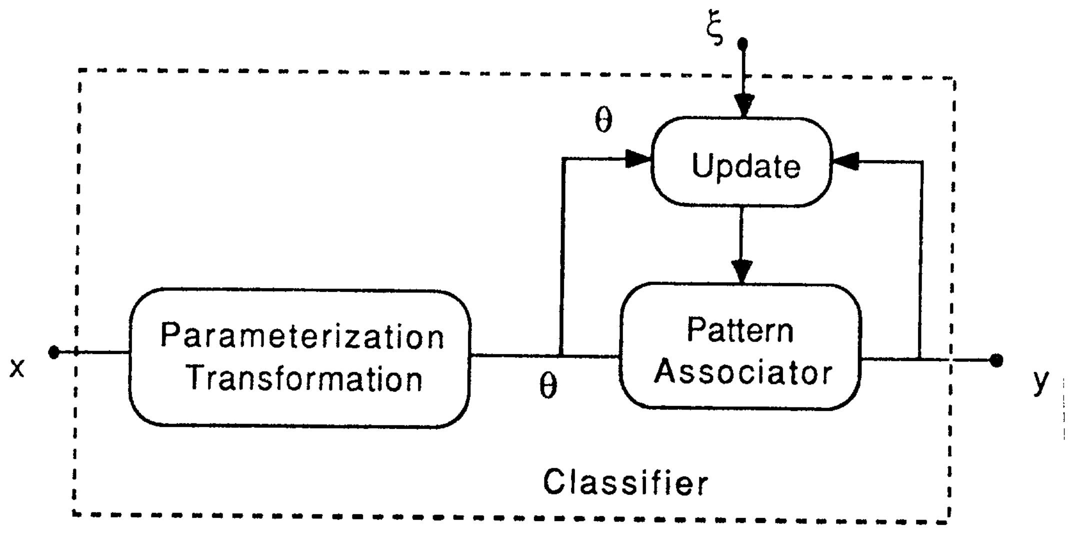


Figure 3.3: *Classification system model*

numbers between zero and one which serve as the input to the neural network. Second, it is necessary to perform some sort of intelligent data reduction on the input patterns. In many cases the input space consists of continuous data, e.g. acoustic signals. Depending on the desired function of the classifier, a simple sampling transformation may or may not be a desirable transformation. Ideally, one would like to determine a transformation process which emphasizes only the pertinent information with respect to classification.

**Pattern Association** Operation of the pattern associator is assumed to be completely governed by a parameter vector, denoted $z$. In general the pattern associator block of the classifier system need not be a neural network, however, a neural network readily lends itself to this class of systems. A feed forward neural network is chosen as the pattern associator with parameter vector $z = (W, b)$, where $W \in \mathbb{R}^{n_o \times n_i}$ represents the weight matrix and $b \in \mathbb{R}^{n_o}$ the bias vector. Justification for such a choice is presented in section 3.3.

**Learning: the Update** Learning is facilitated through changes in the pattern associator parameters $z$. The generalized learning rule can be written as

$$z^{(n+1)} = z^{(n)} + \Delta z^{(n)}. \tag{3.2.1}$$

Types of learning are usually distinguished by inclusion or absence in the class of learning rules that are considered unsupervised. What the term 'unsupervised' means is somewhat subjective and the distinction can sometimes be subtle. Here, the notion of a supervised (vs. unsupervised) learning rule is made precise in the context of the pattern associator update by the inclusion (or exclusion) of the external knowledge vector $\xi$. That is, there is some function $f$ so that

- Unsupervised:

$$\Delta z^{(n)} = f(\Theta, y) \tag{3.2.2}$$

- Supervised:

$$\Delta z^{(n)} = f(\Theta, \xi, y). \tag{3.2.3}$$

In this thesis only a certain class of updates is considered. This class of updates is referred to as minimum energy updates and is described below. As discussed earlier, one may formulate learning as a minimization of some objective function, $E(\cdot)$, representing energy. Such a formulation is motivated by physical

| Variable | $\subset$ | Description |
|---|---|---|
| $n_p$ | $\mathbb{R}$ | number of training patterns |
| $n_i$ | $\mathbb{R}$ | number of pattern associator inputs |
| $n_o$ | $\mathbb{R}$ | number of pattern associator outputs |
| $\gamma^2$ | $\mathbb{R}^+$ | learning rate |
| $\mathcal{P}$ | $\mathbb{N}$ | pattern indices |
| $\mathcal{O}$ | $\mathbb{N}$ | output indices |
| $x_k,\ k \in \mathcal{P}$ | $\mathbb{R}^{n_i}$ | $k$th input pattern |
| $y_k,\ k \in \mathcal{P}$ | $\mathbb{R}^{n_i}$ | $k$th output pattern |
| $\mathcal{T}$ | $\mathbb{R}^{n_p}$ | the training set |
| $\Theta$ | $\mathbb{R}^{n_i}$ | the parameter transform |
| $\xi$ | $\mathbb{R}^{n_o}$ | external knowledge vector of current input, $x(t)$ |
| $z = (W, b)$ | $\mathbb{R}^{n_o \times n_i} \times \mathbb{R}^{n_o}$ | classifier parameters |

Table 3.1: *Classifier notation*

principles of least energy and the fact that nature enforces energy minimization. The generalized learning rule as developed from descent methods on $E$ is simply given as given as

$$z^{(n+1)} = z^{(n)} - \gamma^2 \nabla E, \tag{3.2.4}$$

where $\gamma^2 \in \mathbb{R}^+$ is the step size, usually termed the 'learning rate'. For the case of minimum energy updates then clearly $\Delta z^{(n)} = -\gamma^2 \nabla E$. This indicates that a minimum energy update will be a supervised rule only if the energy function depends on the external knowledge vector.

Much of the classifier notation that will be used throughout this thesis is summarized in table 3.1.

Some of the sets used in table 3.1 are defined below:

$$\mathcal{P} \;=\; \{1, 2, 3, \ldots, n_p\} \tag{3.2.5}$$

$$\mathcal{O} = \{1, 2, 3, \ldots, n_o\} \tag{3.2.6}$$

$$\mathcal{T} = \bigcup_{k=1}^{n_p} \{x_k\} \tag{3.2.7}$$

$$\Theta = \{\theta_k\}_{k=1}^{n_i}. \tag{3.2.8}$$

## 3.3 Existence of Solutions and the Topology

It is evident that the success of the pattern classification is dependent on three major components of the system: transformation, pattern association, and learning. To simplify the discussion it is assumed that no pertinent information is lost in the transformation; e.g., the transformation process is taken to be the identity. The question which must be addressed now is the following:

*Does there exist a pattern associator topology and associated weights and biases which will solve the PCP?.*

If the answer to this question is 'no' then pursuing such an avenue for its solution is certainly pointless. If the answer is 'yes' then it is the task of the learning to find a parameter vector that solves the problem. Fortunately, the answer to this question is in fact 'yes'.

This question has already been addressed, although in a more general setting, by Cybenko [8], [9]. In short, Cybenko has shown that a very wide class of functions (in particular continuous functions of finite support) may be approximated by a feed forward neural network with just a single hidden layer

[8]. Following Cybenko[8], one may relate this work to pattern association by introducing the concept of a *decision* function.

**Definition 3.4** *Let $S = \{s_k\}_{k=1}^N$ be a partition of the space $\mathcal{X}$. A function $d : \mathbb{R}^n \mapsto \mathbb{N}$ is called a decision function if*

$$d(x) = k \iff x \in s_k. \tag{3.3.1}$$

It is clear that a decision function is discontinuous for non trivial partitions on $\mathcal{X}$.

With $m$ representing the Lebesgue measure, the following theorem is a fundamental result of Cybenko [8]. The theorem extends the basic class of approximatable functions to include discontinuous functions of the 'decision' type at the cost of misclassification on a set of arbitrarily small measure.

**Theorem 3.1** *Let $h(\cdot)$ be a continuous sigmoidal function. Let $d(\cdot)$ be the decision function for any finite measurable partition of $I_n \triangleq (0,1)^n$. For any $\epsilon > 0$, there is a finite sum of the form, $G : \mathbb{R}^n \mapsto \mathbb{R}$,*

$$G(x) = \sum_{i=1}^{N} \alpha_i h(w_i^T x + b_i) \tag{3.3.2}$$

*and a set $D \subset I_n$, so that the Lebesgue measure of $D$, $m(D) \geq 1 - \epsilon$ and*

$$|G(x) - d(x)| < \epsilon \quad \forall x \in D, \tag{3.3.3}$$

*where $\alpha_i, b_i \in \mathbb{R}$, and $w_i \in \mathbb{R}^n$.*

| Topology ($n_l$) | Form of Decision Region |
|---|---|
| $n_l = 2$ | half plane bounded by hyperplane |
| $n_l = 3$ | convex |
| $n_l = 4$ | arbitrary |

Table 3.2: *Mapping abilities of feed forward networks*

**Proof** See [8].

Basically the theorem says that an arbitrary decision function may be approximated by a single hidden layer ANN with misclassification occurring in an arbitrarily small neighborhood of the underlying classes. Hence, this topology can solve the PCP simply by chosing the decision function corresponding to a partition which solves the PCP. However, the proof given by Cybenko is non-constructive and says nothing about the number of neurons that are needed in the hidden layer to achieve any given mapping. Typically the determination of this quantity is carried out experimentally for a particular application.

An excellent review of the pattern association abilities with respect to topology (number of layers) of feed forward neural networks is presented in [2]. Essentially, the topology limits the type of regions that are able to be classified. Mapping abilities of two, three, and four layer networks are given in compact form in table 3.2 where $n_l$ is the number of layers.

With the similarity function as in function number two in example 3.1, the cosine of the angle between its two arguments, a two layer network is sufficient.

This is because segmentations based only on angle information may be formed by half planes. For the gaussian function (number one in the same example), the similarity measure is a function of the euclidean distance metric. For this reason decision regions should not in general be half planes. It is more plausible that acceptable decision regions will be convex indicating a three layer topology.

## 3.4 Transformation: The Cochlea Model

The requirement that the transformation process map the pattern space to the domain of the pattern associator is not overly restrictive and generally presents no obstacle to the solution. However, the data reduction requirement is problem specific and has no methodical implementation. The choice of transformation is left as a design decision. Transformations in biological systems are found quite naturally in the tactile, auditory, visual, and olfactory senses. Each transformation process focuses on a specific sensory task underscoring the importance of a pertinent transform in the classification model.

Should the system be required to classify acoustically perceived phenomenon incorporation of a model of cochlea as the transformation seems appropriate (e.g. pitch perception or phoneme recognition). Moreover, much of the perception of the auditory environment is a product of its biological implementation. In this section a model of the cochlea [10] is briefly described which is applied (chapter 5) to the classification of acoustic signals in the cases of pitch and phoneme recognition.

For a detailed description of the cochlea model the reader is referred to [11], [12], [10]. Essentially, the acoustic model consists of a bank of 128 filters. Each of these filters has an associated *characteristic frequency* at which it responds best. A crude approximation to the operation of the cochlea would be to envision each of these filters to be a band pass with center frequency at the characteristic frequency. Typical cochlea outputs may be seen in appendix B for the case of spoken words and sustained musical tones.

# CHAPTER FOUR

## An Orthogonalization Learning Algorithm with Generalization

This chapter details the design of a classifier system of the type outlined in the previous chapter. This system is based on the choice of a similarity function which is restricted to operate only on the system outputs. The update is taken as a minimum energy update, and hence formulated as an optimization problem in the form of *locally* minimizing some energy function with respect to an initial point. That is, it is desirable to find the local minima which is closest to the starting point in a sense to be made precise.

Formulated in this way, the algorithm is endowed with many desirable properties. Among the most important of these are that (i) the learning scheme is unsupervised in the sense that no apriori (or external) knowledge is incorporated; and (ii) learning and operation occur *simultaneously.* Property (ii) is especially significant since learning in real biological neural networks is accomplished in this fashion.

Along with an appropriate initialization scheme, a descent algorithm is developed that is computationally efficient. Through the processes of estimation and annihilation, the computation requirements are reduced drastically while preserving the convergence properties of the underlying descent.

## 4.1 Solution Methodology

A proposed system for the solution of the classification problem is detailed in chapter three. Here, it is seen that the classifier system (figure 3.3) is composed of three main components: transformation, pattern association, and update. It has been previously seen that the transformation portion of the design is problem specific and therefore no further mention of it is made unless with respect to a specific application. The pattern association is simply a parameterized (by $z$) mapping $H_z(\cdot)$ which may be viewed as an artificial neural network. Hence, the only portion of the system not completely specified is the update. The remainder of this chapter is dedicated to investigating one method of determining the update.

As related earlier, the update is chosen to be a minimum energy update (gradient descent with a constant step size). Always the problem is one of searching for a 'closest' local minimum of the objective energy (a function of $z$, the pattern associator parameter vector) with respect to some starting point, denoted $z_0$.

The proposed route to a solution is outlined as follows:

- choosing a similarity which is dependent only on the pattern associator outputs (the cosine function)
- constructing an energy landscape $E_{\perp}(\cdot)$ with local minima aimed at solving the PCP
- incorporating a 'penalty' term $P(\cdot)$ aimed at satisfying the constraints on the pattern associator outputs (to be specified)
- selecting an initial parameter vector $z_o$ aimed at preserving input similarity
- performing descent to a 'closest' local minima of the energy function

$$E(\cdot) \triangleq E_{\perp}(\cdot) + P(\cdot) \tag{4.1.1}$$

## 4.2 Local Minimization

Proposed for the updating rule is a local minimization scheme. It is the purpose of this section to clarify exactly in what sense this local minimization is performed. Intuitively it is clear that one only need identify a 'closest' local minima with respect to some starting location. A precise statement of the local minimization problem is stated with respect to some continuously Fréchet differentiable energy function $E(z)$.

**Problem 4.1** *Local Minimization Problem. Given an initial point* $z^{(0)} = z_0$ *find a local minimizer* $z^*$ *of the function* $E(z)$ *so that the path from* $z_0$ *to* $z^*$ *denoted* $\Gamma$ *satisfies*

1. $\langle \nabla_z E, d \rangle < 0 \quad \forall d$ *tangent to* $\Gamma$

2. $\|z^* - z_0\| \leq \|z - z_0\| \quad \forall z$ *which are local minima of* $E(\cdot)$ *and the path* $\Gamma$ *from* $z_o$ *to* $z$ *satisfies condition 1.*

Condition 1 is a monotonic descent condition which says that the energy must always be decreasing on the path $\Gamma$ from $z_o$ to $z^*$. Condition 2 merely formalizes the notion of the shortest such path.

## 4.3 Energy Function Design

It is the objective of the design to create an energy landscape that has local minima at points which correspond to valid solutions to the pattern classification problem. In order to satisfy the desired constraints (C) on the solutions, the designer can appeal to the penalty method described in the previous section.

### 4.3.1 Construction

Recall that the pattern classification problem is stated with respect to a similarity measure (see section 3.1). One choice of similarity measure, $r(\cdot, \cdot)$, might be the cosine of the angle between the two vector arguments. For discrete vectors $x, y \in [0,1]^n$

$$r(x,y) \triangleq \frac{\langle x, y\rangle}{\|x\|\,\|y\|} = \frac{1}{\|x\|\,\|y\|}\sum_{k=1}^{n} x_k y_k \tag{4.3.1}$$

for notational convenience $r(y_k, y_p)$ will be denoted as simply $r_{kp}$, where $k, p \in \mathcal{P}$.

The energy function is chosen as

$$\begin{aligned} E(y) &= E_{\perp}(y) + P(y) && (4.3.2)\\ E_{\perp}(y) &= \sum_{p\in\mathcal{P}} E_p && (4.3.3)\\ E_p(y) &= \sum_{k\in\mathcal{P}} g_{\alpha}(r_{kp}) && (4.3.4) \end{aligned}$$

where $g_\alpha(\cdot) : [0,1] \rightarrow \mathbb{R}^+$ and $P(y)$ is as in equation 4.6.1.

It will be seen that the role of the function $g_\alpha(\cdot)$ in the classification process is significant. When chosen appropriately, this function will endow the system with an essential clustering / orthogonalization property. 'Orthogonalization' is termed the process by which patterns are cast into different classes. Specifically, given two patterns, $y_k$ and $y_p$, orthogonalization implies $r_{kp} \rightarrow 0$. Similarly the term 'clustering' refers to the merging of patterns into the same class, i.e. $r_{kp} \rightarrow 1$. In other words patterns which are close in similarity should be mapped to the same class while patterns which are not close in similarity should be mapped to distinct classes. With these considerations in mind, the form of the function $g_\alpha(\cdot)$ is now determined.

### 4.3.2 Design of $g_\alpha(\cdot)$

To avoid pathological situations a prerequisite placed on the function $g_\alpha(\cdot)$ is that it be both continuous and differentiable. More importantly it is chosen to facilitate the desired clustering / orthogonalization property. Since $r_{kp} \in \{0, 1\}$ corresponds to orthogonal and parallel conditions on the pair $y_k$ and $y_p$ respectively, requiring the function $g_\alpha(\cdot)$ to attain a local minimum at these and only these points insures that a local minimization scheme will arrive at a solution with $r_{kp} \in \{0, 1\}$. It is not difficult to see that this property is equivalent to the requirement that $g_\alpha(\cdot)$ be concave.

Since a local minimization scheme will at some point involve the gradient of $E_\perp$ it is quite clear that the derivative of the function $g_\alpha(\cdot)$ will appear in this gradient. Therefore, for the sake of computational simplicity, it is desirable that the derivative of the nonlinear function $g_\alpha(\cdot)$ be affine, that is

$$\frac{\partial}{\partial r} g_\alpha(r) = ar + b, \qquad a, b \in \mathbb{R} \tag{4.3.5}$$

In summary, the properties of the function $g_\alpha(\cdot)$ which are sought are given below. It is required that the function $g_\alpha(\cdot)$ be

1. continuous and differentiable,

2. concave,

3. affine in the derivative $\frac{\partial g(r)}{\partial r}$.

It will be seen in the sequel that property 3 indeed helps to minimize computational complexity.

These properties lead directly to the choice of the quadratic

$$g_\alpha(x) = -\left[x - (\alpha - \frac{1}{2})\right]\left[x - (\alpha + \frac{1}{2})\right] + \text{ const} \quad \alpha \in (0,1). \tag{4.3.6}$$

The 'const' term above is chosen so that $g_\alpha(\cdot)$ is always positive on $[0,1]$, i.e. $g_\alpha(x) \geq 0, \ \forall x \in [0,1]$. Related to the resolution of the partition solving the PCP, the $\alpha$ term is used to control the resolution of the resulting classification. Depicted in figure 4.1 is the function $g_\alpha(x)$. It is a concave quadratic whose roots are uniquely determined by the parameter $\alpha$. It is easy to see that the restriction of $\alpha \in (0,1)$ insures that the points $(r, g_\alpha(r)) = (0, g_\alpha(0))$ and $(1, g_\alpha(1))$ are local minima of the constrained problem:

$$\min_{x \in [0,1]} \quad g_\alpha(x) \, . \tag{4.3.7}$$

## 4.4 Constraints on the Solutions

All that is required for an acceptable solution to the pattern classification problem is that outputs to different patterns (with respect to some similarity measure) produce orthogonal outputs. Chosen here is the more demanding (and conceptually pleasing) constraint that the outputs be members in the unit basis set. For clarity the constraints are stated informally as

$$\text{(C)} \quad \{\text{outputs belong to the unit basis set}\} \, . \tag{4.4.1}$$

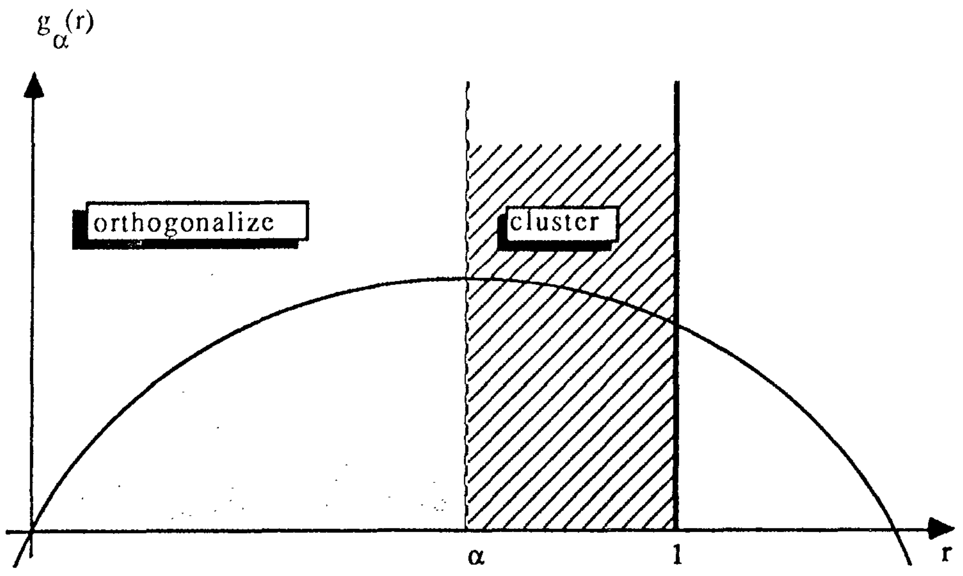


Figure 4.1: *The nonlinear function* $g_\alpha(\cdot)$

In order to develop a mathematically equivalent description of the constraints it is necessary to introduce some notation.

It is customary to think of presenting patterns from the set of input patterns to the network in a sequential manner with respect to some notion of time. Conceptually, though, one can think of the set of all input patterns as a large fixed vector yielding a corresponding large fixed output vector when presented to the network. The reader is referred to table 3.1 for a summary of the classifier notation.

Let $y \in \mathbb{R}^{n_o n_p}$ represent the entire pattern output of the system, where $y$ is partitioned as

$$y = \Big( \left( y_{11}\ y_{12} \cdots\ y_{1n_o} \right)\ \cdots\ \ \cdots\ \Big( y_{n_p 1} \cdots\ y_{n_p(n_o-1)}\ y_{n_p n_o} \Big)\Big)^T \qquad (4.4.2)$$

Physical realities of the problem make it desirable to access specific pattern information embedded in $y$. Define the linear projection operator, $T_k \in \mathbb{R}^{n_o \times n_o n_p}$, as a pattern mask so that

$$T_k : \mathbb{R}^{n_o n_p} \mapsto \mathbb{R}^{n_o} \ni T_k y = y_k \triangleq \begin{pmatrix} y_{k1} \\ y_{k2} \\ \vdots \\ y_{kn_o} \end{pmatrix} \quad \forall k \in \mathcal{P} \tag{4.4.3}$$

Within this framework the constraints (C) on the solution space are now given as

- unit norm, i.e.,

$$\|T_k y\|_2 = 1 \quad \forall k \in \mathcal{P} \tag{4.4.4}$$

- a scalar multiple of a unit basis element, i.e.,

$$u_i{}^T(T_k y) \cdot u_j{}^T(T_k y) = 0 \quad \forall i \neq j,\ i,j \in \mathcal{O}, \quad \forall k \in \mathcal{P} \tag{4.4.5}$$

where $u_i = (\ 0\ 0\ \cdots \underbrace{1}_{ith} \cdots\ 0\ )^T$. [1]

It is easy to show that these constraints are equivalent to

$$\|T_k y\|_1 - \|T_k y\|_2 = 0 \quad \forall k \in \mathcal{P} \tag{4.4.6}$$

Hence, the constraints can be stated mathematically by the the two following equations

[1]Note that $u_i^T x = x_i$ for $u_i, x \in \mathbb{R}^n, i = 1, 2, \ldots n$, where $x_i \in \mathbb{R}$ is the $i$th component of $x$.

$$\text{(C)} \quad \begin{cases} 1 - \|T_k y\|_2 = 0 \\ 1 - \|T_k y\|_1 = 0 \end{cases} \quad \forall k \in \mathcal{P}. \tag{4.4.7}$$

## 4.5 Initialization

It is clear that the question of parameter initialization is a key element in the solution. That is, how should the initial value of the parameter vector, $z_0$, be chosen? Recall that it is desirable to perform the classification based on the input pattern space. Yet, the derivations proceed as if the classification is to be done on the output space (leading to the unsupervisibility of the algorithm). Such an approach is justified by the intuitive notion of continuity that like input patterns should yield like output patterns. Still it is desirable to achieve an initial output mapping which preserves the similarity, $r(\cdot,\cdot)$, in input patterns. An exact solution to this problem is in general not possible due to the sigmoidal nonlinearity [2]. In lieu of an exact solution one may search for a 'best' solution once again in the form a minimization problem. In other words, one would like to pick $z_0$ so that the following minimization is achieved:

$$\min_{z} \sum_{j\in\mathcal{P}} \sum_{k\in\mathcal{P}} \left[r(x_j, x_k) - r\left(H_z(x_j), H_z(x_k)\right)\right]^2 \tag{4.5.1}$$

[2]The dilemma would be simply solved by estimating the sigmoidal as a truncated ramp function and with $z_0 = (W, b)$ chosen appropriately.

where $H_z(\cdot) : \mathbb{R}^{n_i} \mapsto [0,1]^{n_o}$ is the vector transfer function of parameter associator as given in equation 2.3.2. Unfortunately, this route to a solution is usually implausible. The problem is compounded by the fact that in general the patterns $\{x_j\}_{j\in\mathcal{P}}$ are not known apriori. In the two layer case when $n_o = n_i$ it is clear that a solution to the problem may be achieved by choosing $z_o$ so that the sigmoidal best approximates the identity mapping with coupling between layers limited only to a one to one correspondence. For example choosing $z_0 = (W, b)$ so that

$$W = 4I, \quad b = (-2\ -2\ \cdots\ -2\ )^T \tag{4.5.2}$$

would achieve such a requirement. Such an initialization scheme, though, has the drawback that it requires the number of inputs and outputs to be equal. Such a requirement is necessary even in the linear case when $H(x) = Wx$, $(b = 0)$ with $W \in \mathbb{R}^{n_i \times n_o}$. In this case the similarity between outputs, $y = H(x)$, is given as

$$r\left(H(x_j), H(x_k)\right) = \frac{H^T(x_j)H(x_k)}{\|H(x_j)\|_2\,\|H(x_k)\|_2} = \frac{x_j^T W^T W x_k}{\|Wx_j\|_2\,\|Wx_k\|_2}. \tag{4.5.3}$$

With $n_o < n_i$ and $W$ having full row rank, rank$(W) = n_o$, then rank$(W^TW) = n_o$ and $W^TW \in \mathbb{R}^{n_i \times n_i}$ is singular. It is clear that $W^TW = I_{n_i}$ would serve nicely in equation 4.5.3; however, this is clearly impossible since $W^TW \in \mathbb{R}^{n_i \times n_i}$ is singular. So it is seen that even in the linear case it is required that $n_o \geq n_i$ if the initial mapping is to preserve similarity.

In typical classification problems the number of distinct classes in the solution is much smaller than the dimension of the pattern space, i.e. $n_o < n_i$.

Because of this fact the requirement that $n_o \geq n_i$ is computationally wasteful. Alternatively, one may require that $W = (w_i)_{i=1}^{n_o}$ be initialized randomly with a uniform distribution having $E\left(\|w_i\|_2\right) = 1,\ \forall i$. Because of the randomness of $W$ it is expected that

$$\|w_i\|_2^2 \gg \langle w_i, w_j \rangle\,, \quad \forall i \neq j. \tag{4.5.4}$$

It is clear that this weakened condition falls short of a more desirable (yet impossible) orthogonalization property. For this reason the initialization given by equation 4.5.4 is dubbed a 'pseudo-orthogonal' one.

## 4.6 Constraint Satisfaction: A Penalty Method

One may attempt to achieve constraint satisfaction through the addition of a penalty function in the objective. This penalty term is chosen so that violation of the constraints adds a positive non-zero penalty to the objective function while constraint satisfaction yields no penalty or a penalty of zero.

A candidate for the penalty function, $P(y)$, for the constraints given in 4.4.7 is

$$P(y) = \frac{1}{2} \sum_{p \in \mathcal{P}} \left(1 - \|T_p y\|_2\right)^2 + \left(1 - \|T_p y\|_1\right)^2 \tag{4.6.1}$$

Note that the penalty function is nonnegative and equal to zero if and only if the constraints are satisfied. That is

$$P(y) \ \geq \ 0, \tag{4.6.2}$$

$$P(y) \ = \ 0 \iff \text{Constraint conditions (C) in 4.4.7 hold.} \tag{4.6.3}$$

In other words $P(\cdot)$ has global minima at exactly those points of constraint satisfaction.

Attention is now turned to the computation of the gradient of $E_\perp$. This computation requires knowledge of the gradient of $r$ with respect to $y_p$. Computation of this gradient is provided in appendix A. First note that with $p \in \mathcal{P}$

$$\frac{\partial E_\perp}{\partial y_{pi}} = \sum_{k \in \mathcal{P}} \frac{\partial g}{\partial r_{kp}}(r_{kp}) \cdot \frac{\partial r_{kp}}{\partial y_{pi}} \tag{4.6.4}$$

which written in vector notation gives

$$\begin{aligned}
\nabla_{y_p} E_\perp &= \sum_{k \in \mathcal{P}} \frac{\partial g}{\partial r_{kp}}(r_{kp}) \cdot \nabla_{y_p} r_{kp} && (4.6.5)\\
&= \frac{1}{\|y_p\|}\left(I - \bar{y}_p \bar{y}_p^T\right) \sum_{k \in \mathcal{P}} \frac{\partial g}{\partial r_{kp}}(r_{kp}) \bar{y}_k && (4.6.6)\\
&= \frac{1}{\|y_p\|}\left(I - \bar{y}_p \bar{y}_p^T\right) \sum_{k \in \mathcal{P}} 2(\alpha - \bar{y}_p^T \bar{y}_k) \bar{y}_k && (4.6.7)\\
&= \frac{2}{\|y_p\|}\left(I - \bar{y}_p \bar{y}_p^T\right) \sum_{k \in \mathcal{P}} \alpha \bar{y}_k - \left(\bar{y}_k \bar{y}_k^T\right) \bar{y}_p && (4.6.8)\\
&= \frac{2}{\|y_p\|} B(\bar{y}_p) \left[\, \alpha p(\bar{y}) - S(\bar{y})\, \bar{y}_p \,\right] && (4.6.9)
\end{aligned}$$

with the following definitions

$$\begin{aligned}
S(\bar{y}) &\triangleq \sum_{k \in \mathcal{P}} \bar{y}_k \bar{y}_k^T && (4.6.10)\\
p(\bar{y}) &\triangleq \sum_{k \in \mathcal{P}} \bar{y}_k && (4.6.11)\\
B(\bar{y}_p) &\triangleq (I - \bar{y}_p \bar{y}_p^T) && (4.6.12)\\
A(\bar{y}_p) &\triangleq I - B(\bar{y}_p) = \bar{y}_p \bar{y}_p^T && (4.6.13)
\end{aligned}$$

For notational convenience the explicit dependence of $B(\bar{y}_p)$ and $A(\bar{y}_p)$ as defined above may sometimes be neglected by writing simply $B_p$ and $A_p$ respectively.

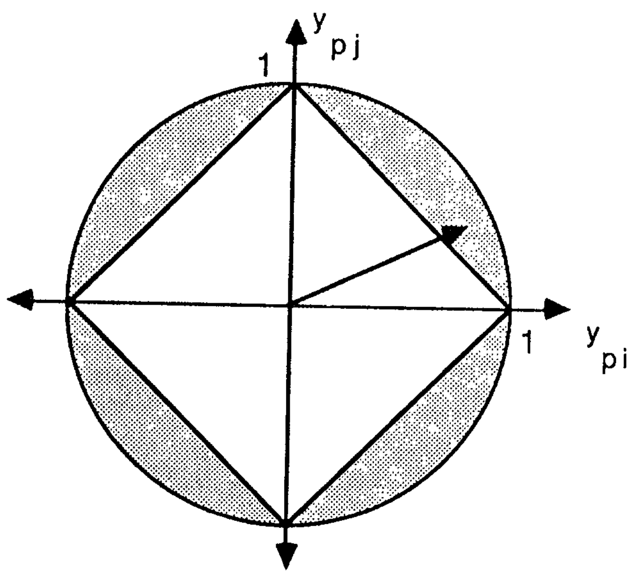


Figure 4.2: *Hyper region to which outputs are restricted when $P'(y) = 0$*

Let $P'(y) \triangleq \frac{\partial P}{\partial y_{pi}}$, then

$$P'(y) = (\|T_p y\|_1 - 1) + (\|T_p y\|_2 - 1)\bar{y}_{pi}. \tag{4.7.15}$$

By the fact that $\|x\|_2 \leq \|x\|_1 \quad \forall x \in \mathbb{R}^n$,

$$(1 + \bar{y}_{pi})(\|T_p y\|_2 - 1) \leq P'(y) \leq (\|T_p y\|_1 - 1)(1 + \bar{y}_{pi}). \tag{4.7.16}$$

Thus,

$$\|T_p y\|_2 \leq \frac{P'(y)}{1 + \bar{y}_{pi}} + 1 \leq \|T_p y\|_1 \tag{4.7.17}$$

Setting $P'(y) = 0$ yields the result. □

The lemma says that should $P(y)$ have a zero derivative with respect to any output then the output vector must lie in the 'hyper-wedge' region depicted in figure 4.2.

**Theorem 4.1** *With*

$$P(y) = \frac{1}{2}\sum_{p\in\mathcal{P}} \left(1 - \|T_p y\|_2\right)^2 + \left(1 - \|T_p y\|_1\right)^2, \tag{4.7.18}$$

*then*

$$\nabla_{y_p} P = 0 \implies \begin{cases} \|T_p y\|_2 = \|T_p y\|_1 = 1 \text{ , i.e. (C) holds} \\ \text{or} \\ T_p y = \alpha(\ 1\ 1 \cdots\ 1\ )^T,\ \alpha \in \mathbb{R} \text{ fixed} \end{cases}. \tag{4.7.19}$$

**Proof**

Assume $\nabla_{y_p} P = 0$. If $\|T_p y\|_2 = 1$ then $\|T_p y\|_1 = 1$ and (C) holds.

Now suppose that $\|T_p y\|_2 \neq 1$ so that $\|T_p y\|_2 < 1$ by lemma 4.1. Since $\nabla_{y_p} P = 0$ by assumption, it is true that

$$\frac{\partial P}{\partial y_{pi}} = (\|T_p y\|_1 - 1) + (\|T_p y\|_2 - 1)\bar{y}_{pi} = 0, \quad \forall i \in \mathcal{O}. \tag{4.7.20}$$

This implies that

$$\bar{y}_{pi} = \frac{\|T_p y\|_1 - 1}{1 - \|T_p y\|_2}, \quad \forall i \in \mathcal{O}. \tag{4.7.21}$$

Since the right side of this equation is independent of $i$, it must be concluded that $T_p y$ is a vector with equal components. □

In view of theorem 4.1 one would expect that the inclusion of the penalty term, $P(\cdot)$, in the energy function insures that a descent on the energy will go to solutions where the constraints are satisfied since $E_\perp$ is concave. $E_\perp$ is plotted in figure 4.3 as a function of $r_{01}$, and $r_{12}$, for the case of only three patterns

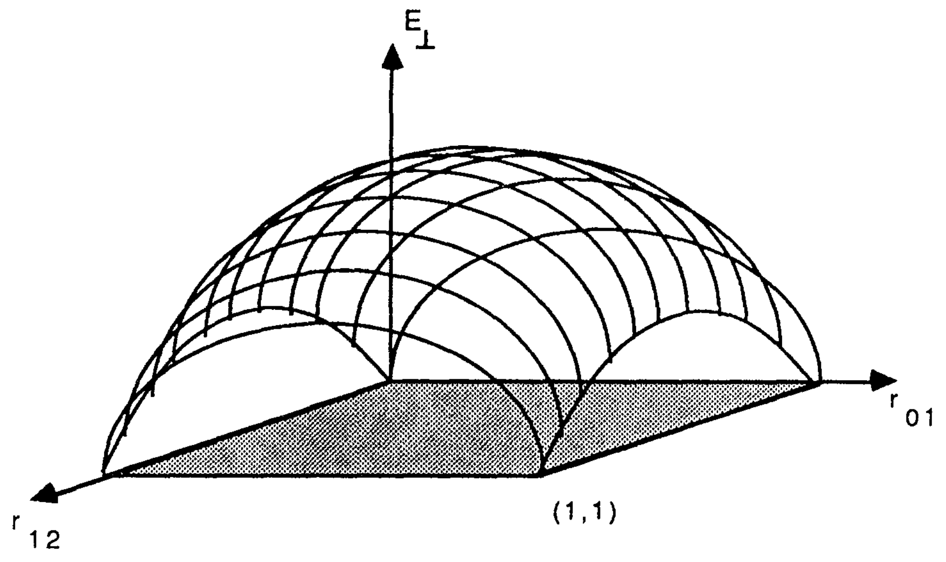


Figure 4.3: *The function $E_\perp$ for $n_p = 3$.*

$(n_p = 3)$ in the training set. Note that the gradient of $E$ is not zero almost everywhere on $(0,1)^{n_o}$.

Crucial to the computation of the gradient equation 4.6.9 is the matrix quantity $\bar{y}_k \bar{y}_k^T$. Therefore, properties of matrices of the form $A \triangleq xx^T$ become of great interest. First, note that $A$ is the projection operator; i.e., if $y = Az$, with $y, z \in \mathbb{R}^n$ then $y$ is the projection of $z$ onto $x$. This relationship is depicted in figure 4.4.

**Fact 4.1** *Suppose $x, y \in \mathbb{R}^n$, with $\|x\|_2 = 1$ and $A = xx^T \in \mathbb{R}^{n \times n}$ then the following statements are true:*

$$\operatorname{tr}(A) = x^T x = 1 \in \sigma(A) \text{ with eigenvector } x \tag{4.7.22}$$

$$0 \in \sigma(A) \text{ with multiplicity n-1} \tag{4.7.23}$$

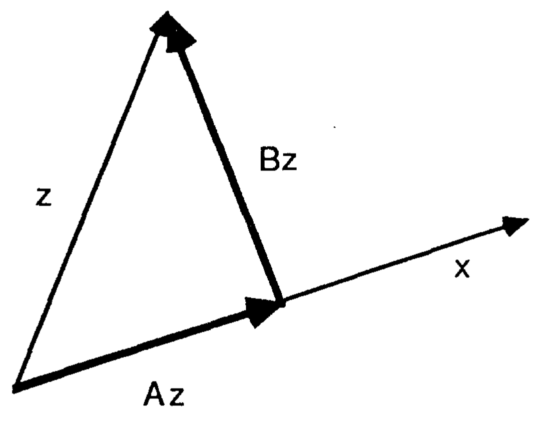


Figure 4.4: *The projection and parallel operators, A, and B.*

$$Az = 0 \iff z \perp x \tag{4.7.24}$$

$$A = A^T \tag{4.7.25}$$

$$A \geq 0 \tag{4.7.26}$$

$$A^k = A, \quad k = 1, 2, 3, \ldots \tag{4.7.27}$$

$$\begin{gathered} \exists J, P \in \mathbb{R}^{n \times n},\ P \text{ unitary } \ni \\ J = P^T A P = \text{diag}(\underbrace{0\ 0 \cdots\ 0}_{n-1}\ 1\ ) \end{gathered} \tag{4.7.28}$$

The columns of $P$ are the normalized eigenvectors of $A$.

Fact 4.1 leads to the following sister statements about $B \triangleq I - A$.

**Fact 4.2** *Suppose $x, y \in \mathbb{R}^n$, with $\|x\|_2 = 1$ and $B = I - xx^T \in \mathbb{R}^{n \times n}$ then the following statements are true:*

$$0 \in \sigma(B) \text{ with multiplicity } 1 \tag{4.7.29}$$

$$1 \in \sigma(B) \text{ with multiplicity n-1} \tag{4.7.30}$$

$$Bz = 0 \iff z \parallel x \tag{4.7.31}$$

$$B = B^T \tag{4.7.32}$$

$$B \geq 0 \tag{4.7.33}$$

$$B^k = B, \qquad k = 1, 2, 3, \ldots \tag{4.7.34}$$

$$\begin{gathered} \exists J, P \in \mathbb{R}^{n \times n}, \;\; P \text{ unitary } \ni \\ J = P^T B P = \operatorname{diag}( \underbrace{1 \; 1 \cdots \; 1}_{n-1} \; 0 \; ) \end{gathered} \tag{4.7.35}$$

The columns of $P$ are the normalized eigenvectors of $B$. Note that the Jordan form of $B$ indicates that there are $(n - 1)$ linearly independent eigenvectors associated with the eigenvalue one.

## 4.8 Local Minimization: A Descent Strategy

As discussed earlier, descent methods are attractive as a numerical optimization techniques. If applied directly to the task of minimizing the designed energy function of the previous section, however, the numerical complexity of the task becomes overbearing.

This section investigates the plausibility of using estimates of quantities which are computationally intensive. Specifically it is desirable to estimate the sums, $S(\bar{y})$ and $p(\bar{y})$, given by equations 4.6.10 and 4.6.11 respectively.

### 4.8.1 Numerical Complexity

Computation of both the quantities $S(\bar{y})$ and $p(\bar{y})$ in equations 4.6.10 and 4.6.11 presents problems to a real-time pattern classifier. Both quantities require knowledge of the entire set of outputs for the current state of the system for their computation. If the system is to operate on a per pattern basis delays (memory) are required as well as on the order of $(n_p - 1)$ times the number of computations for just one pattern. Further, knowledge of the number of patterns present is needed *apriori.* Clearly, for a pattern classifier system to be practical, this requirement must be removed.

This section deals with the problem of removing the above requirement of complete output knowledge. Indeed, it is shown that the requirement can be completely removed without affecting the global descent properties of the original algorithm.

### 4.8.2 Reduction of Numerical Complexity

Two methods of reduction in numerical complexity are employed. First, quantities which are nonessential with respect to the convergence of the general algorithm are simply dispensed in some appropriate manner. This method of reduction is termed annihilation. Second, other important quantities which are numerically expensive are estimated while still preserving the overall descent properties of the algorithm.

The following shows that in gradient equation 4.6.9 the term $\frac{1}{\|y_p\|}B(\bar{y}_p)$ may be ignored, while the terms $S(\bar{y})$ and $p(\bar{y})$ can be estimated.

## Annihilation

It is desirable to show that the term $\frac{1}{\|y_p\|}B(\bar{y}_p)$ in equation 4.6.9 is negligible with respect to descent. To this end the following theorems are established.

**Theorem 4.2** *Suppose that $f : \mathbb{R}^n \mapsto \mathbb{R}$ is continuously Fréchet differentiable, and $\exists A \in \mathbb{R}^{n \times n}$, $A > 0$ and $p \in \mathbb{R}^n$ so that*

$$\nabla f(x) = Ap \tag{4.8.1}$$

*then $d \triangleq -p$ is a descent direction for $f(x)$.*

**Proof**

Since $f(\cdot)$ is continuously Fréchet differentiable, there exists a Taylor expansion of $f(\cdot)$ around $x$ with $t \in \mathbb{R}^+$ which gives

$$\begin{aligned} f(x+td) &= f(x) + t\,\langle \nabla f(x), d\rangle + o(t) && (4.8.2) \\ &= f(x) + t\,\langle Ap, -p\rangle + o(t) && (4.8.3) \end{aligned}$$

where by definition of $o(t)$,

$$\lim_{t \downarrow 0} \frac{o(t)}{t} = 0. \tag{4.8.4}$$

Since $A > 0$ then $\langle Ap, p\rangle > 0$ by definition. For $t > 0$ small enough in 4.8.3

$$f(x+td) - f(x) = t\,\langle Ap, -p\rangle + o(t) < 0. \tag{4.8.5}$$

Hence $d = -p$ is a descent direction. $\square$

**Lemma 4.2** *Suppose that $x, y \in \mathbb{R}^n$, $y \neq 0$, with $\|x\|_2 = 1$ and $B = I - xx^T \in \mathbb{R}^{n \times n}$. If $\langle y, By \rangle = 0$ then $y \in \mathcal{N}(B)$.*

<u>**Proof**</u>

Assume $\langle y, By \rangle = 0$.

By virtue of fact 4.2 $\exists P$ unitary with columns denoted as $p_k$, $k = 1, 2, \dots, n$ so that $p_n \in \mathcal{N}(B)$ and

$$u_i^T P^T B P u_j = p_i^T B p_j = \begin{cases} 1, & i = j \neq n \\ 0, & \text{otherwise} \end{cases} \qquad i = 1, 2, \dots, n \tag{4.8.6}$$

Since $\{p_k\}_{k=1}^n$ spans $\mathbb{R}^n$, $\exists b \in \mathbb{R}^n$ with components $b_k, k = 1, 2 \dots, n$ such that

$$y = \sum_{k=1}^{n} b_k p_k, \quad b \neq 0 \tag{4.8.7}$$

Write

$$\begin{aligned} \langle y, By \rangle &= \left( \sum_{k=1}^{n} b_k p_k^T \right) B \left( \sum_{j=1}^{n} b_j p_j \right) && (4.8.8) \\ &= \sum_{k=1}^{n-1} \sum_{j=1}^{n-1} b_k b_j p_k^T B p_j && (4.8.9) \\ &= \sum_{k=1}^{n-1} b_k^2 = 0 \text{ by assumption.} && (4.8.10) \end{aligned}$$

But this can be true if and only if $b_k = 0 \;\; \forall k = 1, 2, \dots, n-1$. So this implies that

$$y = b_n p_n, \quad b_n \neq 0 \tag{4.8.11}$$

since $y \neq 0$. Since $p_n$ was chosen in $\mathcal{N}(B)$ then $y \in \mathcal{N}(B)$. $\square$

**Theorem 4.3** *Suppose that $f : \mathbb{R}^n \mapsto \mathbb{R}$ is continuously Fréchet differentiable and $x, p \in \mathbb{R}^n$ so that*

$$\nabla f(x) = \left(I - xx^T\right) p \tag{4.8.12}$$

*then either $d \triangleq -p$ is a descent direction for $f(x)$ or $\nabla f(x) = 0$.*

**Proof**

Since $\left(I - xx^T\right) \geq 0$ then either

1. $\left\langle p, \left(I - xx^T\right) p\right\rangle > 0$ or
2. $\left\langle p, \left(I - xx^T\right) p\right\rangle = 0$.

if case 1 is true then $d \triangleq -p$ is a descent direction following an identical argument as in the proof of theorem 4.2. This leaves case 2. But by lemma 4.2 $p \in \mathcal{N}\left(I - xx^T\right)$ and hence $\nabla f(x) = \left(I - xx^T\right) p = 0$. $\square$

The above theorem shows that the term $\frac{1}{\|y_p\|} B(\bar{y}_p)$ is negligible with respect to descent.

## Estimation

Through the process of estimation, a great reduction in numerical complexity can be realized. It is shown that employing simple first order recursive estimation is sufficient to generate 'good' descent directions.

It is desirable to provide estimators for a sequence of the form

$$s^{(n)} = \frac{1}{n_p} \sum_{k \in \mathcal{P}} q_k^{(n)}, \quad q_k^{(n)} \in [0,1], \ \forall k \in \mathcal{P}, \ \forall n \tag{4.8.13}$$

It is clear that sum in equation 4.8.13 is dependent on outputs due to every pattern in $\mathcal{P}$; however, at any specific time instant, $n$, only knowledge of the past outputs is possible. Moreover, the computational complexity and memory requirements make it implausible to explicitly retain all past information. This suggests that some sort of recursive estimation scheme may hold the solution. By assumption there are only a fixed number of patterns, $n_p$, that will be presented to the network. In view of this assumption, the situation is modeled as one where at a specific time instant a pattern is chosen randomly by nature (with a uniform distribution on $\mathcal{P}$) to be presented to the network. This will be made more precise in the following.

Note that the above equation 4.8.13 is a scalar equation; however, both the quantities $S(\bar{y})$ and $p(\bar{y})$ given by equations 4.6.10 and 4.6.11 respectively, though not scalar, have elements which are of this form. For ease of analysis the estimation is envisioned to be done on an element per element basis. Note also that the sequence $\{s^{(n)}\}_1^\infty$ is bounded on $[0,1]$ and hence there exists for sure a converging subsequence, i.e. $\exists s \in [0,1], \ \mathbb{N}' \subset \mathbb{N}$ so that

$$\lim_{\substack{n \in \mathbb{N}' \\ n \to \infty}} s^{(n)} = s. \tag{4.8.14}$$

In fact, one may argue that the incorporation of the penalty term, $P(y)$, will force all subsequences of $\{s^{(n)}\}$ to converge to the same $s$; in other words the convergence will be on all of $\mathbb{N}$. This is because the penalty term leads to solutions where the constraints $(C)$ are satisfied, which in turn implies that the elements of the summation, $q_k^{(n)}$, will converge to either zero or one:

$$\lim_{n\to\infty} q_k^{(n)} = q_k \in \{0,\ 1\} \quad \forall k \in \mathcal{P}. \tag{4.8.15}$$

Pursuing the idea of a recursive estimation scheme, let $\tilde{s}^{(n)}$ denote an estimate of $s^{(n)}$ where the estimate is given as

$$\begin{cases} \tilde{s}^{(n)} = d \cdot \tilde{s}^{(n-1)} + g \cdot q_{I_n}^{(n)} \\ \tilde{s}^{(0)} = s^{(0)} \end{cases} \tag{4.8.16}$$

where $d \in (0,1)$ and $g \in \mathbb{R}$. It will be seen in the sequel that $g = 1 - d$ is the desired value for $g$. Here, $I_n$ is a discrete random variable which takes values in the set $\mathcal{P}$, i.e.

$$\sum_{j\in\mathcal{P}} \Pr(I_n = j) \ = \ 1, \quad \forall n. \tag{4.8.17}$$

It should be noted that for each $n$ an independent random experiment is performed yielding a sample in $\mathcal{P}$ which serves as the index of $q$. With the set of random variables, $\{I_k\}_{k=1}^{n}$, independent and identically uniformly distributed over $\mathcal{P}$ the joint probability distribution is given as

$$\Pr(\ I_1 = i_1,\ I_2 = i_2,\ \cdots,\ I_n = i_n\ ) = \begin{cases} \left(\frac{1}{n_p}\right)^n, & i_k \in \mathcal{P},\ \forall k = 1, 2, \ldots, n \\ 0, & \text{else} \end{cases} \tag{4.8.18}$$

For an estimate to be useful it is necessary that it be a somewhat faithful representation of the quantity being estimated. We should expect that as time progresses the estimation process will become increasingly more accurate. More precisely, it is desirable to show that the estimator $\tilde{s}^{(n)}$ approaches $s^{(n)}$ as $n$ becomes large.

**Theorem 4.4** *Let $s^{(n)}$ and $\tilde{s}^{(n)}$ be as in equations 4.8.13 and 4.8.16 respectively. Further assume that the set of random variables, $\{I_k\}_{k=1}^{n}$, are independent and identically uniformly distributed. Then*

$$\lim_{n\to\infty} E\left(\tilde{s}^{(n)} - s^{(n)}\right) = 0, \tag{4.8.19}$$

*provided $g = 1 - d$.*

<u>**Proof**</u>

In view of equation 4.8.15, given an $\epsilon > 0$ there is a function $K(\cdot)$ so that $\left|s^{(n)} - s\right| < \epsilon \quad \forall n > K(\epsilon)$. For ease of analysis take $\tilde{s}^{(0)} = 0$ and suppose that $g = 1 - d$.

$$\begin{aligned}
&\left|E\left(\tilde{s}^{(n)} - s^{(n)}\right)\right| \\
&= \left|g\sum_{k=1}^{n} d^{n-k} E\left(q_{I_k}^{(k)}\right) - s^{(n)}\right| = \left|g\sum_{k=1}^{n} d^{n-k} s^{(k)} - s^{(n)}\right| \\
&\leq g\sum_{k=1}^{n} d^{n-k}\left|s^{(k)} - s\right| + \left|g\sum_{k=1}^{n} d^{n-k} s - s^{(n)}\right| \\
&= g\sum_{k=1}^{K(\epsilon/2)} d^{n-k}\left|s^{(k)} - s\right| + g\sum_{k>K(\epsilon/2)}^{n} d^{n-k}\left|s^{(k)} - s\right| + \left|sg\sum_{k=1}^{n} d^{n-k} - s^{(n)}\right| \\
&< \underbrace{d^n\left(d^{-K(\epsilon/2)}g\right)\sum_{k=1}^{K(\epsilon/2)} d^{n-k}\left|s^{(k)} - s\right|}_{<M_\epsilon \in \mathbb{R}} + \frac{g\epsilon}{2}\sum_{k>K(\epsilon/2)}^{n} d^{n-k} + \left|sg\frac{1-d^n}{1-d} - s^{(n)}\right|
\end{aligned}$$

$$
\begin{aligned}
&< d^n M_\epsilon + \frac{\epsilon}{2} + \left|s\left(1-d^n\right) - s^{(n)}\right| \\
&< d^n M_\epsilon + \frac{\epsilon}{2} + \underbrace{\left|s^{(n)} - s\right|}_{<\frac{\epsilon}{2}} + d^n\left|s\right| \\
&< d^n\left(1+M_\epsilon\right) + \epsilon
\end{aligned}
$$

Taking limit as $n \to \infty$ yields

$$\lim_{n\to\infty} \left|E\left(\tilde{s}^{(n)} - s^{(n)}\right)\right| \leq \epsilon. \tag{4.8.20}$$

Since $\epsilon$ is arbitrary the result is established. $\square$

Introduce the estimators for the quantities in equations 4.8.13 and 4.8.16 as $\tilde{S}(\bar{y})$ and $\tilde{p}(\bar{y})$ respectively as

$$\tilde{S}^{(n)} = d\tilde{S}^{(n-1)} + (1-d)\,\bar{y}_I\bar{y}_I^T \tag{4.8.21}$$

$$\tilde{p}^{(n)} = d\tilde{p}^{(n-1)} + (1-d)\,\bar{y}_I. \tag{4.8.22}$$

The above theorem indicates that both these estimates are expected to have elements which asymptotically approach the value being estimated. One may conclude from this element convergence that

$$\lim_{n\to\infty} E\left(\left\|\tilde{S}^{(n)} - S^{(n)}\right\|\right) = 0 \tag{4.8.23}$$

$$\lim_{n\to\infty} E\left(\left\|\tilde{p}^{(n)} - p^{(n)}\right\|_2\right) = 0. \tag{4.8.24}$$

The norm in the top equation is an appropriate matrix norm (i.e. the matrix sup norm) while the norm in the bottom equation is the usual $L_2$ norm.

The following facts concerning the relationship between some norms and the inner product will be useful.

**Fact 4.3** *Let $x, y \in \mathbb{R}^n$, then the following statements are true:*

$$|\langle x, y\rangle| \leq \|x\|_\infty \cdot \|y\|_1 \tag{4.8.25}$$

$$\|x\|_\infty \leq \|x\|_2 \tag{4.8.26}$$

These facts are employed in the following key theorem.

**Theorem 4.5** *Suppose that $f : \mathbb{R}^m \mapsto \mathbb{R}$ is continuously Fréchet differentiable, $y, p^{(n)} \in \mathbb{R}^m$, and $B, S \in \mathbb{R}^{m\times m}$, $B \triangleq \left(I - \bar{y}\bar{y}^T\right) \geq 0$ and $\nabla f(y) = B(\alpha p^{(n)} - S^{(n)}y)$ with $\alpha \in (0,1)$. If $\widetilde{S}^{(n)}$ estimates $S^{(n)}$ and $\widetilde{p}^{(n)}$ estimates $p^{(n)}$ so that equations 4.8.23 and 4.8.24 hold then $\exists\ K \in \mathbb{N} \ni d = -(\alpha\widetilde{p}^{(n)} - \widetilde{S}^{(n)}y)$ is an expected descent direction or $\nabla f(y) = 0\ \forall n > K$.*

**Proof**

Let

$$\begin{aligned} w^{(n)} &\triangleq \alpha p^{(n)} - S^{(n)}y, && (4.8.27)\\ \tilde{w}^{(n)} &\triangleq E\left(\alpha\widetilde{p}^{(n)} - \widetilde{S}^{(n)}y\right). && (4.8.28)\end{aligned}$$

Conditions 4.8.23 and 4.8.24 clearly imply that

$$\lim_{n\to\infty} \left\|\tilde{w}^{(n)} - w^{(n)}\right\|_2 = 0. \tag{4.8.29}$$

Case 1: If $w^{(n)}$ is such that $\left\langle w^{(n)}, Bw^{(n)}\right\rangle = 0$ then $\nabla f = Bw^{(n)} = 0$ by theorem 4.3.

Case 2: Now assume that $\left\langle w^{(n)}, Bw^{(n)} \right\rangle > 0$. Write

$$\begin{aligned} \left\langle \tilde{w}^{(n)}, \nabla f(y) \right\rangle &= \left\langle \left(\tilde{w}^{(n)} - w^{(n)}\right), \nabla f(y) \right\rangle + \left\langle w^{(n)}, \nabla f(y) \right\rangle & (4.8.30) \\ &= \left\langle \left(\tilde{w}^{(n)} - w^{(n)}\right), \nabla f(y) \right\rangle + \gamma^2, \quad \gamma \neq 0 & (4.8.31) \end{aligned}$$

Since $\lim_{n\to\infty} \left\| \tilde{w}^{(n)} - w^{(n)} \right\|_2 = 0$, then $\exists$ a function $K(\cdot)$ so that

$$\left\| \tilde{w}^{(n)} - w^{(n)} \right\|_2 < \acute{\epsilon}, \quad \forall n > K(\acute{\epsilon}) \tag{4.8.32}$$

where $\acute{\epsilon} \triangleq \epsilon / \|\nabla f(y)\|_1 > 0$. Using fact 4.3 leads to

$$\begin{aligned} \left| \left\langle \left(\tilde{w}^{(n)} - w^{(n)}\right), \nabla f(y) \right\rangle \right| &\leq \left\| \left(\tilde{w}^{(n)} - w^{(n)}\right) \right\|_\infty \|\nabla f(y)\|_1 & (4.8.33) \\ &\leq \left\| \left(\tilde{w}^{(n)} - w^{(n)}\right) \right\|_2 \|\nabla f(y)\|_1 & (4.8.34) \\ &< \acute{\epsilon} \|\nabla f(y)\|_1 = \epsilon. & (4.8.35) \end{aligned}$$

Taking $n > K(\frac{\gamma^2}{\|\nabla f(y)\|_1})$ equation 4.8.31 insures that

$$\left\langle \tilde{w}^{(n)}, \nabla f(y) \right\rangle > 0 \tag{4.8.36}$$

and hence

$$d = -(\alpha \tilde{p}^{(n)} - \tilde{S}^{(n)} y) \tag{4.8.37}$$

is an expected descent direction at $y$ for $f(y)$. $\square$

## 4.9 The Parameter Update

With the vector $z$ denoting the parameter vector for the pattern associator, the remainder of this thesis uses the convention that $z$ is formed as the concatenation of the rows of the weight matrix, $W$, and the bias vector, $b$:

$$z = (\ w_{11}\ w_{12}\ \ldots\ w_{n_o n_i}\ b_1\ \ldots\ b_{n_o}\ )^T . \tag{4.9.1}$$

It is with respect to these parameters that the minimization must be performed. For this reason it is necessary to make explicit the formula for descent directions in theorem 4.5 as seen from the point of view of the parameter vector $z$.

**Fact 4.4** *In the case of a two layer network the parameter estimated descent directions are computed directly via*

$$d_z^{(n)} = -\sum_{i \in \mathcal{O}} \underbrace{\left[u_i^T \left(\alpha \tilde{p}^{(n)} - \tilde{S}^{(n)} y_p + \nabla_{y_p} P(y)\right)\right]}_{\widetilde{\frac{\partial E}{\partial y_{pi}}}} \nabla_z y_{pi}. \tag{4.9.2}$$

Extension to multi-layered (more than two) pattern associator networks can be accomplished in exactly the same manner as in the backpropagation scheme presented in chapter two. Essentially, this involves providing a recursive method to compute $\widetilde{\frac{\partial E}{\partial y_{pi}}}$.

Note that from theorems 4.5 and 4.1 and the fact that for nontrivial inputs $(x_k \neq 0)$,

$$\left\{ \begin{array}{l} \frac{\partial y_{pi}}{\partial z_k} \geq 0 \\ \exists\ i \in \mathcal{O} \ni \frac{\partial y_{pi}}{\partial z_k} > 0 \end{array} \right. \qquad \forall k \tag{4.9.3}$$

evaluated at points where the constraints (C) are not satisfied (see chapter 2). This fact guarantees that directions generated through equation 4.9.2 will not be driven to zero due to the partials of $y_p$ with respect to $z$. In other words, directions generated by equation 4.9.2 will be zero only in the case when directions generated by 4.8.37 are zero. This property discounts the event of a descent type algorithm terminating prematurely.

Established now is a proposition which asserts that the gradient of the energy as a function of the pattern associator parameter vector, $z = (W, b) \in \mathbb{R}^{n_o \times n_i} \times \mathbb{R}^{n_o}$, is equal to zero when the constraints (C) 4.4.7 are satisfied. In view of the previous theorem 4.5, this means that descent directions may be generated by the estimates given in 4.9.2.

**Proposition 4.1** *The constraints (C) as in equation 4.4.7 are satisfied* $\Longrightarrow$ $\nabla_z E = 0$.

**Proof**

Write

$$\frac{\partial E}{\partial z_k} = \frac{\partial E}{\partial y_{pi}} \frac{\partial n_{pi}}{\partial z_k} \cdot \frac{\partial y_{pi}}{\partial n_{pi}}, \tag{4.9.4}$$

where $y_{pi} = h(n_{pi})$ is the sigmoidal function and $n_{pi} = \sum_{j \in \mathcal{I}} w_{ij} x_j + b_i$ (refer to chapter two for further notation). Constraint satisfaction implies that $y_k \rightarrow \{0, 1\}^{n_o}$. Write

$$\lim_{y_{pi} \downarrow 0} \frac{\partial E}{\partial z_k} = \lim_{n_{pi} \rightarrow -\infty} \frac{\partial E}{\partial z_k} = \frac{\partial E}{\partial y_{pi}} \frac{\partial n_{pi}}{\partial z_k} \cdot \lim_{n_{pi} \rightarrow -\infty} \frac{\partial y_{pi}}{\partial n_{pi}} = 0, \tag{4.9.5}$$

as a result of the boundedness and monotonicity properties of the sigmoid. $\square$

**Proposition 4.2** *If the constraints (C) are satisfied then the estimated gradient*

$$\widetilde{\nabla}_z E = 0, \tag{4.9.6}$$

*where*

$$\frac{\widetilde{\partial E}}{\partial z_k} = \frac{\widetilde{\partial E}}{\partial y_{pi}} \frac{\partial y_{pi}}{\partial n_{pi}} \frac{\partial n_{pi}}{\partial z_k}. \tag{4.9.7}$$

The proof mimics that of the proposition 4.1 and is omitted.

At this point a remark is warranted. It has been seen through theorem 4.5 that directions, say $\left\{d^{(n)}\right\}$ generated from either equations 4.8.37 or 4.9.2 are *expected* descent directions. These directions inherit their random nature from the randomness in the selection of the patterns. What is meant by an 'expected' descent direction is precisely that $E\left(d^{(n)}\right)$ is a descent direction. There is no guarantee that a particular sample of the process $\left\{d^{(n)}\right\}$ will in fact be a descent direction; however, it seems likely that most of the directions $d^{(n)}$ will be descent directions since (i) $E\left(d^{(n)}\right)$ is a descent direction, and (ii) any direction which forms an acute angle (less than 90 degrees) with the gradient of the objective is also a descent direction. Then, we should expect that the objective will be decreased with the application of $d^{(n)}$ most of the time. Such an observation provides justification for an implementation of the type proposed here. It is left to the application portion of this thesis (see chapter 5) to actually demonstrate the merits of this approach.

## 4.10 The Learning Algorithm

Previous efforts are cast in the direction of providing a learning scheme for pattern classification. An energy minimization formulation of the problem has proven to be a fruitful one; provided that estimators of quantities depending on past inputs be incorporated. This section details the underlying descent algorithm that has been in mind all along. Until now there has been no mention as to the initial state of the parameter vector $z_0$. The question of initialization of the network is deferred until after a detailed description of the algorithm.

The main contribution of this thesis is the following learning algorithm.

**Algorithm 4.1**

```
1. compute $S^{(0)}$ and $p^{(0)}$, set $\tilde{S}^{(0)} = S^{(0)}$, $\tilde{p}^{(0)} = p^{(0)}$
2. Initialize $z_0 \in \mathbb{R}^{n_o \times n_i} \times \mathbb{R}^{n_o}$, set $k = 0$
3. while convergence criterion unreached
        (i) present a pattern $x_p$ to the network, denote $y_p$ as the output
        (ii) compute the gradient estimate, e.g. 2 layer case:
                $d_z^{(k)} = -\sum_i u_i^T \left(\alpha \tilde{p}^{(k)} - \tilde{S}^{(k)} y_p + \nabla_{y_p} P(y_p)\right) \nabla_z y_{pi}$
        (iii) perform pseudo descent update
                set $z_{k+1} = z_k + \gamma^2 d_z^{(k)}$
        (iv) perform estimate update
                a) set $\tilde{p}^{(k+1)} = d \cdot \tilde{p}^{(k)} + (1-d)\bar{y}_p$
                b) set $\tilde{S}^{(k+1)} = d \cdot \tilde{S}^{(k)} + (1-d) \cdot \bar{y}_p \bar{y}_p^T$
        (v) increment $k$
```

Presented in this chapter has been a learning algorithm aimed at solving the PCP which is computationally efficient and averts many of the drawbacks of the backpropagation method. Most predominant among these is that the learning is conducted in an unsupervised and continuous manner. It is continuous in the sense that learning and operation occur simultaneously. Moreover, explicit knowledge of the entire training set is not needed; hence, training and classification occur on a pattern per pattern basis.

CHAPTER

# FIVE

## Applications

Explored in this chapter are several applications of the proposed classifier system. In particular the update rule is given by the learning algorithm previously developed (algorithm 4.1).

The first 'application' is a simple experiment designed to demonstrate the controllability of the resolving power of the resulting segmentation. The remaining two applications are both acoustic ones. In the one case the algorithm is applied to the task of phoneme recognition, while in the other to the problem of pitch perception. As acoustic applications, the transformation stage of the system is taken to be the model of the cochlea described in chapter three. Clear statements of both these problems as well as the respective solutions determined by the algorithm are provided.

In each case the topology of the pattern associator neural network is constrained to have only two layers. This is due primarily to (see chapter three) the form of the chosen similarity function, the cosine function, in the algorithmic

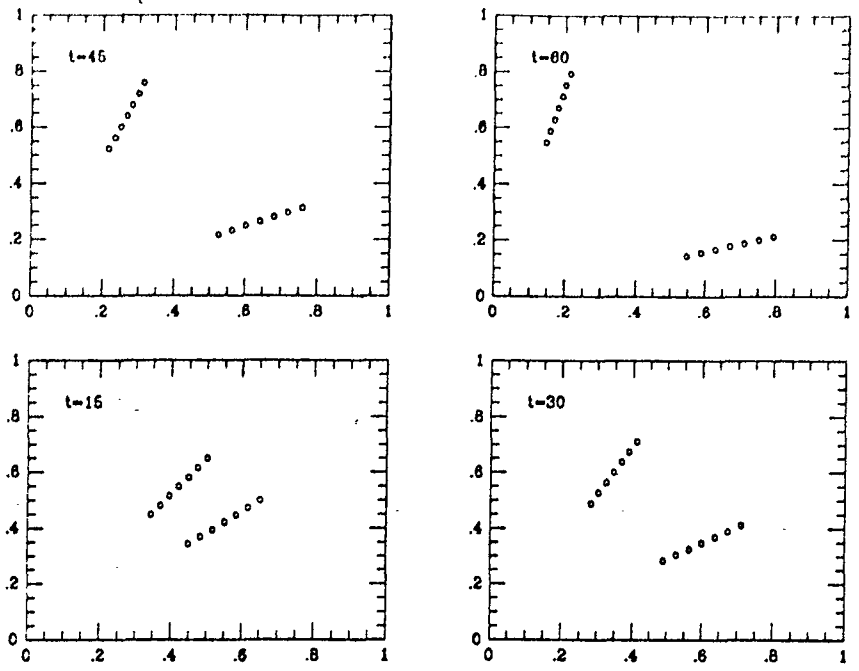


Figure 5.1: *Training patterns for the line separation experiment*

derivation.

## 5.1 Line Separation

It was previously claimed that the resolution of the resulting segmentation via the learning algorithm 4.1 could be controlled by the parameter $\alpha$ in equation 4.3.6. An experiment aimed at verifying this claim is detailed below and followed by some experimental results.

For this simple demonstration the topology of the feed forward network is taken to be simply a two input two output layered network (2:2). Initialization simply sets the weights and biases so that the pattern associator performs essentially the identity mapping 4.5.2. Thus the classification is to be performed on a two dimensional pattern space. A training set, $\mathcal{T}(\theta)$, of patterns is formed by taking patterns on two straight lines situated at an angle of $\theta$ with respect to each other. For simplicity, the straight lines are taken to be at an angle of $\pm\theta/2$ from the 45° line (see figure 5.1). With $R(\phi) : \mathbb{R}^2 \mapsto \mathbb{R}^2$ denoting the rotation (by $\phi$) operator, the training set may be written as

$$\mathcal{T}(\theta) \triangleq \left\{ x : x = a \cdot R(\phi) \begin{pmatrix} 0.5 \\ 0.5 \end{pmatrix}, \quad \phi \in \{\pm\theta/2\}, \quad a \in \{1 + k\Delta\}_{k=-N}^{N} \right\}, \tag{5.1.1}$$

where the pair $(\Delta, N)$ determine the spread and number of samples on each line.

Training sets with $\theta = 15°,\ 30°,\ 45°,\ 60°$ and $(\Delta, N) = (0.03,\ 3)$ are depicted in figure 5.1. These training sets were presented to the network for learning via algorithm 4.1 with the parameter, $\alpha$, taking on the values $\alpha = 0.5,\ 0.7,\ 0.9,\ 0.99$. A summary of the resulting segmentations are presented in table 5.1. The actual results of the experiment are given in appendix B in figures B.1, B.2, B.3, B.4 respectively.

Clearly, it is seen that there is a direct relationship between the resolution of the resulting classification and the parameter $\alpha$. Moreover, the results indicate that one may control the resolution of the final segmentation by picking $\alpha$ suitably. [1]

[1]It is not surprising that for no value of $\alpha$ is the classifier able to segment patterns forming an angle of 15° since (i) the similarity measure used to distinguish patterns is the cosine ($\cos(15°) = 0.966$) and (ii) the sigmoidal nonlinearity smears the input similarity, i.e. the initial mapping is not the identity.

| $\alpha$ | $\theta$ | separation |
|---|---|---|
| 0.5 | 15° | no |
|  | 30° | no |
|  | 45° | no |
|  | 60° | no |
| 0.7 | 15° | no |
|  | 30° | no |
|  | 45° | no |
|  | 60° | yes |
| 0.9 | 15° | no |
|  | 30° | no |
|  | 45° | yes |
|  | 60° | yes |
| 0.99 | 15° | no |
|  | 30° | yes |
|  | 45° | yes |
|  | 60° | yes |

Table 5.1: *Line separation summary*

## 5.2 Phoneme Classification

All speech may be decomposed into substructures known as phonemes [13], [14]. There is no question that the classification of phonemes is a crucial element in the overall goal of recognition of natural speech.

Employing the classifier system and associated learning algorithm ( 4.1 ), a two layer network is proposed as the pattern associator for the phoneme classification problem. The system is tested with the two different voiced numerals [2] , 'one' and 'six'. Experimental results correlate highly to intuitive expectations.

Determined by the output set of the cochlea model, less those bands which

[2]As spoken by Bill Byrne, Neural Systems Laboratory, UMD

never reach a threshold $t$ (taken to be $t = 0.2$), the number of units in the input layer may vary between words. In the case of the data 'one', a 49:5 two layer feed forward neural network is employed as the pattern associator, while in the case of the data 'six', a 57:10 architecture is used. The pattern associator parameter vector is initialized with a random pseudo-orthogonal set of weights and biases 4.5.4.

Simulation results for each word, 'one' and 'six', are depicted in figures 5.2 and 5.3 respectively. These figures depict the response to the entire pattern set generated by the cochlea model when trained on a *reduced* sample set. This reduced sample set is given in appendix B. Desiring a high degree of resolving power, the parameter $\alpha$ is set at $\alpha = 0.9$. Each figure displays the cochlea response to the respective word (top graph) while the lower group of five windows are representative of five corresponding output neurons of the pattern associator. In the case of the 57:10 topology the five remaining outputs are assigned to zero and not displayed in figure 5.3.

An explanation of the figures follows. Essentially a three dimensional plot, the cochlea response graph has a horizontal axis which represents time and a vertical axis which represents both frequency and amplitude information. The frequency axis is ordered (starting from the bottom of the graph and going upwards) in a decreasing fashion. In other words low frequencies are represented at the top of the graph and high frequencies at the bottom. At each frequency the amplitude response of the model is plotted as a function of time. Note that at each fixed time instant a pattern (across frequencies) emerges. Hence, for a finite set of sampling times a set of patterns results which may be labeled from one to the number of samples. This is the pattern axis of the figures. The remaining plots indicate the response of selected output neurons as a function of pattern (time).

## 5.3 Pitch Perception

One form of musical knowledge is represented in the conventional western musical scoring system developed hundreds of years ago. This system still remains the prevalent method of conveying musical information among musically literate humans. It has long been recognized that the conventional scoring system is deficient in the sense of providing a complete representation of a piece of music. This is demonstrated by a perfectly literal playing of a musical piece by computer. The resulting rendition invariably is perceived as highly artificial and mechanical. Hence, the problem of reconstructing a musical piece from its conventional musical representation is highly non-trivial. Also of interest is the inverse problem of generating a musical representation (not necessarily the conventional one) from the acoustic data generated from a musical piece. Hence, the perception of pitch plays a crucial role in the solution.

Pitch is not reserved strictly for sounds that are found in music but is an attribute of any complex sound. Studies in the perception of pitch [15], [16], [17],

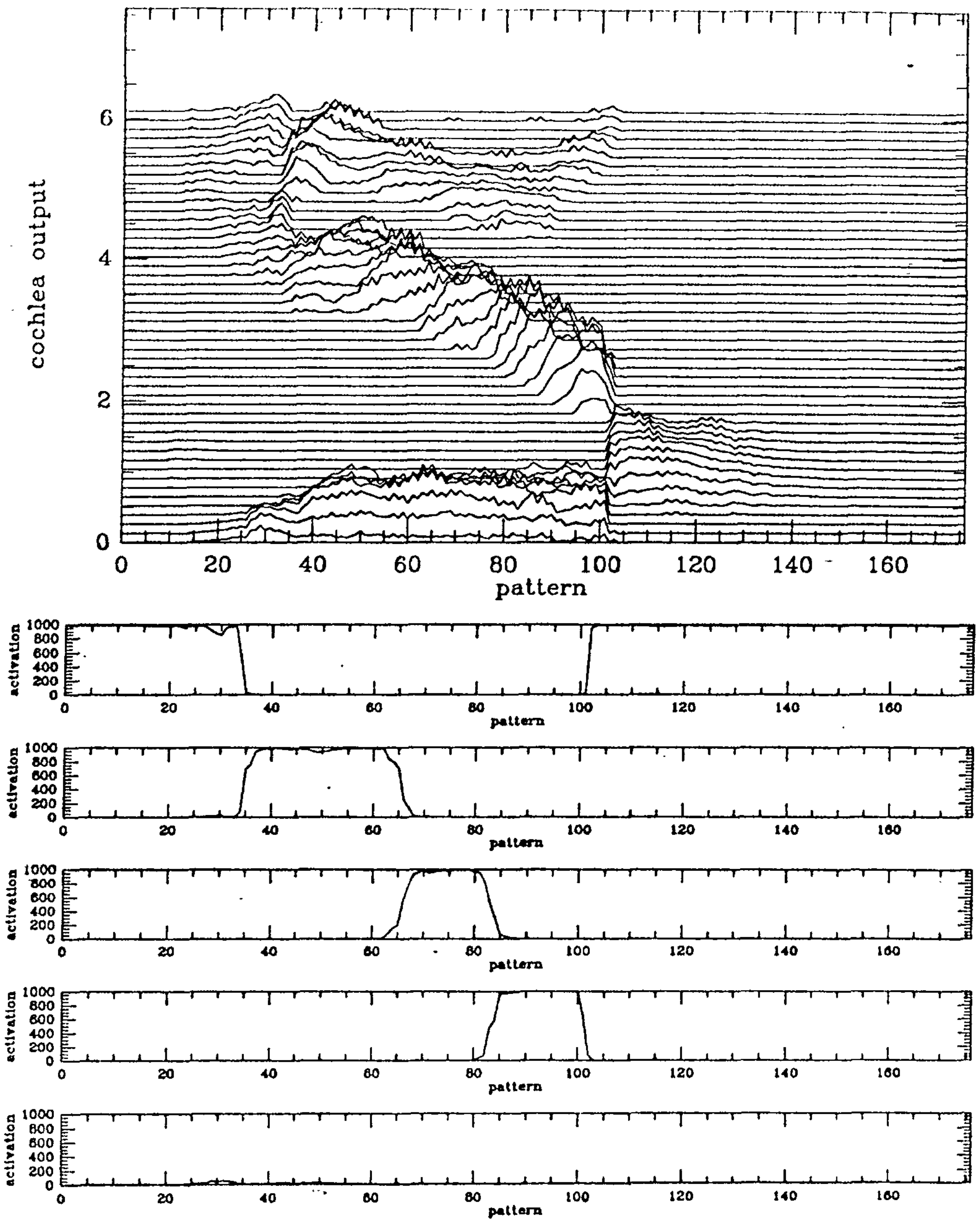


Figure 5.2: *Cochlea output and segmentation of the spoken word 'one'*

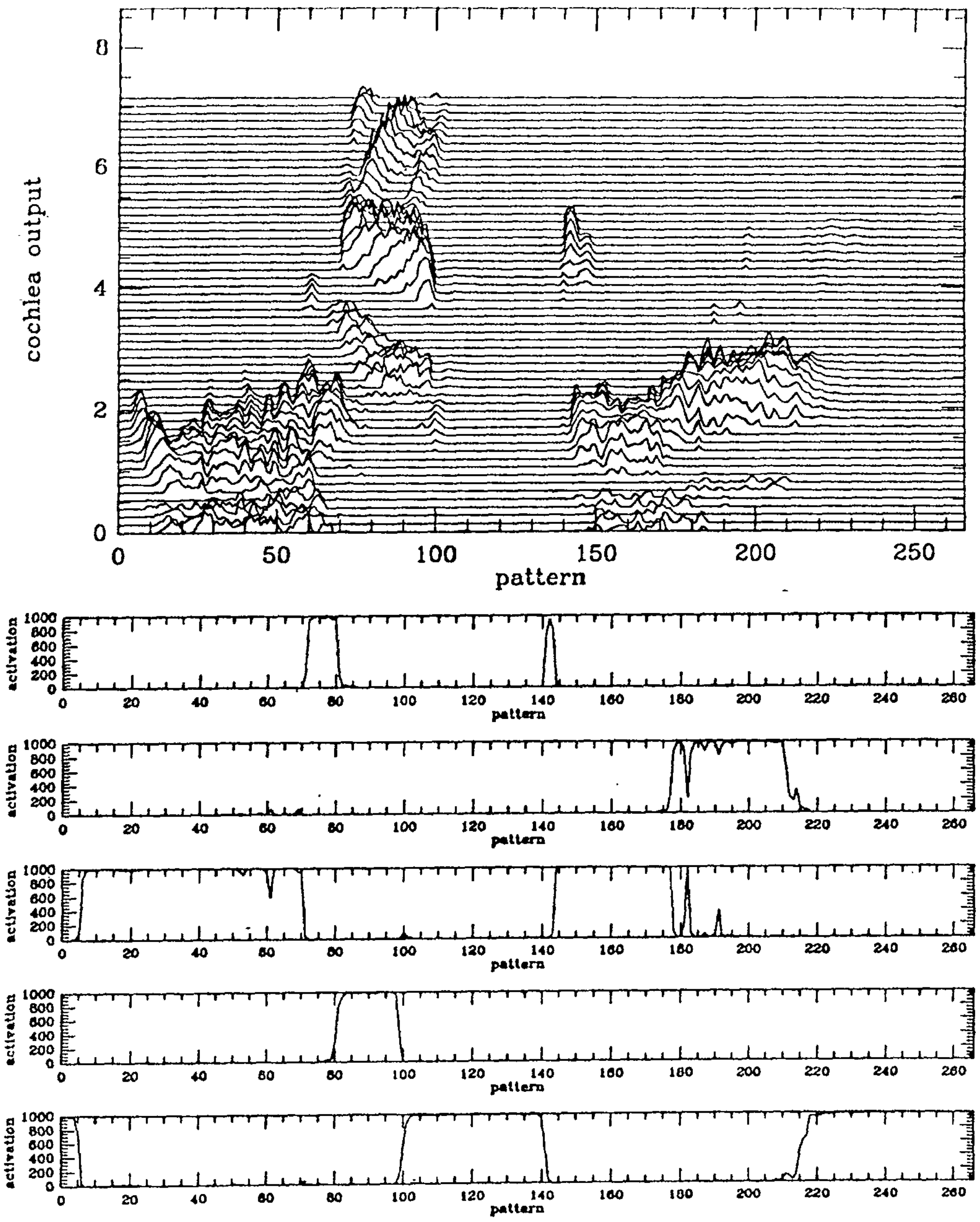


Figure 5.3: *Cochlea output and segmentation of the spoken word 'six'*

[18], [19], show that it is the low order harmonics which are important. Such a result supports the choice of the cochlea model as the sensory transformation process. This is due to the cochlea's model logarithmic structure (i.e. higher order harmonics are muddled together along the tonotopic axis).

The pitch perception problem is easily cast as a pattern classification problem. As such it presents itself as an excellent candidate for the classification system proposed here. Again a simple two layer feed forward neural architecture is employed as the pattern associator with the learning algorithm given by algorithm 4.1.

It is then experimentally verified that the system is capable of correctly learning to identify the pitch of monophonically sounding musical instruments in a timbre independent way. A **simple tone** is a single frequency sinusoidal sound wave. A **complex tone** or **complex sound** is any nonsinusoidal periodic sound wave.

Psychoacoustic studies show that the pitch associated with a simple tone is simply determined by its frequency, i.e., there is a monotonic relationship between the perceived pitch and the frequency of the simple tone. Complex tones, however, require a more sophisticated analysis. Studies [15], [16], [17], indicate that the perception of the pitch of a complex sound may be formulated as a pattern matching (or classification) problem. In particular, Goldstein [17] has proposed a model which simply determines the pitch of a sound complex

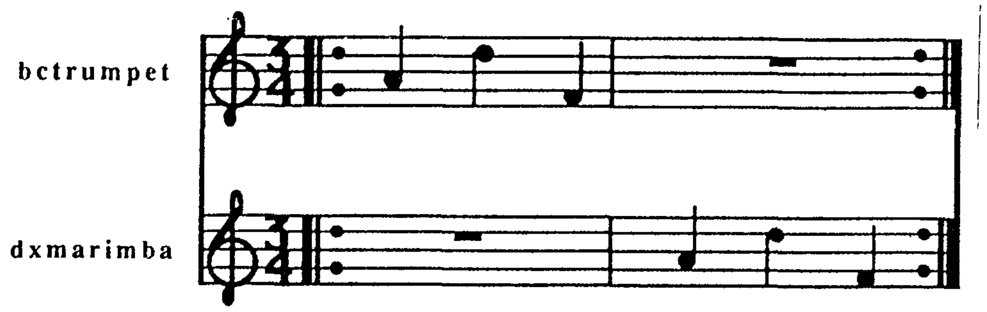


Figure 5.4: *Bi-timbred monophonic music presented for pitch classification*

by matching its lower order harmonics to an harmonic series in a maximum likelihood sense. In any case, the determination of the pitch of a complex sound is non trivial.

The acoustic data is obtained from a commercially available electronic synthesizer, a Yamaha DX7. As such the the perceived pitch of the generated data is known. For two different timbres, 'bctrumpet' and 'dxmarimba', a sequence of three notes is generated resulting in the 'song' displayed in figure 5.4.

Cochlea outputs corresponding to each timbre represented in figure 5.4 are depicted in the appendix B. It can be seen that these timbres exhibit a desirable stationarity property, i.e. the change in patterns with respect to time is small. Sequences of pitches are formed simply from the concatenation of small steady state segments of these cochlea outputs.

A 30:5 two layer feed forward neural network is employed as the pattern associator. As in the case of phoneme recognition, the pattern associator parameter vector is initialized with a random pseudo-orthogonal set of weights and biases (see chapter four). Experimental results (after 2400 epochs) for the 'song' represented in figure 5.4 is depicted in figure 5.5.

The figure shows that the system has not only determined that there are only three distinct pitches, but that they occur in the correct sequence. In other words the system is able to reconstruct the melody presented in figure 5.4 independent of the timbre producing it.

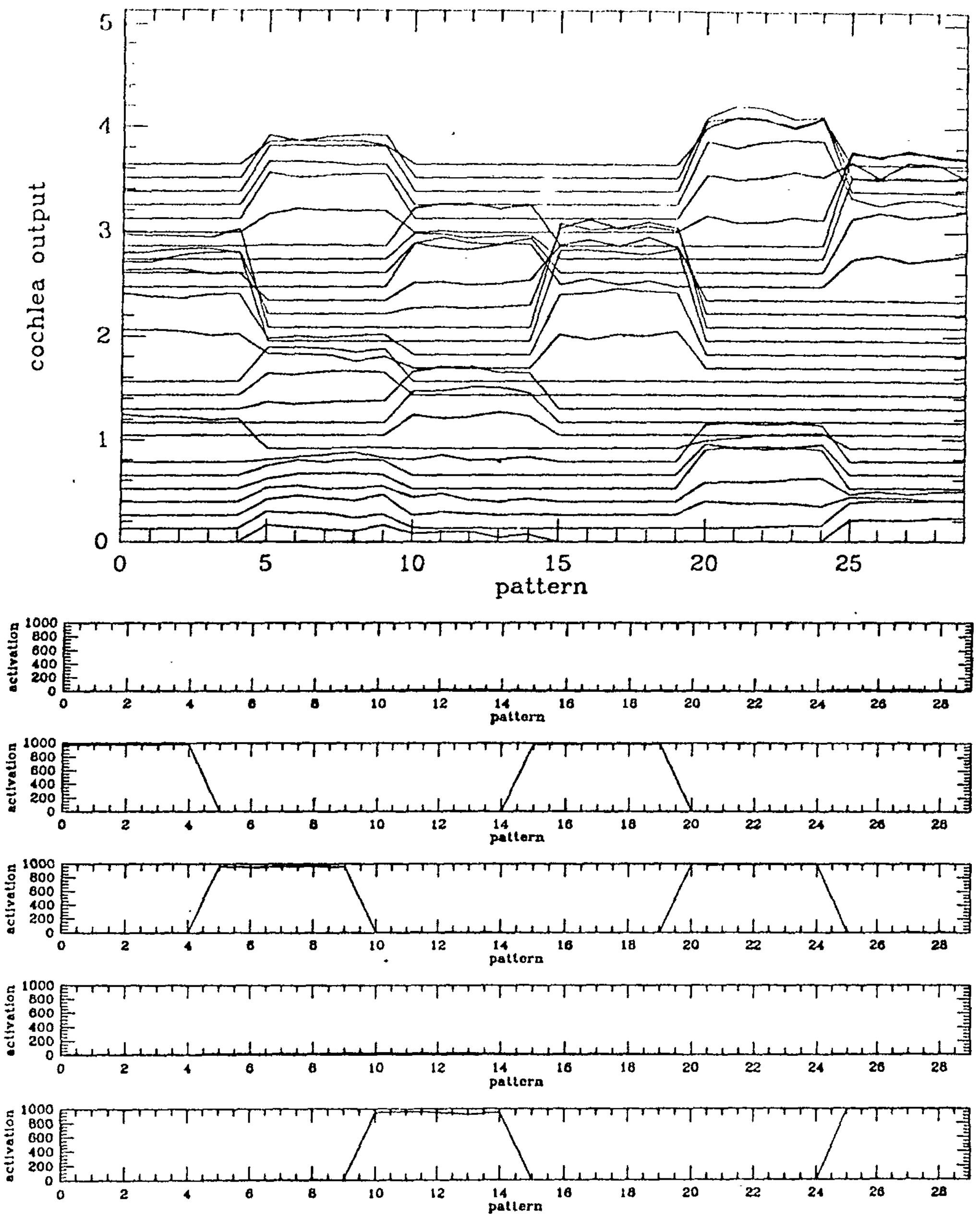


Figure 5.5: *Resulting segmentation for pitch classification*

CHAPTER
# SIX
## Conclusions and Future Research

Up until the recent past, the power of multi layer feed forward artificial neural networks has been untapped mainly due to the lack of algorithms to train them. With the emergence of the backpropagation algorithm; however, this deficiency has been removed. Despite this innovation, the backpropagation method is still not without its drawbacks. Among these the most prominent are the facts that i) the learning is conducted in a supervised manner and ii) that learning and operation must occur in two distinct phases.

Because of these properties, the backpropagation algorithm falls short of solving a 'true' pattern classification problem. This is not to say that a network could not be trained via backpropagation to mimic a previously solved pattern classification scheme; but that the backpropagation method is not capable of autonomously generating classification schemes.

A more realistic (and certainly more useful) learning scenario is that patterns would be presented without supervision to the system continuously; consequently, the system will begin to group like patterns into similar classes and continue to do so indefinitely; i.e. continuous learning. It is exactly this type of learning that has been developed in chapter four.

The following is a brief synopsis of what has been discussed here:

- Reviewed the discipline of neural networks; in particular the concept of a feed forward artificial neural network. Formulated the pattern classification problem and introduced the notion of a similarity measure.

- Proposed a neural system for a solution to the pattern classification problem which consists of three main components: transformation, pattern association, and update. Have shown that the resolution of the resulting classification is controllable via an external parameter.

- Based on an energy minimization scheme, developed a computationally efficient learning (update) rule for the system in the case of a magnitude independent similarity measure; leading to a two layer topology for the pattern associator. Among the most prominent benefits of the learning algorithm are computational efficiency, unsupervisedness, and continuous learning.

- Applied the system to several recognition tasks. Line separation, which demonstrated the resolving abilities of the system. With a model of the cochlea as the transformation stage, the system was assigned to the tasks of phoneme recognition as well as pitch perception. In each case the results indicated the potential success of such an algorithm.

Attention is now turned to the future. Listed below are only a small segment of the numerous avenues possible for further investigation. The list is not intended to be exhaustive. Possible areas of future research:

- Developing a rigorous method of choosing the parametrization transform so as to insure the optimality of the classification in some sense.
- Extending the learning algorithm to include the class of recurrent networks.
- Modeling musical dissonance as conflicts in classification.
- Forming a comparison of the optimum processor theory of pitch perception based on the maximum likelihood estimate scheme proposed by Goldstein [17].
- Investigating alternative similarity measures.

A rather interesting similarity measure with which to experiment may be the gaussian similarity measure

$$r(x,y) = e^{-\frac{\|x-y\|_2^2}{\sigma^2}}, \quad \sigma \in \mathbb{R}.$$

With such a similarity function its gradient with respect to the output pattern $y$ is readily computed as

$$\nabla_y r(x,y) = \frac{2}{\sigma^2} r(x,y)(y-x).$$

Using the same energy function as in 4.3.2 this leads to an expression for the energy gradient as

$$\nabla E_\perp = \frac{2}{\sigma^2} \sum_{k \in \mathcal{P}} r_{kp}(1 - 2r_{kp})(y_p - y_k),$$

where as before $r_{kp} \triangleq r(y_k, y_p)$. Now assuming maximum resolving power is desired, one may set the parameter $\alpha > 1$ in $g_\alpha(\cdot)$. This in turn implies that $\frac{\partial g_\alpha(x)}{\partial x} > 0, \forall x$. Since $r_{kp} \in (0,1)$ is positive, the gradient may be crudely estimated as

$$\begin{aligned} \nabla E_\perp &= \frac{2}{n_p \sigma^2} \left( y_p - \sum_{k \in \mathcal{P}} y_k \right) \\ &= \frac{2}{n_p \sigma^2} (y_p - p(y)) \end{aligned}$$

where the quantity $p(y)$ may be estimated just as in chapter four.

# Appendix A

# Some Derivations

## A.1 Computation of the Gradient

The following presents a careful derivation of the expression for the gradient of $r_{kp} \triangleq \bar{y}_k^T \bar{y}_p$. Let

$$r = a^T \bar{x} = a^T \frac{x}{\|x\|_2}, \tag{A.1.1}$$

where $a = \bar{y}_k$ and $x = y_p$. Further, let $x_i$ denote the $i$th component of $x$ and $\bar{x} \triangleq \frac{x}{\|x\|_2}$. With this notation one may write

$$r = \frac{1}{\|x\|_2} \sum_{k=1}^{n_o} a_k x_k \tag{A.1.2}$$

First, note

$$\frac{\partial\left(\|x\|_2^{-1}\right)}{\partial x_i} = -\|x\|_2^{-2} \bar{x}_i. \tag{A.1.3}$$

Now,

$$\begin{aligned} \frac{\partial r}{\partial x_i} &= -\|x\|_2^{-2} \bar{x}_i a^T x + \|x\|_2^{-1} a_i & \text{(A.1.4)} \\ &= \|x\|_2^{-1} \left(a_i - \bar{x}_i \bar{x}^T a\right), & \text{(A.1.5)} \end{aligned}$$

which gives

$$\nabla_x r = \frac{1}{\|x\|_2}\left(I - \bar{x}\bar{x}^T\right)a. \tag{A.1.6}$$

Then substituting for $x$ and $a$,

$$\nabla_{y_p} r_{kp} = \frac{1}{\|y_p\|_2}\left(I - \bar{y}_p\bar{y}_p^T\right)\bar{y}_k. \quad \square \tag{A.1.7}$$

## A.2 Concavity of $E_p$

Let $a \in \mathbb{R}^n$ and $r : \mathbb{R}^n \mapsto \mathbb{R}$ be given as

$$r(x) = a^T x, \tag{A.2.1}$$

and $g : \mathbb{R} \mapsto \mathbb{R}$ as

$$g(r) = r(1-r). \tag{A.2.2}$$

The expression for $E_p$ may then be written as

$$E_p = \sum_{k\in\mathcal{P}} g(r_k), \tag{A.2.3}$$

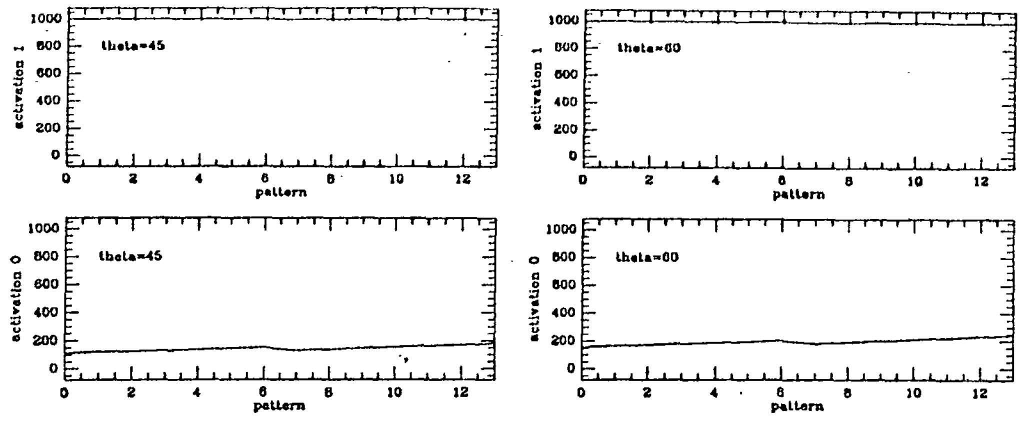
where $r_k = r(x_k),\ x_k \in \mathbb{R}^n$.

Clearly $g(r)$ is concave in $r$, and $r(x)$ is linear in $x$. Now, take $w, z \in \mathbb{R}^n$ and $\lambda \in [0,1]$ so that

$$\begin{aligned} g\left(r(\lambda w + (1-\lambda)z)\right) &= g(\lambda r(w) + (1-\lambda)r(z)) & \text{(A.2.4)} \\ &< \lambda g(r(w)) + (1-\lambda)g(r(z)) & \text{(A.2.5)} \end{aligned}$$

The first equality follows since $r(\cdot)$ is linear, while the last inequality follows from the concavity of $g(\cdot)$. But this inequality states that $g(r(\cdot))$ is concave. So $E_p$ and hence $E_\perp \triangleq \sum_{p\in\mathcal{P}} E_p$ is concave since the sum of concave functions is concave. $\square$

numerals ‘one’ and ‘six’.



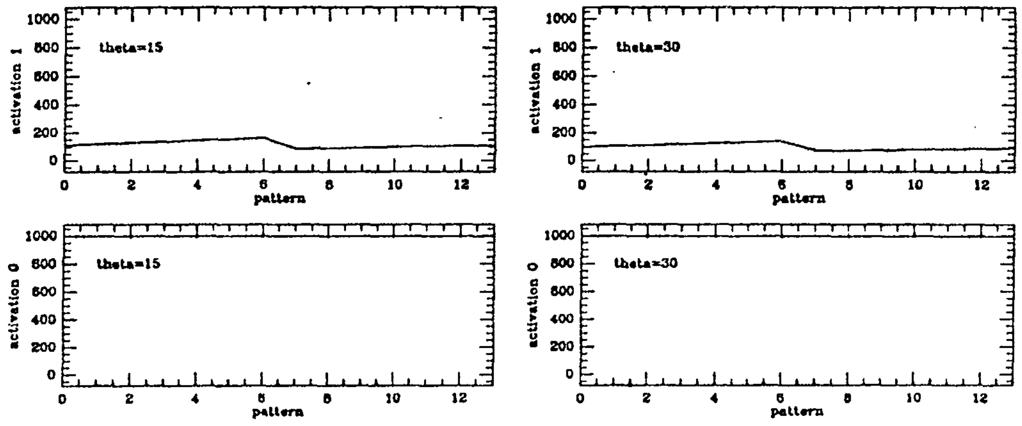


Figure B.1: *Resulting segmentations for* $\alpha$ = *0.5*

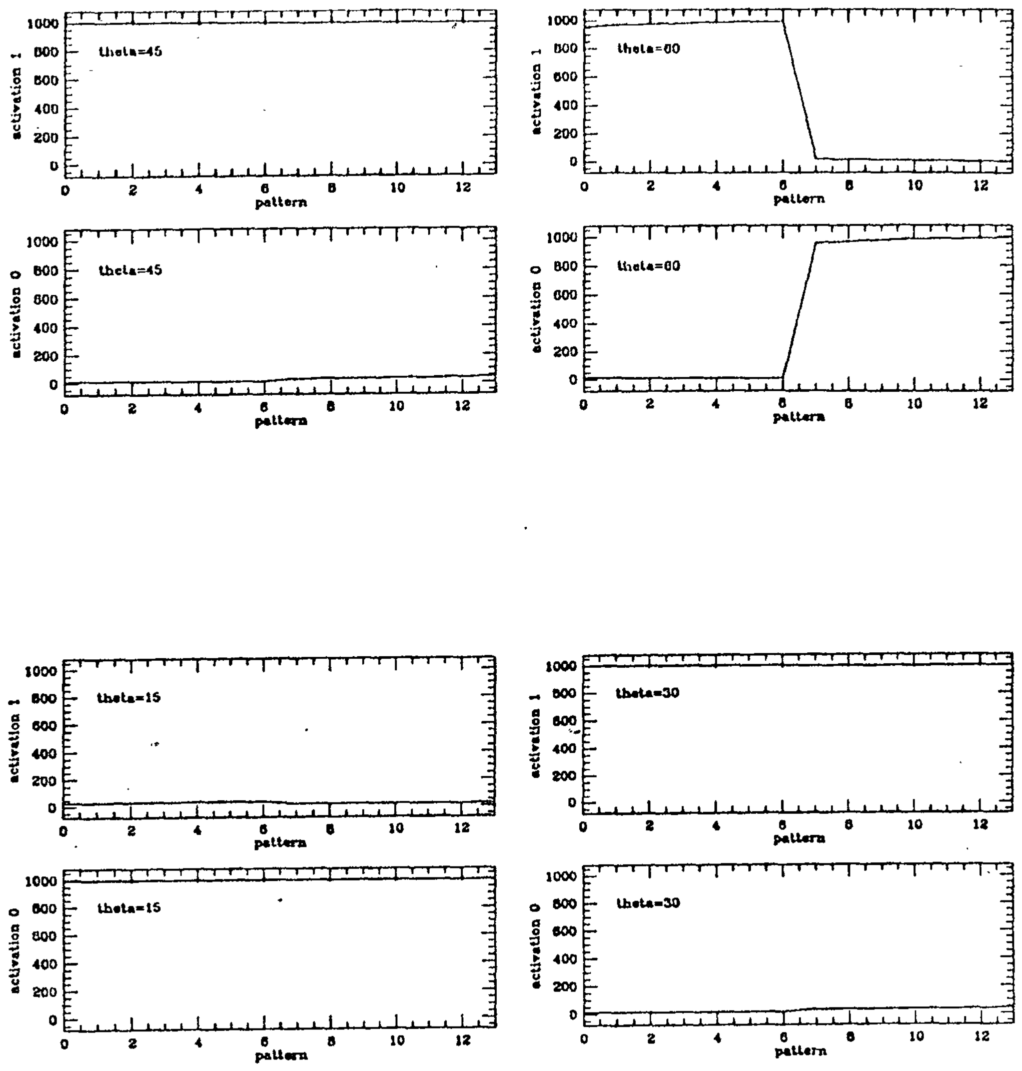


Figure B.2: *Resulting segmentations for $\alpha$ = 0.7*

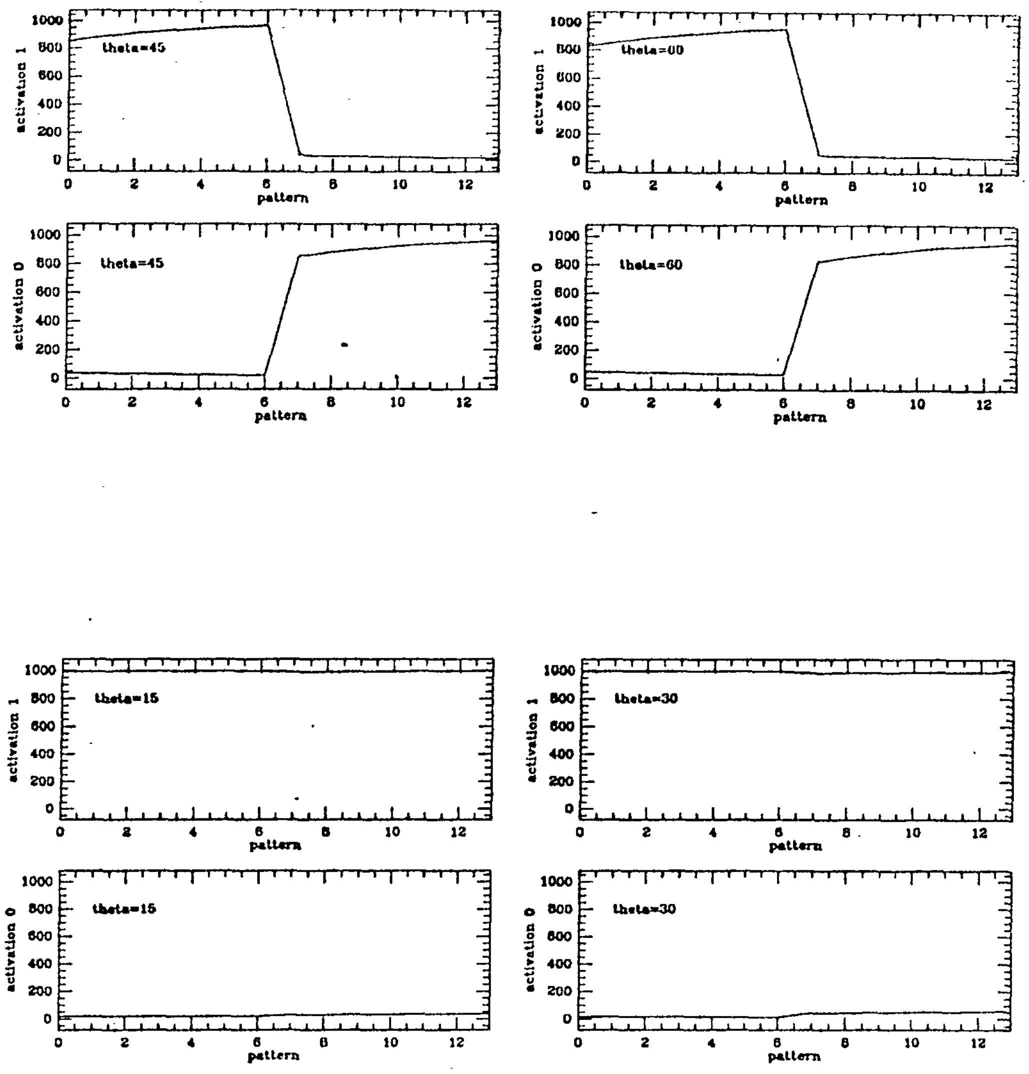


Figure B.3: *Resulting segmentations for $\alpha$ = 0.9*

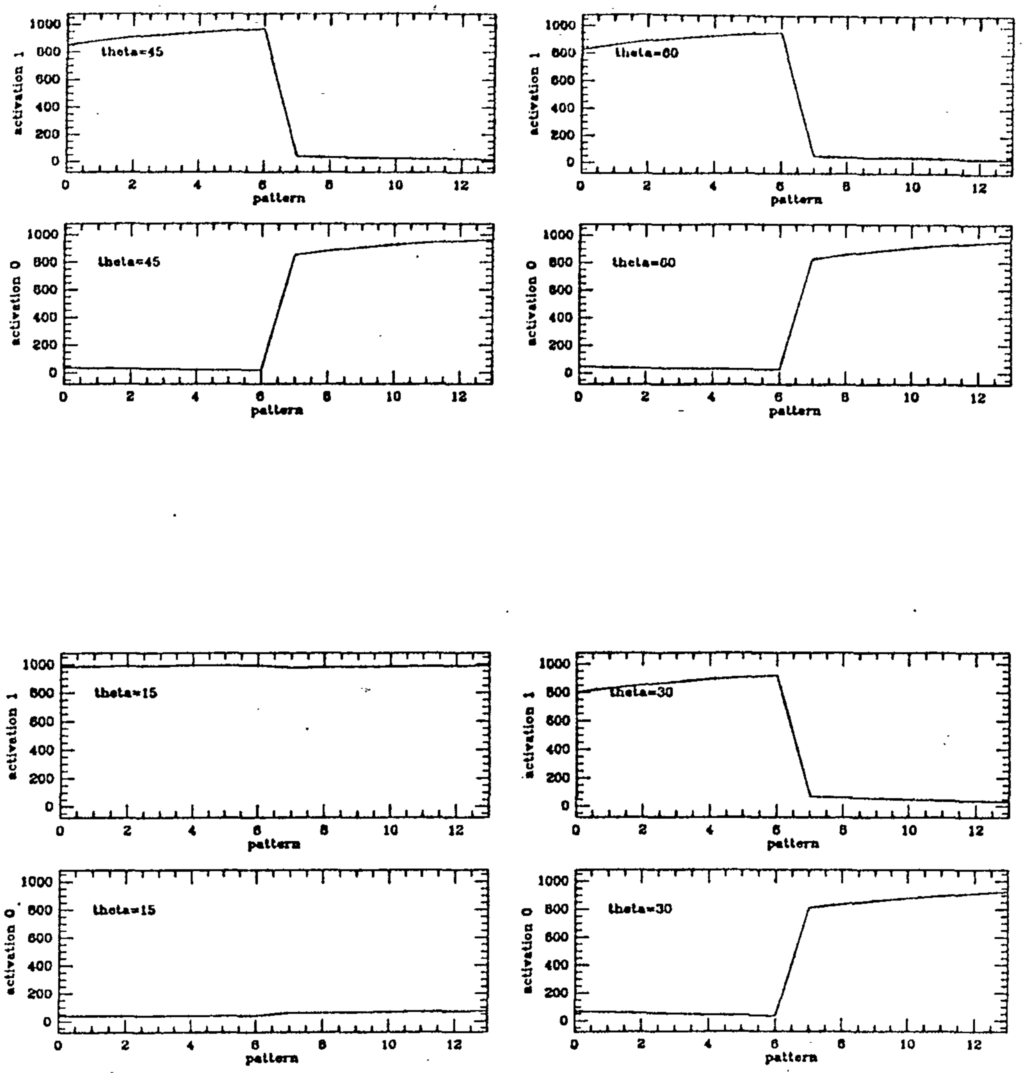


Figure B.4: *Resulting segmentations for $\alpha$ = 0.99*

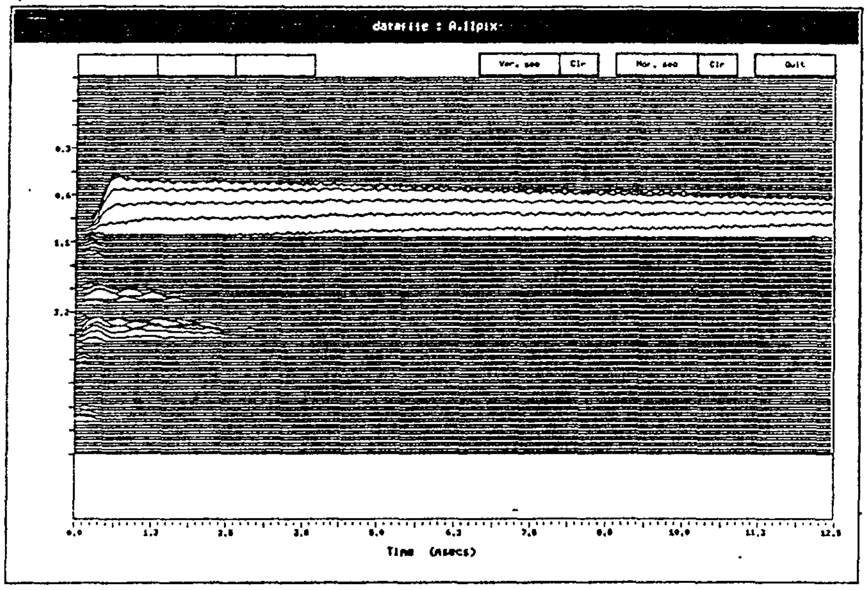


Figure B.5: *Cochlea response to a 'dxmarimba' A = 440Hz*

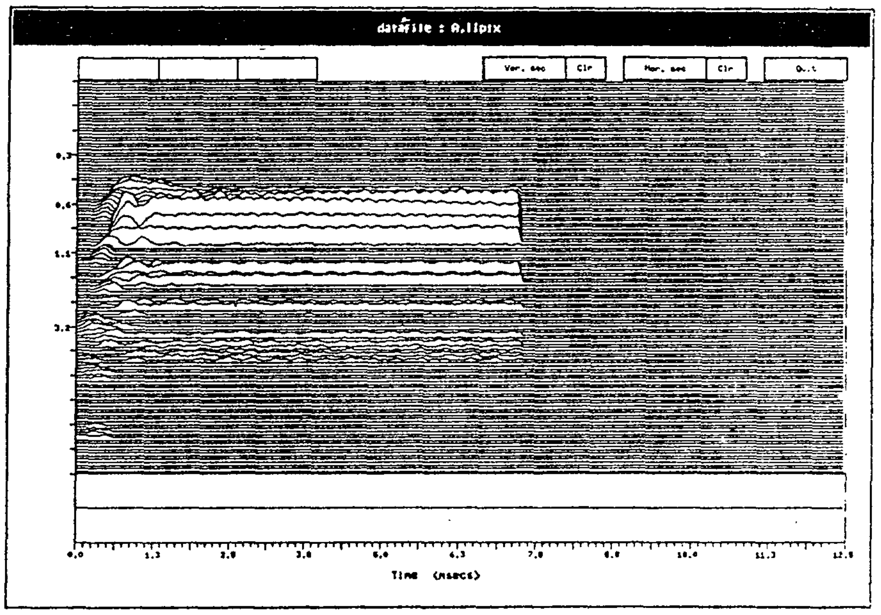
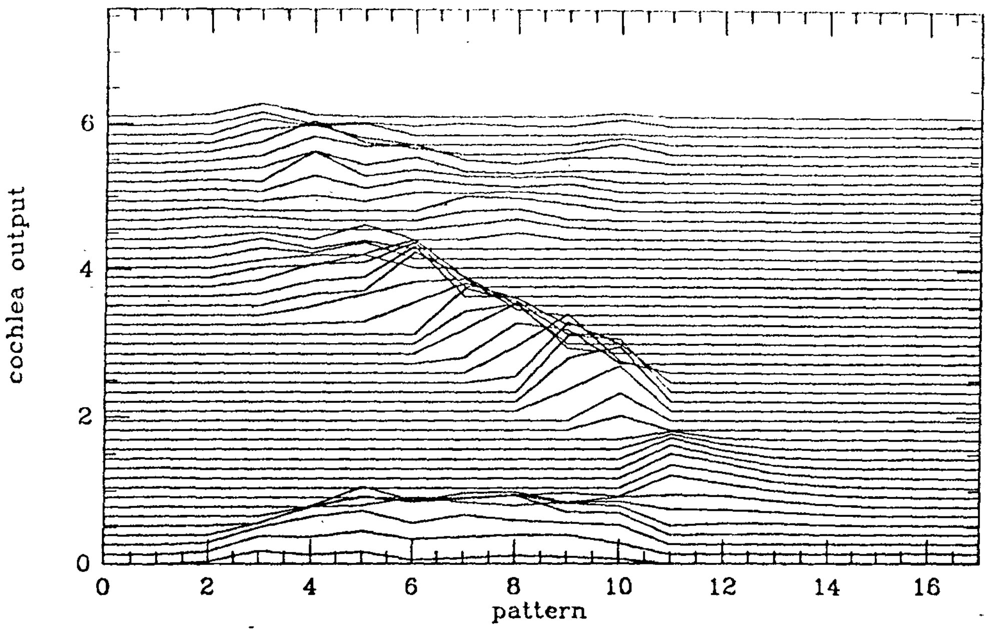


Figure B.6: *Cochlea response to a 'bctrumpet' A = 440Hz*



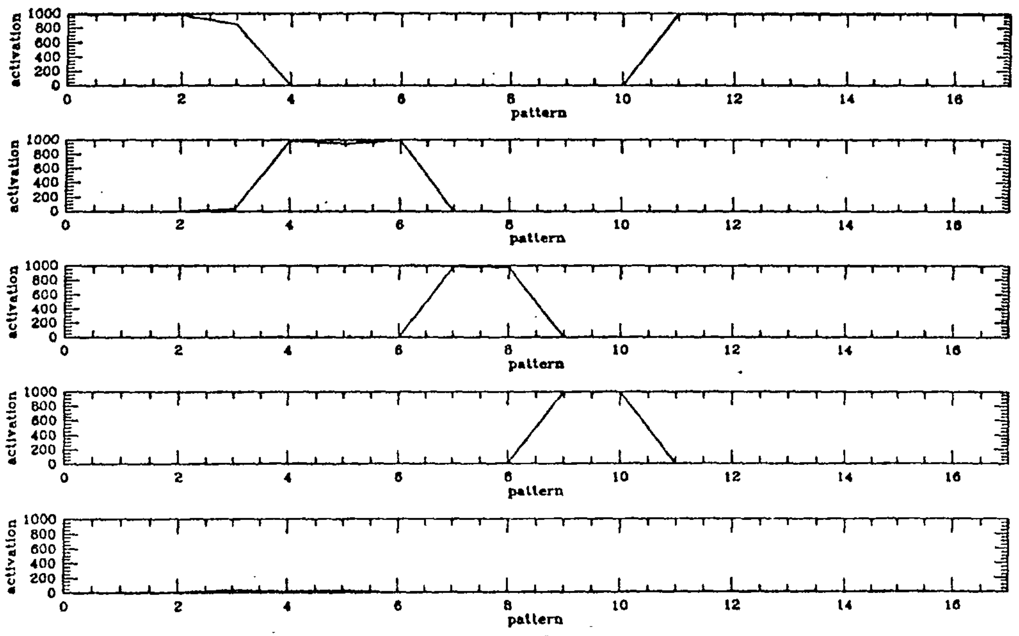


Figure B.7: *Training data and segmentation for word 'one'*

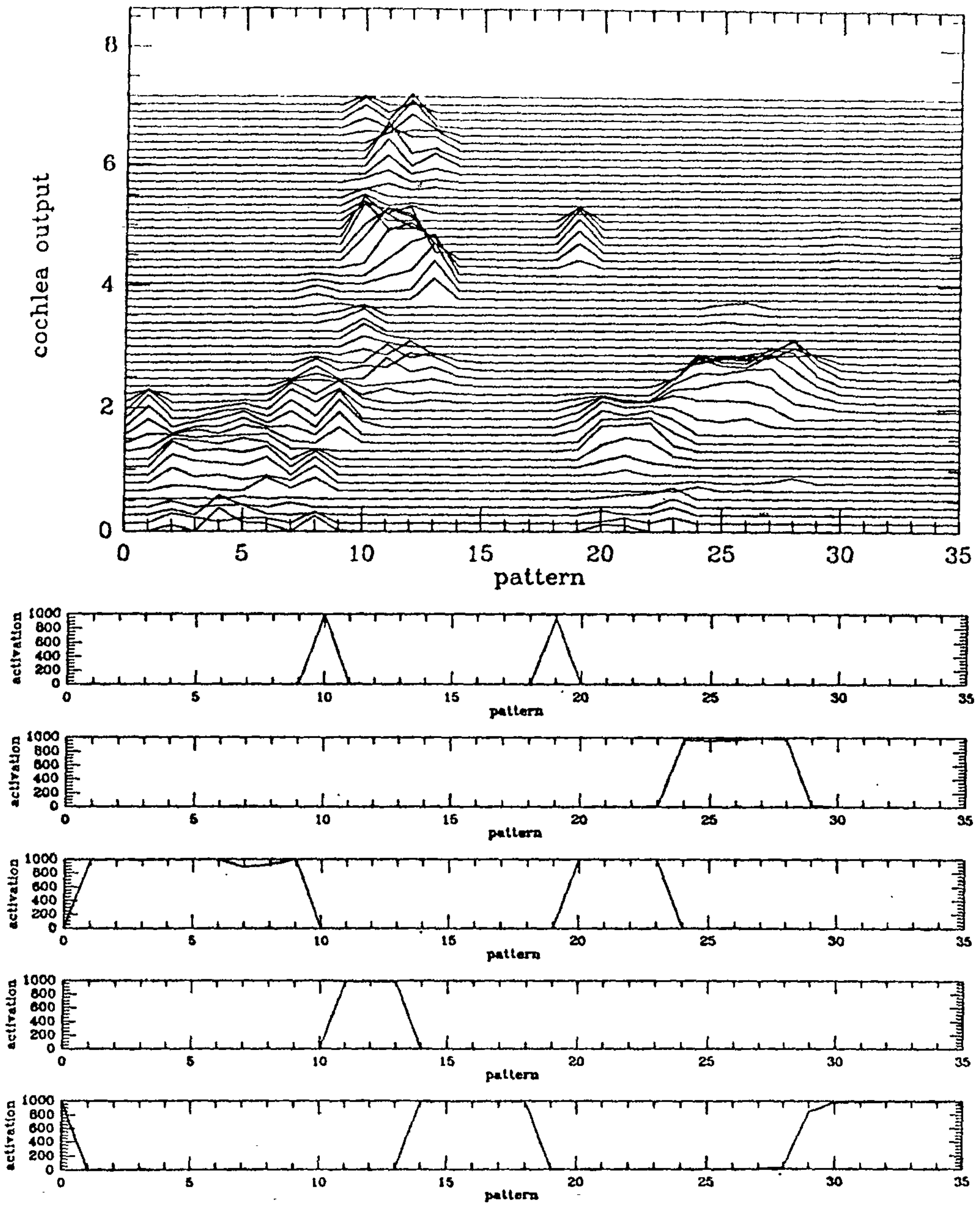


Figure B.8: *Training data and segmentation for word 'six'*

# BIBLIOGRAPHY

[1] C. R. K. Marrian, M. C. Peckerar, I. Mack, and Y. C. Pati, *Electronic Neural Net for Solving Ill-posed Problems with an Entropy Regularizer.* Cambridge, U.K.: Kluwer Academic, 1988. J. Skilling Editor.

[2] R. P. Lippmann, "An introduction to computing with neural nets," *IEEE ASSP Magazine.*, pp. 4–22, April 1987.

[3] D. Rumelhart and G. McClelland, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition.* Cambridge, Ma.: MIT Press, 1988.

[4] J. C. Platt and A. H. Barr, "Constrained differential optimization," American Institute of Physics, 1988.

[5] J. Hopfield, "Neurons with graded responses have collective computational properties like those of two-state neurons," *Proc. Natl. Acad. Sci., U.S.A.*, vol. 81, pp. 3088–3092, May 1984.

[6] D. Tank and J. Hopfield, "Neural computation of decisions in optimization problems," *Biological Cybernetics*, vol. 52, pp. 141–152, 1985.

[7] J. C. Platt and J. J. Hopfield, "Neural networks for computing," in *Proceedings American Insitute of Physics*, pp. 364–369, 1986.

[8] G. Cybenko, "Approximation by superpositions of a sigmoidal function," Technical Report, Tufts University, Department of Computer Science, October 1988.

[9] G. Cybenko, "Continuous valued neural networks with two hidden layers are sufficient," Technical Report, Tufts University, Department of Computer Science, March 1988.

[10] S. Shamma, "The acoustic features of speech sounds in a model of auditory processing: vowels and voiceless fricatives," *Journal of Phonetics*, vol. 16, pp. 77–91, 1988.

[11] S. Shamma, "Speech processing in the auditory system i: the representation of speech sounds in the responses of the auditory nerve," *Journal of The Acoustical Society of America*, vol. 78, pp. 1612–1621, November 1985.

[12] S. Shamma, "Speech processing in the auditory system ii: lateral inhibition and the central processing of speech evoked activity in the auditory nerve," *Journal of The Acoustical Society of America*, vol. 78, pp. 1622–1632, November 1985.

[13] C. G. Fant, “Speech sounds and features,” Tech. Rep., MIT, Cambrige, MA, 1973.

[14] A. J., “A perspective on man-machine communication by speech,” *Proceedings of the IEEE*, vol. 73, pp. 1541–1550, November 1985.

[15] E. Terhardt, “Pitch, consonance, and harmony,” *Acoustical Society of America*, vol. 55, pp. 1061–1069, May 1974.

[16] F. L. Wightman, “The pattern-transformation model of pitch,” *Acoustical Society of America*, vol. 54, pp. 407–416, May 1974.

[17] J. L. Goldstein, “An optimum processor theory for the central formation of the pitch of complex tones,” *Acoustical Society of America*, vol. 54, pp. 1496–1516, 1973.

[18] J. F. Shouten, “The residue revisted,” *Journal of The Acoustical Society of America*, vol. 54, pp. 41–58, 1973.

[19] R. J. Ritsma, “Frequencies dominant in the perception of pitch of complex sounds,” *Journal of The Acoustical Society of America*, vol. 42, pp. 191–198, 1967.

[20] R. Rosenblatt, *Principles of Neurodynamics.* New York: Spartan Books, 1959.

[21] D. G. Luenberger, *Linear and Nonlinear Programming.* Addison Wesley, 1984.

[22] G. W. Stewart, *Introduction to Matrix Computations.* Academic Press, 1973.

[23] J. Hopfield, "Neural networks and physical systems with emergent collective computational abilities," *Proc. Natl. Acad. Sci., U.S.A.*, vol. 79, pp. 2554–2558, April 1982.

[24] R. B. Stein, "Neural computation of decisions in optimization problems," *Kybernetik*, vol. 15, pp. 1–9, 1974.

[25] J. H. Li, A. N. Michel, and W. Porod, "Qualitative analysis and synthesis of a class of neural networks," *IEEE Transactions on Circuits and Systems*, vol. 35, pp. 976–986, August 1988.

[26] G. Carpenter and S. Grossberg, "A massively parallel architecture for a self-organizing neural pattern recognition machine," Technical Report, Boston University, Department of Mathematics, February 1986.