Adaptive Techniques in Signal Processing and Connectionist Models

By

M.R.Lynch

Christ's College

A Dissertation submitted to the University of Cambridge
for the degree of Doctor of Philosophy

Department of Engineering June 1990

To My Parents

Acknowledgements

I would like to thank my supervisor Dr. P.J. Rayner of Cambridge University Engineering Department for his guidance, encouragement and aid during the course of this research. I would like also to thank the Science and Engineering Research council and Dr. W. Fitzgerald and Marconi for their aid in financing the research.

Declaration

I certify that this dissertation contains an account of my own work, except were specific reference is made to others, and includes nothing which is the outcome of work done in collaboration. The work was carried out in Cambridge University Engineering Department between October 1986 and January 1990. This dissertation has not been submitted to any other university for any other degree. This dissertation comprises 171 pages in total.

M.R.Lynch.

Keywords:

Signal Processing, Adaptive, Digital Filters, Image Processing, Connectionist Models, Neural Networks, Non-linear Systems, Volterra Analysis, Conjugate Gradients, Least Mean Squares, Pattern Recognition, Image Registration, Non-linear filters.

Summary

This thesis covers the development of a series of new methods and the application of adaptive filter theory which are combined to produce a generalised adaptive filter system which may be used to perform such tasks as pattern recognition . Firstly, the relevant background adaptive filter theory is discussed in Chapter 1 and methods and results which are important to the rest of the thesis are derived or referenced. Chapter 2 of this thesis covers the development of a new adaptive algorithm which is designed to give faster convergence than the LMS algorithm but unlike the Recursive Least Squares family of algorithms it does not require storage of a matrix with n^2 elements, where n is the number of filter taps. In Chapter 3 a new extension of the LMS adaptive notch filter is derived and applied which gives an adaptive notch filter the ability to lock and track signals of varying pitch without sacrificing notch depth. This application of the LMS filter is of interest as it demonstrates a time varying filter solution to a stationary problem. The LMS filter is next extended to the multidimensional case which allows the application of LMS filters to image processing. The multidimensional filter is then applied to the problem of image registration and this new application of the LMS filter is shown to have significant advantages over current image registration methods. A consideration of the multidimensional LMS filter as a template matcher and pattern recogniser is given. In Chapter 5 a brief review of statistical pattern recognition is given, and in Chapter 6 a review of relevant connectionist models. In Chapter 7 the generalised adaptive filter is derived. This is an adaptive filter with the ability to model non-linear input-output relationships. The Volterra functional analysis of non-linear systems is given and this is combined with adaptive filter methods to give a generalised non-linear adaptive digital filter. This filter is then considered as a linear adaptive filter operating in a non-linearly extended vector space. This new filter is shown to have desirable properties as a pattern recognition system. The performance and properties of the new filter is compared with current connectionist models and results demonstrated in Chapter 8. In Chapter 9 further mathematical analysis of the networks leads to suggested methods to greatly reduce network complexity for a given problem by choosing suitable pattern classification

indices and allowing it to define its own internal structure. In Chapter 10 robustness of the network to imperfections in its implementation is considered. Chapter 11 finishes the thesis with some conclusions and suggestions for future work.

Contents

Symbol Definitions

Where possible the following conventions are used:

$\mathbf{w}$ Weight Vector.

$\mathbf{x}$ Input Vector.

$\mathbf{d}$ Desired Signal.

ϵ Error signal.

μ Adaption Coefficient.

$\mathbf{v}$ Expanded Space Vector.

E() Expectation operator.

$\mathbf{R}$ Correlation Matrix.

$\mathbf{p}$ Cross-correlation vector.

$\mathbf{\Lambda}$ Diagonal matrix.

λ Eigenvalue.

Abbreviations

The following are common abbreviations:

LMS Least Mean Squares.

RLS Recursive Least Squares.

ADF Adaptive Digital Filter.

FIR Finite Impulse Response.

SVD Singular Value Decomposition.

IIR Infinite Impulse Response.

OCR Optical Character Recognition.

RBF Radial Basis Function.

MLP Multilayer Perceptron.

HLBP Hidden Layer Back Propagation.

Introduction

Although the areas of adaptive signal processing and connectionist modeling owe their early origins to the same work, the two subjects have tended to become independent. This thesis uses some approaches and methods more recently developed in the field of signal processing and applies them to the field of connectionist models. Consequently the thesis is divided into two parts, the first of these covers work in the area of adaptive signal processing, and the second the application of signal processing methods to connectionist modeling to develop a new connectionist model and methods by which to analyse it and refine it.

Part I

Chapter 1

Current Adaptive Methods and Theory.

1.1 Introduction

Practical information processing systems must be able to function in continually changing environments; they must also be capable of taking into account the many complexities of the real world.

Adaptive signal processing gives a method for producing such systems. Adaptive systems may track changes in their environments. The adaption process may also allow a system to design itself taking account of many complex aspects of the task it is required to perform.

The basis of adaptive signal processing is the adaptive filter. In signal processing adaptive filters have until recently tended to be linear , that is, the filter output is a linear function of the data applied to its input.

The statistical approach to designing a linear filter is to assume various statistical measures are known, normally the mean and autocorrelation functions of the information and noise signal inputs.

The adaptive approach does not require this 'a priori ' statistical knowledge. The adaption algorithm forms internal estimates of this information whilst the filter is operating, and uses these estimates to adapt the filter to a suitable state.

One of the first applications of adaptive signal processing was that of adaptive beam forming. A beam former for the reception of RADAR or radio signals was required to be able to null out jamming signals coming from a localised direction. Bernard Widrow [8]

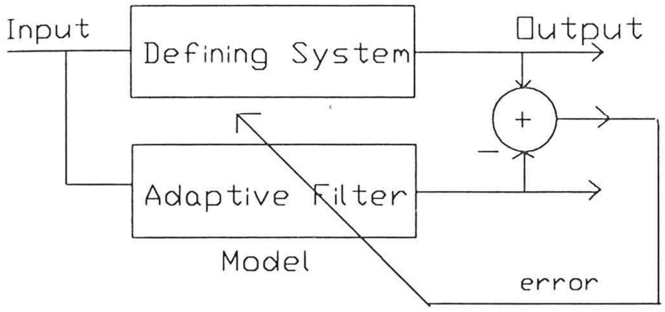


Figure 1.1: Adaptive Filter

developed such a system based on an error surface gradient following method, which was later to become known as the Least Mean Squares (LMS) algorithm.

The similarity of the mathematics applying to the beam forming problem and digital filtering led to the application of this new approach to digital filters. One of the classical applications of these adaptive digital filters is that of noise cancellation [1][8].

In general an adaptive filter is required to minimise some function of an error signal; this error is normally the difference of the output of the filter and some desired signal. A common function to minimise is the mean square of this error signal.

Such a filter may be configured to model another system as shown in figure 1.1. The importance of adaptive filters has become apparent in the field of signal processing due to their ability to take account of slow time varying systems such as communication channels in channel equalisation, or acoustic pathways in active noise cancelling.

Another aspect of adaptive systems, the ability to adapt themselves to solve problems, is making them very important in the field of artificial intelligence.

It has become apparent that it is very difficult to define the rules that will allow a machine to perform many 'real-world' tasks , such as visual pattern recognition under general conditions. The beauty of adaptive systems for such problems is that it is no longer necessary to derive a set of explicit rules as the system can be left to adapt itself 'on the job' to solve the problem without being explicitly programmed with sets of rules.

In considering such systems it is necessary to return to the basic Wiener Filter. In this thesis the discussion is limited to discrete time systems. The definition of such systems

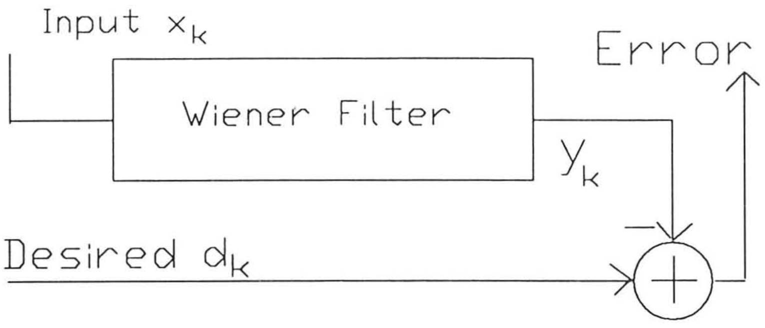


Figure 1.2: Wiener filter

may be found in [34].

1.1.1 The FIR Wiener Filter.

The Wiener filter is an optimal filter in the mean square sense. As shown in figure 1.2 an output error is defined as the difference between the desired reponse and the output of the filter. Classical Wiener theory uses the mean square error as the function to be minimised, and assumes that the information and noise processes are statistically stationary. Such a filter is said to be optimal in the least square sense.

In the Wiener filtering problem it is required to design the optimal filter for producing an output signal as similar (in the mean square sense) to the desired signal as possible.

Where d_k is the desired signal, x_{ik} the input signal for the i th delay point in an N tap FIR filter, y_k the output signal and w_i the filter coefficients. An error signal ϵ_k may be defined by the filter equation:

$$\epsilon_k = d_k - y_k = d_k - \sum_{i=1,N} w_i x_{ik}$$

The expectation of the error squared $E(\epsilon_k^2)$ is:

$$E(\epsilon_k^2) = E((d_k - \sum_{i=1,N} w_i x_{ik})^2)$$

For notational convenience we may define a vector composed of the filter weights:

$$\mathbf{w} = \begin{pmatrix} w_o \\ w_1 \\ w_2 \\ \vdots \\ w_n \end{pmatrix}$$

Likewise the input values in the filter:

$$\mathbf{x_k} = \begin{pmatrix} x_{ok} \\ x_{1k} \\ x_{2k} \\ \vdots \\ x_{nk} \end{pmatrix}$$

The filter equation becomes:

$$E(\epsilon^2) = E((d_k - \mathbf{w}^{\mathbf{t}}\mathbf{x_k})^2)$$

Define $\sigma^2 = E[d_k^2]$:

$$E(\epsilon^2) = \sigma^2 - 2d_k\mathbf{w}^{\mathbf{t}}E[\mathbf{x_k}] + \mathbf{w}^{\mathbf{t}}E[\mathbf{x_k}\mathbf{x_k^t}]\mathbf{w}$$

$$E(\epsilon^2) = \sigma^2 - 2\mathbf{w}^{\mathbf{t}}\mathbf{p} + \mathbf{w}^{\mathbf{t}}\mathbf{R}\mathbf{w} = J$$

Where $\mathbf{R}$ is the correlation matrix:

$$\mathbf{R} = \mathbf{E}\begin{pmatrix} x_{0k}^2 & \ldots x_{0k}x_{nk} \\ x_{1k}x_{0k} & \ldots x_{1k}x_{nk} \\ x_{2k}x_{0k} & \ldots x_{2k}x_{nk} \\ \vdots & \\ x_{nk}x_{0k} & \ldots x_{nk}^2 \end{pmatrix}$$

and $\mathbf{p}$ is the crosscorrelation $E[d_k\mathbf{x_k}]$, and J is the mean square error.

The minimum square error may be easily found by finding the derivative of the mean square error with respect to the weight vector and setting this to zero.

$$\nabla(E(\epsilon^2)) = -2\mathbf{p} + 2\mathbf{R}\mathbf{w}$$

$$\mathbf{R}\mathbf{w} = \mathbf{p}$$

$$\mathbf{w_{opt}} = \mathbf{R}^{-1}\mathbf{p}$$

This requires that the correlation matrix is non-singular so that its inverse exists.

It is necessary to know a priori the signal statistics of the input, reference, and desired signals. This may not be possible in many practical cases. In a non-stationary environment the Wiener filter theory is deficient as the optimal filter is time varying, that is, the coefficients of the filter are time dependent. Although Kalman filter theory can provide an approach to obtaining results for such problems an often more convenient approach is that of the adaptive filter.

The adaptive digital filter avoids the need to obtain these a priori estimates and consequently may be used where the signal statistics are unknown or changeable. The statistics must be 'pseudo-stationary' that is vary sufficiently slowly with time for the adaption algorithms to work.

In the case of Rank-deficient systems it is necessary to use a method which will give reliable results with singular matrices. A good method for use in such situations is that of singular value decomposition. This gives the minimum norm solution [1].

1.2 Properties of the Correlation Matrix and its Eigen-Values

It is important to state some basic properties of the correlation matrix which will be used later. Further properties and their proofs may be found in Haykin[1].

- The correlation matrix of a stationary discrete time stochastic process is Hermitian:

$$\mathbf{R}^H = \mathbf{R}$$

  Where H denotes conjugate transpose.

- The correlation matrix of a stationary discrete time stochastic process is Toeplitz.

- The correlation matrix of a stationary discrete time stochastic process is always non-negative definite.

$$\mathbf{x^H R x \geq 0}$$

Where $\mathbf{x}$ is an arbitrary non-zero M by 1 complex valued vector (The correlation matrix being M by M).

The eigenvector equation defines a vector $\mathbf{q}$ (M by 1):

$$\mathbf{R}\mathbf{q} = \lambda\mathbf{q}$$

1.3 The Adaptive Approach

Wiener theory is limited in that it assumes stationary random processes with statistics which are known a priori. In practical applications it is rare to encounter full stationarity or to have precise knowledge of statistics prior to the required operation. One approach might be to use the operating data to estimate the statistics and use these estimates to solve the Wiener filter. This would be computationally heavy and unlikely to be a real-time solution. The adaptive filter is a system that avoids these problems. It functions as an automaton with the capacity to tailor its operation to suit an unknown, and possibly time-varying, statistical environment. This is achieved by use of a performance feedback recursive algorithm which alters the filter parameters to optimise performance in some sense. In fact the standard linear transversal adaptive filter will converge to give the optimal Wiener solution for a statistically stationary problem. It should be borne in mind that the adaptive filter parameters being data dependent lead to the adaptive filter as a system being non-linear. However, this non-linear aspect is not usually referred to and the terms 'non-linear' or 'linear' are applied to adaptive filters in respect of their transfer functions at a point in time where their parameters are stationary in time.

There is a series of different algorithms to perform the parameter adjustments for FIR transversal filters, but the desired properties of an adaption algorithm are similar for many applications.

- Convergence Speed: This is the number of algorithm iterations required to reduce the error to within a given tolerance of the optimal solution. This should be as fast as possible, that is, require as few iterations as possible to get within a given tolerance of the optimal solution.

- Misadjustment: This is the difference between the performance of the filter after it has been allowed to adapt for an infinite time and the optimal filter.
- Robustness: This is the ability of the filter to operate in unsuitable signal and noise environments, for example ill-conditioned data for linear adaptive filters.
- Computational Load, Structure and Numerical Sensitivity: The amount and type of requisite arithmetical operations, the sensitivity of the algorithm to inaccurate calculation, and the required storage, may all be of importance in various applications. The structure of the algorithm may be of importance in consideration of algorithm implementation.

The following is a discussion of the more important of these adaption algorithms.

1.4 The LMS or stochastic gradient algorithm.

This is the pioneering algorithm of adaptive signal processing and is notable for its light computational load and ease of implementation.

From the above equations it can be seen that the error surface of the FIR filter is quadratic in form. Thus the error surface exhibits no local minima, as it contains only the global minimum. An optimisation algorithm may be applied to find the global minimum. The simplest of these algorithms is the stochastic gradient algorithm. This algorithm, which is also known as the method of steepest descent, involves use of the local gradient to determine the direction of the next iteration.

The mean square error equation can be differentiated with respect to the weight vector to give an expression for the gradient on the mean square error surface:

$$\nabla_w J = -2\mathbf{p} + 2\mathbf{R}\mathbf{w}$$

The steepest descent algorithm may thus be applied by always moving in the direction of steepest descent (changing the the time index k to the iteration index n):

$$\mathbf{w}_{n+1} = \mathbf{w}_n + \frac{1}{2}(\mu[-\nabla_{w(n)} J(n)])$$

where n is the iteration number and μ is the adaption coefficient.

As there is no a priori information about the correlation matrix $\mathbf{R}$ and the cross-correlation vector $\mathbf{p}$ it is not possible to use the above equations to give an exact value of the gradient. The LMS algorithm forms instantaneous estimates of $\mathbf{R}$ and $\mathbf{p}$ from the following:

$$\mathbf{R} = E(\mathbf{x}(\mathrm{n})\mathbf{x}^{\mathbf{t}}(\mathrm{n}))$$

is estimated by:

$$\mathbf{R_e} = \mathbf{x}(\mathrm{n})\mathbf{x}^{\mathbf{t}}(\mathrm{n})$$

similarly for $\mathbf{p}$. These estimates are unbiased (correct in their mean), but they may have large variances . However the LMS algorithm smooths these out. The gradient estimate is thus:

$$\begin{aligned}\nabla_{w(n)} J(n) &= -2\mathbf{x}(n)d(n) + 2\mathbf{x}(n)\mathbf{x}^t(n)\mathbf{w}(n) \\ &= -2\mathbf{x}(n)e(n)\end{aligned}$$

1.4.1 Problems of the LMS algorithm

The LMS algorithm suffers from some disadvantages such as slow convergence times and sensitivity to eigenvalue spread, and misadjustment.

Misadjustment

Consider the optimal solution (in the Wiener Sense) for the weight vector. This value of the weight vector will be denoted as $\mathbf{w_0}$. At any iteration in the algorithm a weight error vector may be defined:

$$\mathbf{e}_w(n) = \mathbf{w}(n) - \mathbf{w_0}$$

Haykin [1] gives an involved analysis of the LMS algorithm. This analysis is difficult as the random nature of $x(n)$ and $d(n)$ propagates into the weight vector estimate to give non-stationary weight vector estimates. Haykin[1] gives the result:

$$E(\mathbf{e}_w(n+1) = (\mathbf{I} - \mu\mathbf{R})E(\mathbf{e}_w(n))$$

It can be shown that the above equation converges to zero, provided μ is chosen in the range $0 < \mu < \frac{2}{\lambda_{max}}$. This is known as convergence in the mean. The propagation of the weight error vector variance is represented by $\mathbf{K}$ the weight error vector correlation matrix:

$$E(\mathbf{e}_w(n)\mathbf{e}_w^t(n)) = \mathbf{K(n)}$$

Haykin [1] gives the following recursion relationship in $\mathbf{K}$:

$$\mathbf{K(n+1)} = \mathbf{K(n)} - \mu[\mathbf{RK}(n) + \mathbf{K}(n)\mathbf{R}] + \mu^2\mathbf{R}tr[\mathbf{RK}(n)] + \mu^2 J_{min}\mathbf{R}$$

where $tr[]$ is the trace operator.
It can be seen that $K(n)$ can never reach zero as the last term represents a forcing term. That is the weight vector does not reach the optimal vector but fluctuates around it. This is due to the non-exact nature of the gradient estimate. This wandering leads to an excess error J_{ex} above the J_{min} value. The magnitude of this misadjustment may be found (Haykin [1]) who shows the excess error at $n = \infty$:

$$E(J_{ex}) = \frac{\mu J_{min} \sum_{k=1}^{N} \lambda_k}{2 - \mu \sum_{k=1}^{N} \lambda_k}$$

In which the λ_k are the eigenvalues of the correlation matrix $\mathbf{R}$
The misadjustment is defined as:

$$\frac{J_{ex}}{J_{min}}$$

Thus it can be seen that in cases when the eignevalues are widely spread, the misadjustment is mainly limited by the larger eigenvalues.

Effect of Eigenvalue Spread

Considering the previously defined weight error vector equation:

$$E(\mathbf{e}_w(n+1) = (\mathbf{I} - \mu\mathbf{R})E(\mathbf{e}_w(n))$$

This equation shows that the time evolution modes of the error weight vector are coupled. By using the unitary similarity transform the convergence modes may be uncoupled:

$$\mathbf{R} = \mathbf{Q}\mathbf{S}\mathbf{Q}^t$$

The weight error vector may be redefined in terms of the principal axes:

$$\mathbf{v}(n) \quad = \mathbf{Q}^t\mathbf{e}_w(n)$$

Yielding:

$$E(\mathbf{v}(n+1) = (\mathbf{I} - \mu\mathbf{S})E(\mathbf{v}(n))$$

This shows that each of the convergence modes is separated into a homogeneous first order difference equation with solution:

$$E(v_k(n)) = (1 - \mu\lambda_k)^n E(v_k(0))$$

It is clear that unless all the λ_k are equal the convergence rates for each mode are different and an optimal μ cannot be found for all the modes.

1.4.2 Choice of the adaption coefficient

It can be found experimentally that the performance of the LMS algorithm can be greatly affected by the choice of the adaption coefficient. Yassa [3] developed a useful approach to the question of the choice of adaption coefficient.

Considering the expression for the square error of an LMS filter:

$$\epsilon^2 = \sigma^2 - 2\mathbf{w}^t\mathbf{p} + \mathbf{w}^t\mathbf{R}\mathbf{w}$$

In the case of a complex filter this becomes for the m+1 iteration:

$$\epsilon^2_{m+1} = \sigma^2 - (\mathbf{w}^t_{m+1}\mathbf{p} + \mathbf{w}^{*t}_{m+1}\mathbf{p}^*) + \mathbf{w}^t_{m+1}\mathbf{R}\mathbf{w}^*_{m+1}$$

Also the complex gradient vector is given by:

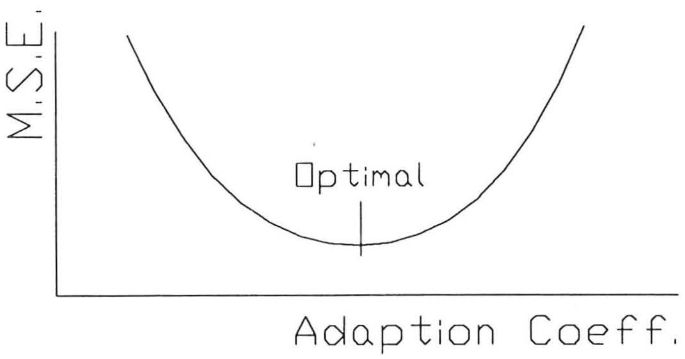


Figure 1.3: Parabolic relationship of adaption coefficient and mean square error.

$$(\nabla\epsilon^2)_m = 2[\mathbf{R}^t\mathbf{w}_m - \mathbf{p}^*_m]$$
$$= 2[\mathbf{R}^*\mathbf{w}_m - \mathbf{p}^*]$$

Using this it can be seen:

$$\epsilon^2_{m+1} = \epsilon^2_m - \mu_m(\nabla\epsilon^2)^t_m(\nabla\epsilon^2)^*_m + \mu^2_m(\nabla\epsilon^2)^t_m\mathbf{R}(\nabla\epsilon^2)_m$$
$$=, \epsilon^2_m - \mu \parallel (\nabla\epsilon^2)_m \parallel^2 + \mu^2_m q_m$$

where :

$$q_m = (\nabla\epsilon^2)^t_m\mathbf{R}(\nabla\epsilon^2)_m$$

Given a nonzero $(\nabla\epsilon^2)_m$ it can be shown:$q_m > 0$. Hence the relationship between μ and the mean square error is parabolic.

The optimal μ may thus easily be found as:

$$\mu^{OPT}_m = \frac{1}{(2\alpha_m)}$$

where:

$$\alpha = \frac{q_m}{\parallel (\nabla\epsilon^2)_m \parallel^2}$$

In the case of the LMS algorithm :

$$\nabla\epsilon^2 = -2\epsilon\mathbf{x}^*$$

$$\mathbf{R} = \mathbf{x}\mathbf{x}^*$$

These reduce the expression for α to:

$$\alpha = (\mathbf{x}^{*t}\mathbf{x})$$

$$= \| \mathbf{x} \|^2$$

Thus a measure of the input power to the filter when suitably low pass filtered may be used to set and keep μ near its optimal value.

1.5 Recursive Least Squares

The performance of the LMS algorithm is not as good in terms of number of iterations to convergence as may be expected for an adaptive system from a theoretical analysis of adaptive methods [2][1]. Indeed there are methods which offer convergence in fewer iterations than the LMS algorithm. Perhaps the most common of these, and certainly the basis of a host of related algorithms, is the recursive least squares algorithm [1], which itself arises from the classical method of least squares.

Defining: Input data vector $\mathbf{u}$, a forgetting factor λ which if close to one gives the algorithm infinite memory and if smaller allows the algorithm to track time varying problems, and a matrix $\mathbf{P}$ which is of order m by m where m is the number of elements of $\mathbf{u}$.

Initialise algorithm by setting:

$$\mathbf{P} = \delta^{-1}\mathbf{I}$$

$$\mathbf{w}_0 = \mathbf{0}$$

For each time instant $n = 1, 2, \ldots$ compute:

$$\mathbf{k}(n) = \frac{\lambda^{-1}\mathbf{P}(n-1)\mathbf{u}(n)}{1 + \lambda^{-1}\mathbf{u}^H(n)\mathbf{P}(n-1)\mathbf{u}(n)}$$

$$\alpha(n) = d(n) - \mathbf{w}^H(n-1)\mathbf{u}(n)$$

$$\mathbf{w}(n) = \mathbf{w}(n-1) + \mathbf{k}(n)\alpha^*(n)$$

$$\mathbf{P}(n) = \lambda^{-1}\mathbf{P}(n-1) - \lambda^{-1}\mathbf{k}(n)\mathbf{u}^H(n)\mathbf{P}(n-1)$$

It is immediately clear that the algorithm requires the storage of a matrix and although this is not a problem for many adaptive filter problems it is a prohibitive storage requirement for systems with many weights such as image filters or connectionist models.

1.6 IIR Adaptive Filters

There has been a great deal of interest in adaptive IIR filters. It has however proved very difficult to develop a practical algorithm. The problems are mainly due to the multimodal nature of the error surface and the difficulty of constraining the filter to be stable for all iterations of the algorithm. As adaptive IIR filter theory is not used elsewhere in the thesis the subject is not pursued here, however it can serve to illustrate some of the problems encountered by adaption methods with multimodal error surfaces.[29][32]

1.7 Singular Value Decomposition

This is not an adaptive method but is very important for the study and analysis of adaptive systems in which the correlation matrix is rank deficient.
The singular value decomposition theorem states:
For a matrix **A** of rank w there are two unitary matrices **XY** such that:

$$\mathbf{Y}^H\mathbf{A}\mathbf{X} = \begin{pmatrix} \Sigma & 0 \\ 0 & 0 \end{pmatrix}$$

where:

$$\Sigma = diag(\sigma_1, \sigma_2, \ldots, \sigma_w)$$

and:

$$\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_w > 0$$

Normally we could solve the least squares problem by inverting the autocorrelation matrix to give the optimal vector.

$$\mathbf{w} = \mathbf{R}^{-1}\mathbf{p}$$

This theorem (SVD) can be used to derive a solution for **w** even when the matrix **R** is singular. By solving:

$$\mathbf{p} = \mathbf{R}\mathbf{w}$$

for **w**. More generally this equation may be rewritten:

$$\mathbf{w} = \mathbf{A}^{\mathbf{m}}\mathbf{b}$$

Where $\mathbf{A^m}$ is the L by M Moore-Penrose generalised inverse or pseudoinverse of matrix $\mathbf{A}$. Singular Value Decomposition can give a unique solution to the equation, giving the one solution with the minimum-norm. The pseudoinverse of the matrix $\mathbf{A}$ may be defined as:

$$\mathbf{A}^m = \mathbf{X}\begin{pmatrix} \Sigma^{-1} & 0 \\ 0 & 0 \end{pmatrix}\mathbf{Y^H}$$

where :

$$\Sigma^{-1} = diag(\sigma_1^{-1}, \sigma_2^{-1}, \ldots, \sigma_w^{-1})$$

And w is the rank of the matrix $\mathbf{A}$. This is used as follows:

$$\begin{aligned} \mathbf{w} &= \mathbf{A^m b} \\ &= \mathbf{X}\begin{pmatrix} \Sigma^{-1} & 0 \\ 0 & 0 \end{pmatrix}\mathbf{Y^H b} \end{aligned}$$

The unitary matrices may be partitioned:

$$\begin{aligned} &= [\mathbf{X_1}, \mathbf{X_2}]\begin{pmatrix} \Sigma^{-1} & 0 \\ 0 & 0 \end{pmatrix}\mathbf{Y^H b} \\ &= \mathbf{X_1}\Sigma^{-1}\mathbf{Y_1^H b} \end{aligned}$$

Assuming $(L > M)$:

$$\begin{aligned} &= \mathbf{X_1}\Sigma^{-2}\mathbf{X_1^H A^H b} \\ &= \sum_{i=1}^{W} \frac{\mathbf{x_i}}{\sigma_i^2}\mathbf{x_i^H A^H b} \end{aligned}$$

This equation may be used by first computing the singular values of the data matrix $\mathbf{A}$ and the associated singular vectors $\mathbf{x_1} \ldots \mathbf{x_w}$ and substituting them into the above equation to give $\mathbf{w}$.

This gives a numerically well behaved method which may even be used for rank deficient systems. More recently, interesting work has been done on recursive implementations of SVD [43], although the computational load of such methods is still high.

Chapter 2

Conjugate Gradient Method

2.1 Introduction

Over recent years a series of algorithms for the updating of adaptive filter coefficients have been developed. Some of these methods, notably those related to the method of Recursive Least Squares (RLS) [1] have provided significant increases in performance over the Least Mean Squares (LMS) [1] algorithm. For many applications the problems posed by the lack of performance of the LMS algorithm can be overcome by implementing one of these other algorithms.

However, almost all of this new generation of algorithms require the storage of a matrix and a significant increase in the computational load. This is not a problem for most adaptive filters which are one-dimensional and utilise a limited number of coefficients. In the case of multidimensional adaptive filters, such as those used in image processing [7], or connectionist models or for any adaptive filter with many coefficients however, it may be totally impractical to consider storage of a matrix containing $O(n^2)$ coefficients where n is the number of coefficients. Likewise the increase in computational load associated with current approaches may be too great.

In this chapter new algorithm is presented which gives a substantial increase in performance over the LMS algorithm but does not require the storage of a matrix and has a computational burden of approximately five times that of LMS.

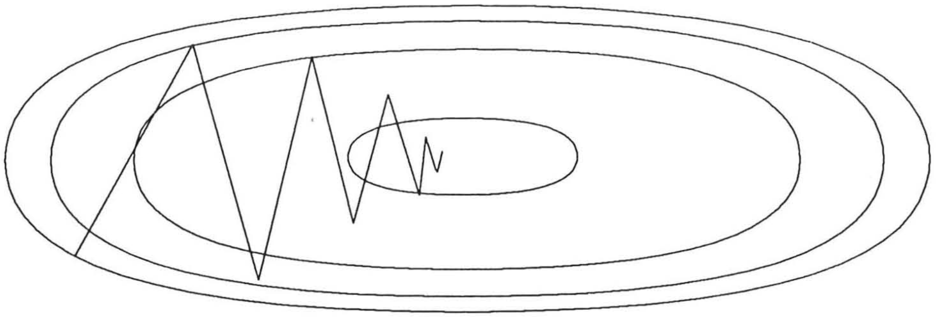


Figure 2.1: LMS Convergence Path

2.1.1 The LMS Algorithm

In order to construct a new algorithm with increased performance over the LMS algorithm it is necessary first to consider the failing of the LMS algorithm. The update equation for the LMS or stochastic gradient method is [1]:

$$\mathbf{w_{k+1}} = \mathbf{w_k} + 2\mu\epsilon\mathbf{x}$$

or more generally:

$$\mathbf{w_{k+1}} = \mathbf{w_k} - \alpha\nabla(\epsilon^{\mathbf{2}})$$

Where $\mathbf{w}$ is the filter weight vector and k is the time index, α and μ are adaption coefficients, ϵ the filter error and $\mathbf{x}$ the filter input data vector. Consider an error surface which is a long narrow valley. The steepest descent type algorithms, such as LMS, will in general tend to zig-zag down the valley as shown (Fig.2.1). This behaviour even occurs for perfectly parabolic surfaces and so leads to large inefficiencies in terms of convergence time.

2.2 Derivation of the Algorithm

Consider the minimisation of a function f over an N dimensional coordinate system. The function can be approximated by the first three terms of a Taylor series of the function about a point P.

$$f(\mathbf{w}) \approx f(\mathbf{P}) + \sum_i \frac{\partial f}{\partial w_i} w_i + \frac{1}{2}\sum_{ij} \frac{\partial^2 f}{\partial w_i \partial w_j} w_i w_j$$

$$= c - \mathbf{bw} + \frac{1}{2}\mathbf{wAw}$$

Where:

$$c = f(\mathbf{P})$$

$$\mathbf{b} = -\nabla f$$

$$[A]_{ij} = \frac{\partial^2 f}{\partial w_i \partial w_j}$$

It should be noted that the above is exact for a quadratic function. Matrix A is the second partial derivative matrix of the function f, the Hessian of f, at $\mathbf{P}$. If a minimisation in the direction of vector $\mathbf{u}$ is performed it is necessary to find a direction in which to minimise so as not to destroy the previous minimisation. Consequently the next direction ∂w must be such that the gradient is perpendicular to $\mathbf{u}$, that is $\mathbf{v}$:

$$\mathbf{u}.\partial(\nabla \mathbf{f}) = \mathbf{u}.\mathbf{A}.\mathbf{v} = \mathbf{0}$$

By calculating the gradient of the above approximation with respect to the weights we obtain:

$$\nabla \mathbf{f} = \mathbf{A}.\mathbf{w} - \mathbf{b}$$

And hence, the effect on the gradient of a small change in $\mathbf{w}$ is:

$$\delta \nabla \mathbf{f} = \mathbf{A}.\delta \mathbf{w}$$

$\mathbf{u}$ and $\mathbf{v}$ are referred to as mutually conjugate. It is also necessary that our search directions must be mutually orthogonal so that the whole space may be spanned in the search.

Following the derivation of the Fletcher-Reeves [4] algorithm it is possible to invoke a theorem which will allow the construction of mutually orthogonal and conjugate vector sequences.

2.2.1 Theorem 1

If $\mathbf{A}$ is a symmetric, positive definite N by N matrix and $\mathbf{g_0}$ is an arbitrary vector. For values of $i = 0, 1, 2......$ two series of vectors may be defined:

$$\mathbf{g_{i+1}} = \mathbf{g_i} - \lambda_\mathbf{i} \mathbf{A} \mathbf{h_i}$$

$$\mathbf{h_{i+1}} = \mathbf{g_{i+1}} + \gamma_i \mathbf{h_i}$$

Where:

$$\lambda_i = \frac{\mathbf{g_i^t}.\mathbf{g_i}}{\mathbf{g_i}.\mathbf{A}.\mathbf{h_i}}$$

$$\gamma_i = \frac{\mathbf{g_{i+1}}.\mathbf{A}.\mathbf{h_i}}{\mathbf{h_i^t}.\mathbf{A}.\mathbf{h_i}}$$

Then for all i such that $i \neq j$:

$$\mathbf{g_i^t}.\mathbf{g_j} = \mathbf{0}$$

$$\mathbf{h_i^t}.\mathbf{A}.\mathbf{h_j} = \mathbf{0}$$

That is, the $\mathbf{g_i}$ are mutually orthogonal and the $\mathbf{h_i}$ are mutually conjugate. The proof of this theorem is beyond the scope of this chapter but may be found in Polak's book [2]. It is a form of Gram-Schmidt orthogonalisation. However, this theorem still requires storage of a matrix $\mathbf{A}$. Another theorem enables this problem to be overcome.

2.2.2 Theorem 2

If $\mathbf{g_i} = -\nabla \mathbf{f}(\mathbf{P_i})$ and a minimisation of f in the direction $\mathbf{h_i}$ of f is performed to find a new point $\mathbf{P_{i+1}}$ and $\mathbf{g_{i+1}} = -\nabla \mathbf{f}(\mathbf{P_{i+1}})$ Then the same sequence is generated as in theorem 1. This may be proved by induction [2].

Consequently a series of mutually orthogonal vectors and a series of mutually conjugate vectors may be generated without knowledge or storage of $\mathbf{A}$.

2.2.3 Application to Adaptive Filters

In order to use the above theorems it is still necessary to perform line minimisations and evaluate the gradient. The gradient evaluation may be easily performed for the adaptive filter case. For the standard adaptive filter [1] as shown below (Omitting the time index for convenience):

$$\epsilon = d \ - \mathbf{w^t x}$$

Taking the mean square error:

$$E(\epsilon^2) = E(d^2) - 2\mathbf{wv} + \mathbf{wRw}$$

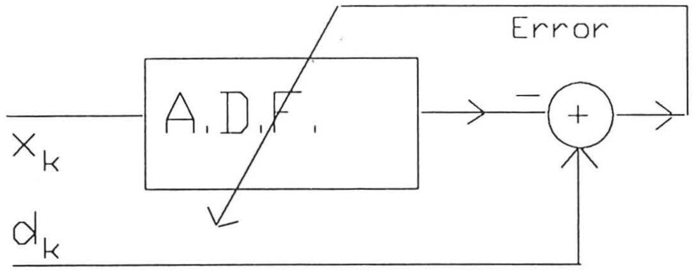


Figure 2.2: Standard Adaptive Filter

Where **x** is the vector representing the filter inputs, d is the desired response,**w** the weight vector,**R** is the autocorrelation matrix of the input and **v** the cross-correlation vector of the input and the desired signal. The gradient may thus be calculated as:

$$\nabla(\epsilon^2) = \frac{\partial \epsilon^2}{\partial \mathbf{w}} = 2\epsilon \frac{\partial \epsilon}{\partial \mathbf{w}} = -2\epsilon \mathbf{x}$$

The other problem is the line minimisation. The mean square error is a parabola as a function of distance along the line. Consider the line generated by:

$$\mathbf{w} = \mathbf{w} + \beta \mathbf{h}$$

Where β is the line generating scalar. The filter error can be written:

$$\epsilon = d - \sum_i x_i(w_i + \beta h_i)$$

The second derivative of the mean square error with respect to distance along the line **h** may be calculated as:

$$\frac{\partial^2 \epsilon^2}{\partial \beta^2} = \frac{\partial}{\partial \beta}\frac{\partial \epsilon^2}{\partial \beta} = \frac{\partial}{\partial \beta}(2\epsilon \frac{\partial \epsilon}{\partial \beta}) = 2(\sum_i x_i.h_i)^2$$

Given expressions for the derivative, second derivative and the knowledge that the relationship is parabolic it is possible to find that:

$$\beta = \frac{\epsilon}{\mathbf{x}^{\mathbf{t}}.\mathbf{h}}$$

at the line minimum along **h**. This gives only an estimate of the line minimum as the instantaneous error values could be affected by noise and incorporate no time averaging aspect. The approximation is found in practice to be sufficient for many applications. However this approximation can lead to a set of mutually conjugate and mutually orthogonal vector series which use up all viable directions before arriving at the minimum. This problem can be cured by periodic resetting of the algorithm, but a much more elegant solution is to use the Polak-Ribiere[2] extension of the Fletcher-Reeves algorithm which by using a slightly different expression for γ leads to an algorithm which in effect automatically resets the algorithm if necessary.[33]

2.2.4 The Algorithm

$$\mathrm{w_{i+1}} = \mathrm{w_i} + \frac{\mathrm{h}\epsilon}{\mathrm{x^t.h}}$$

$$\epsilon = d - \mathrm{x^t.w}$$

$$\mathrm{g_{i+1}} = 2\epsilon\mathrm{x}$$

$$\gamma = \frac{(\mathrm{g_{i+1}} - \mathrm{g_i})^t.\mathrm{g_{i+1}}}{\mathrm{g_i^t.g_i}}$$

$$\mathrm{h_{i+1}} = \mathrm{g_{i+1}} + \gamma\mathrm{h_i}$$

With initial values:

$$\mathrm{g_0} = 2\epsilon\mathrm{x}$$

$$\mathrm{h_0} = 2\epsilon\mathrm{x}$$

2.3 Results

The algorithm was tested against the LMS algorithm. Two filters, one LMS and one conjugate gradient filter (CG) were run in parallel modelling a FIR system.

The experiment was repeated with random coefficients in the 50 th order FIR system being modelled. The input signal was also varied: it was composed of a series of sinusoids of random amplitude and frequency. The convergence curves for these runs where averaged and plotted in figure 2.4.

The position of the sudden slight rise in error which can be seen on the conjugate traces even after averaging over runs can be shown to be related to the filter length. This may be explained as the algorithm resetting itself internally after having converged to a point and

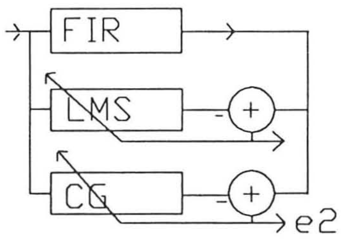


Figure 2.3: CG v LMS Convergence test system

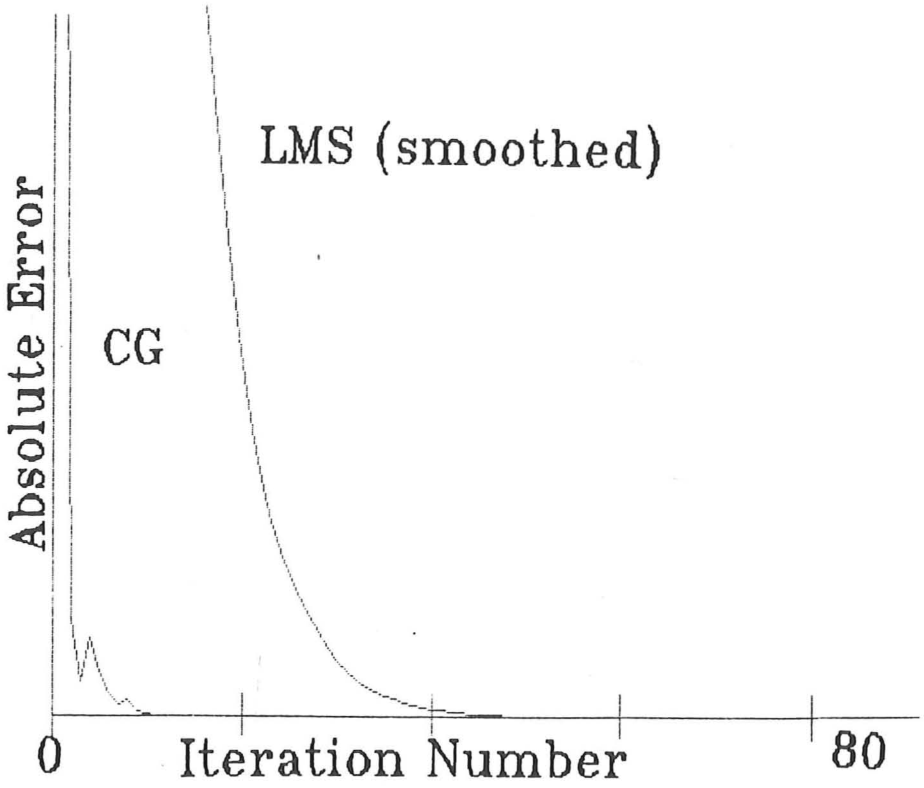


Figure 2.4: Plot of average absolute error over fifty runs.

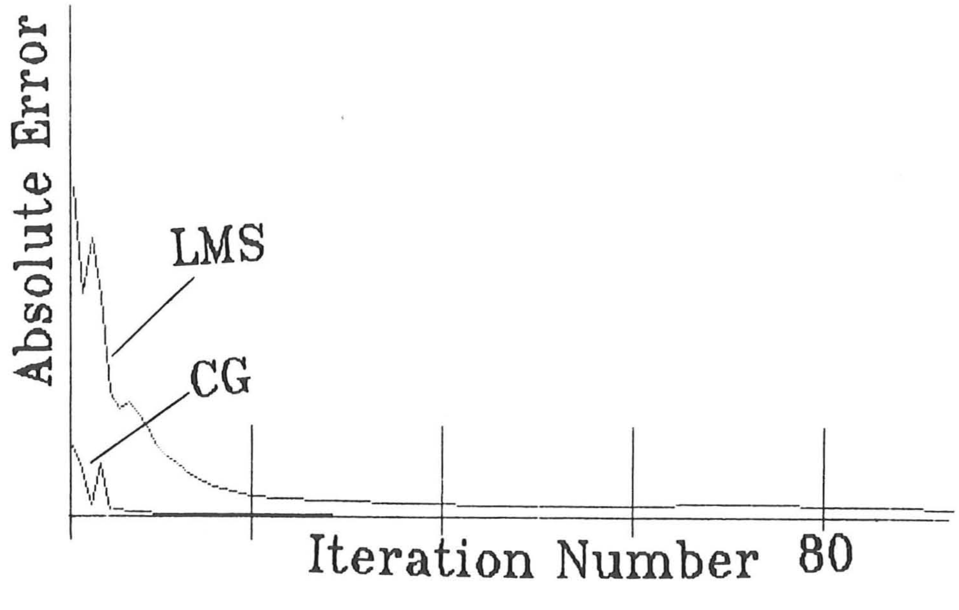


Figure 2.5: Typical Run with no averaging. Note smooth convergence of LMS algorithm compared with the jumps of the conjugate gradient algorithm.

yet having an error still present. In fact the alorithm continues to reset itself periodically as can be shown by monitoring the algorithm terms, however the error is not changed much by these resets as the solutions all lie very close to the optimal solution.

2.4 PARTAN Algorithm

There is a method closely related to that of conjugate gradients known as partial tangents (or PARTAN) [25]. This is of interest as it is insensitive to inaccuracies in the gradient determinations.

Starting at an arbitrary point x_0 the point x_1 is found by a single step of steepest descent. After that from a point x_k the corresponding y_k is first found by steepest descent from x_k , and then x_{k+1} is taken to be the minimum point on the line connecting x_{k-1} and y_k. The process is continued for n steps and then restarted with steepest descent. It can be shown that for a quadratic objective function the x_k are the same points as would arise from conjugate gradients[25]. The disadvantage of the algorithm is the need for two line searches per step as opposed to the one of Fletcher-Reeves. However, PARTAN is insensitive to inaccuracies in the line searches, which is a desirable quality for an adaptive

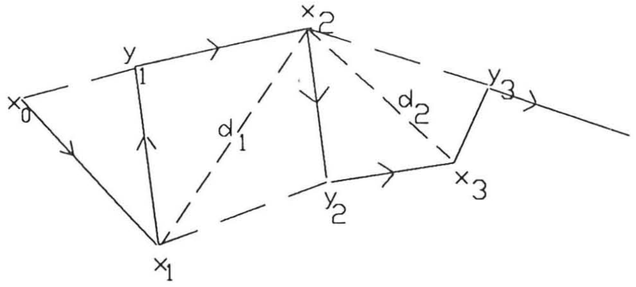


Figure 2.6: PARTAN Method

filter algorithm as the effect of noise can seriously upset line searches.

2.5 Conclusion

The algorithm performs well, giving significant increases in convergence rates over the LMS algorithm and requires only $O(4n)$ storage rather than $O(n^2)$ for other fast methods. It is necessary to note that the basic algorithm is operating more in a Least Squares rather than a Mean Least Squares mode. This can lead to unexpected results, especially for short filters with low frequency signals, and some form of error averaging may be needed to give a Mean Least Squares solution. Provided that this complication is considered, the algorithm is an improvement to LMS and may be used when methods such as Recursive Least Squares or Singular Value Decomposition cannot be applied due to storage or computational load constraints.

Chapter 3

Pitch Extraction

3.1 Introduction

An adaptive method of pitch estimation is presented which will estimate the pitch of a harmonic signal in noise to a high degree of accuracy, and track any slow drifts in the pitch of the harmonic signal. The approach is based on an adaptive notch filter and hence avoids the problems encountered in the use of integer length adaptive delay lines. The approach is particularly efficient when using an LMS notch filter and will allow increased performance of the notch filter by accurately fixing the reference signal frequency.

Adaptive pitch estimators for harmonic signals in noise can be difficult to design due to the nature of the error surface encountered if one attempts to adapt a fundamentally time stationary system. This approach solves this problem by utilising a time varying system.

An attempt was made to remove a harmonic interference from an archive gramophone recording using a standard FIR LMS adaptive notch filter [1]. This approach worked well in periods in which the harmonic interference was present with background noise of similar amplitude, but failed in periods with music or speech present. The amplitude of the music was many times greater than that of the interference and caused the adaptive notch filter to ring and destroyed its interference cancelling properties. As in the case encountered by Lim [35] it was decided to freeze the adaptive notch filter in these regions. However, as these regions persisted for tens of thousands of samples the reference sinusoids to the adaptive notch filter had to have their pitches accurately determined so that in the frozen periods the interference and anti-interference output from the FIR filter should stay synchronised.

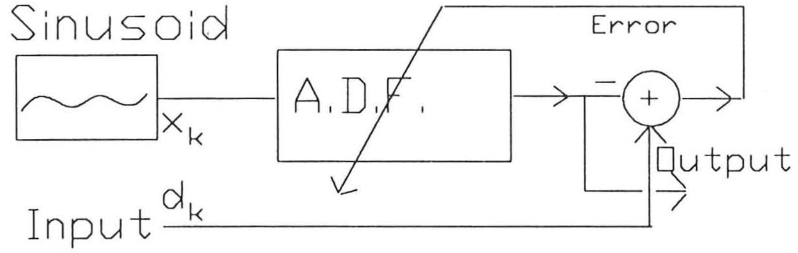


Figure 3.1: Standard Adaptive LMS Notch Filter

Thus an accurate and computationally efficient pitch estimator was required which could also track a slowly drifting pitch. Consider an LMS adaptive notch filter with one notch. Glover [28] analysed the LMS adaptive notch filter and showed its behaviour to include time varying aspects under certain conditions.

Let $C_i(z)$ be the Z-transform of the i_{th} coefficient of the adapted notch filter, with adaption coefficient α and reference sinusoid of frequency ω_r and amplitude C, $U(z)$ is the Z transform of the LMS weight update equation (an integrator) as shown in Fig 3.2.

$$C_i(z) = \frac{\alpha C}{2} U(z)[E(ze^{-j\omega_r T})e^{j\theta_i} + E(ze^{j\omega_r T})e^{-j\theta_i}]$$

Glover shows that the pole zero plot of E(z) would indicate poles at $z = e^{\pm j\omega_d T}$ and $E(ze^{-j\omega_r T})$ represents a counterclockwise rotation of $E(z)$ through angle $\omega_r T$.

Therefore the pole-zero plot of :

$$[E(ze^{-j\omega_r T})e^{j\theta_i} + E(ze^{j\omega_r T})e^{-j\theta_i}]$$

would show poles at $\pm(\omega_r + \omega_d)T$ and at $\pm(\omega_r - \omega_d)T$. This rotated spectrum is filtered through $U(z)$ to give $C_i(z)$. Each weight therefore consists of the sum and difference frequency of ω_r and ω_d. $U(z)$ is strongly low pass, hence the difference frequency dominates. For the FIR delay line $\theta_i = \theta - \omega_r T[i-1]$ which shows that the coefficients c_i are a sinusoid of frequency ω_r with each weight varying as a sinusoid at frequency $\omega_r - \omega_d$. Glover refers to this as the sinusoid in the coefficients 'moving' at a rate equal to the difference frequency between the desired and reference frequencies. This may be viewed

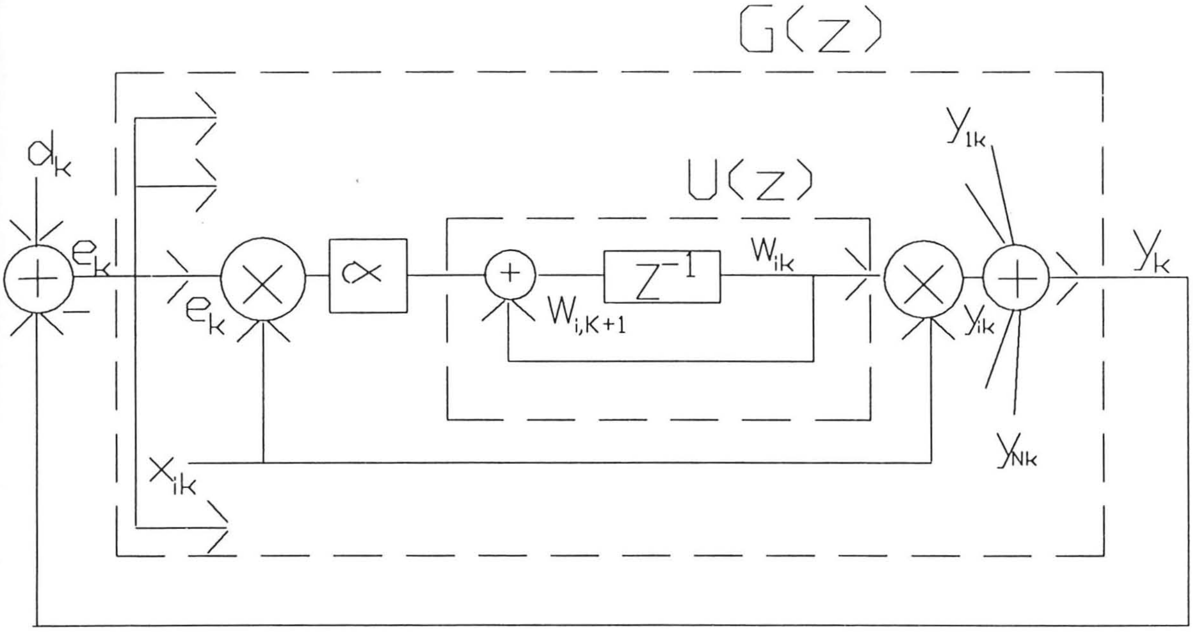


Figure 3.2: Notch Filter Analysis Diagram

as a heterodyning process and is a time varying aspect to the solution for an adaptive notch filter; it may also be considered as a simple case of a quasistationary filter using a slowly varying phase reference at the desired frequency. It is this time varying aspect that the adaptive pitch estimator (APE) will use.

A single notch adaptive filter is constructed and the motion of the sinusoid is detected by the following algorithm:

$$P_{n+1} = P_n + \mu_p \frac{1}{\phi_k} \frac{\partial c_k}{\partial t}$$

Where ϕ_k is the gradient of the coefficients with respect to i at coefficient k, and n is the time index. The coefficient of pitch adaption μ_p sets the sensitivity and convergence rate of the pitch estimator.

The time derivative may be replaced by:

$$\frac{\partial c_n}{\partial t} \approx c_{nk} - c_{k(n-1)}$$

And the gradient by:

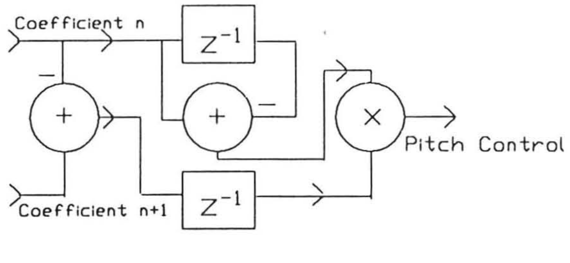


Figure 3.3: Coefficient Motion Detector

$$\phi_k \approx c_{n(k-1)} - c_{n(k+1)}$$

A simpler form of the algorithm which does not require the division and hence avoids any possible divide by zero problems is:

$$P_{n+1} = P_n + \mu_p sgn[\phi_k \frac{\partial c_k}{\partial t}]$$

In practice Glover's analysis does, not always apply. If we consider the frequency response of the FIR stage of the adaptive notch filter we can see that there are an infinite number of possible solutions for the case of a single sinusoid. The above algorithm, however, requires the particular solution with a sinusoid in the coefficients to be adopted. If this solution is not adopted or is disrupted the system will fail. Occasionally in practice it may be necessary to force the filter to assume the sinusoidal solution. In this case we must force its stopband to include all frequencies except the reference frequency. This is easily done by injecting some white noise with the reference sinusoid. This gives a good stable sinusoid in the coefficients as the filter should block as much of this white noise as possible to minimise the mean square error.

The other two disruptions of the sinusoidal solution are a transient effect as the system is first set running and a destabilising effect caused by varying the pitch of the reference sinusoid too rapidly. In this case the assumption of quasistationarity breaks down and the reference input is no longer a good approximation to a pure single frequency.

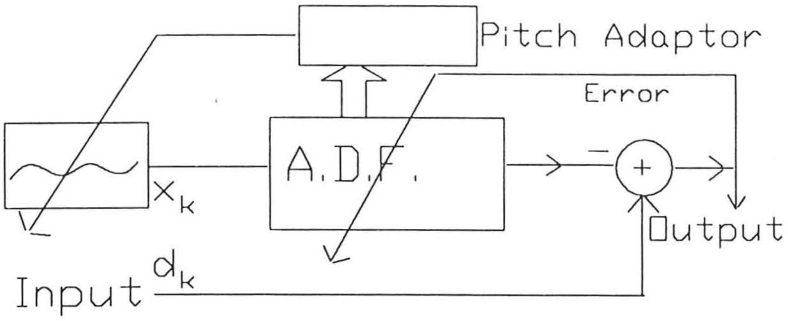


Figure 3.4: Pitch Tracking System

3.1.1 Harmonic Signal Case:

In the case of harmonic signals the accuracy of the system may be increased by using a reference sinusoid of frequency near one of the higher harmonics present rather than the fundamental, as this will increase the difference frequency and increase the coefficient sinusoid movement by a factor equal to the harmonic number. Thus leading to a larger effect for the same pitch error in the fundamental.

3.1.2 Limitations:

The limitations on the accuracy of the algorithm are set by the adaption noise found on the coefficient values. This adaption noise can be estimated from the expression for the minimum mean square error. It is clear that it may be reduced by reducing the adaption coefficient of the notch filter. The bandwidth of the notch filter is set by this coefficient which in turn determines the range of frequency error over which the above analysis applies. Thus there is a trade off between final pitch accuracy and initial lock range. The initial lock range being defined as the maximum difference in frequency between the desired and reference signals for which the system will converge.

The notch bandwidth is given by:

$$BW \approx \frac{N\alpha C^2}{2T} rads/s.$$

Where N is the filter length.

Limitations on the rate of convergence are set by the need to prevent instability due to undermining the quasistationary assumptions. The changes in the pitch of the reference oscillator must therefore be suitably slow, this may be achieved by using a suitably small pitch adaption coefficient μ_p.

3.1.3 Results:

The algorithm was found to be capable of high accuracy pitch estimates in quite high levels of noise and was capable of pitch convergence in SNR of -15dB. In the case of SNR of 10dB the system could attain pitch accuracy estimates in the order of 1 part in four thousand. A series of convergence curves are shown in Fig:3.5. The occasional initial divergences are caused by transient effects as the data enters the filter.

Fig. 3.5 Pitch tracking curve .

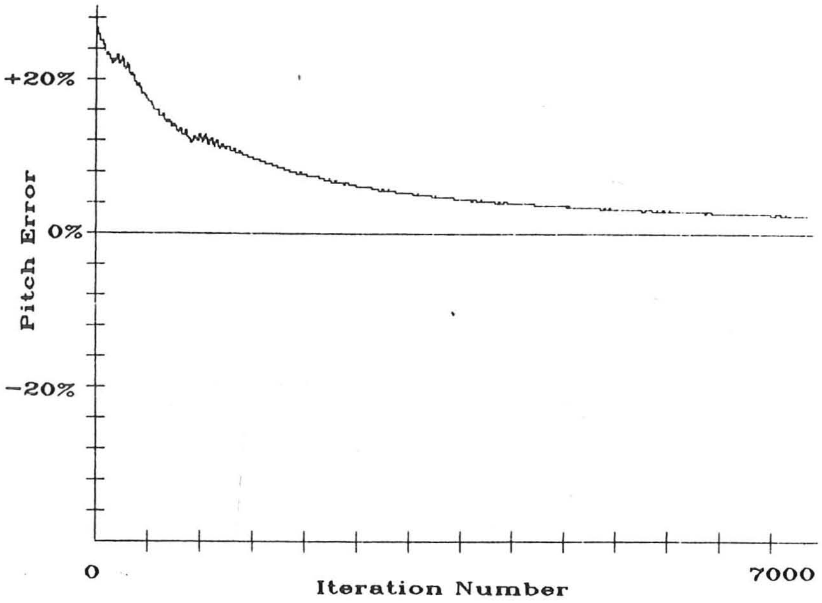
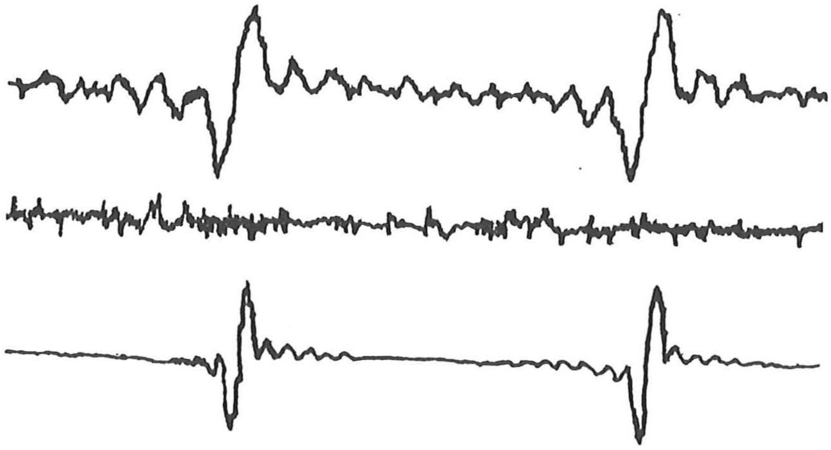


Fig. 3.6 Input Musical Waveform (top), Oscillator waveform (bottom) and Output. (quiet section with no music).



3.1.4 Conclusions:

The system is best suited to applications requiring high accuracy pitch estimates on stationary signals for which a low accuracy pitch estimate is already known. The necessary computations for the system are based around a FIR filter, thus making it easily implementable in hardware. The filter may be quite short in practice and in the case of an adaptive notch filter to remove a single sinusoid the pitch estimator may be added at very little extra computational cost by using the cancelling filter as the pitch estimating filter. This leads to deeper notches and increased performance and as such may be a simple improvement that may be added to FIR adaptive notch filters. It is clear that there are many possibilities for variations on the above theme that may be well suited to other applications.

Chapter 4

Image Registration

4.1 Image Registration Utilising Multidimensional LMS Adaptive Filters.

The need for image registration algorithms arises in many fields such as airborne ground sensing, aircraft navigation, medical imaging. In the latter separate frames of an image must be registered in the presence of involuntary patient movement. This problem arises wherever there is relative movement between the image sensor and the image scene. In practical cases the images may be degraded by noise, may contain weakly spatially varying shifts or one of the two images to be registered may contain objects not found in the other or moving independently of the rest of the scene. All of these effects can significantly reduce the performance of current image registration methods such as phase correlation.

4.1.1 Current Registration Methods

Broadly speaking there are two practically used approaches to image registration. The first of these is a syntactic approach which normally involves the extraction of simple basic image elements such as line segments or boundaries. These are compared in the two images and some search method is employed to register the two images. This method works well for simple shapes in clean images under constant conditions . It is also necessary that the image contain the features employed, such as sharp line segments. In general, however, images can be noisy, have variable content, be complex and arise from environments with varying lighting and so on. Thus this method cannot usually be applied well to general images.

The second method is that of phase correlation. This method is based on properties of the Fourier Transform.

Consider two images I_1 and I_2 which are to be registered. The images are assumed to be related by a spatially invariant shift.

Define $F_1(\omega_x, \omega_y)$ to be the Fourier transform of I_1, $FT(I_1)$.

Define $F_2(\omega_x, \omega_y)$ to be the Fourier transform of I_2.

Then by the shift theorem:

$$F_1 = e^{-j(\omega_x \Delta x + \omega_y \Delta y)} F_2$$

Where Δx is the x direction shift and similarly Δy.

Thus the function:

$$P(x, y) = FT^{-1}(\frac{F_2}{F_1})$$

returns a delta function at the shift value.

However noise effects and the fact that many shifts are not constant over the image area mean that this delta function may become distorted or lost under noise in the phase correlation output image.

4.1.2 The New Method

In the case of images the new method employs a two dimensional Least Mean Squares adaptive filter which makes almost no a priori assumptions about the images or their degradations. This allows the method to be used with general images and consequently it is not limited to images with sharp edges or similar artifacts as may be required by syntactic methods.

4.2 Derivation

The derivation of the multidimensional LMS adaptive filter is closely related to that of the one dimensional case. The derivation is given for a N dimensional adaptive filter with each dimension of order m.

Consider an input tensor $\mathbf{x}$ of rank N order m and an N rank weight tensor $\mathbf{w}$. Both tensors elements have N indices $i \ldots z$.

The N-dimensional adaptive linear combiner is given by (k is the iteration index):

$$y_k = \mathbf{w_k x_k}$$

Where the operation $\mathbf{wx}$ is defined as the inner product, that is:

$$\sum_{i=1,m} \sum_{j=1,m} \cdots \sum_{z=1,m} x_{i \ldots z} w_{i \ldots z}$$

The error may be defined as the difference between a desired output d_k and the actual output y_k:

$$\epsilon_k = d_k - y_k$$

$$\epsilon_k = d_k - \mathbf{w_k x_k}$$

Hence:

$$\epsilon_k^2 = d_k^2 + (\mathbf{w x_k})(\mathbf{w x_k}) - 2 d_k \mathbf{w x_k}$$

We now assume that ϵ_k, d_k and $\mathbf{x_k}$ are statistically stationary and consider the expectation of the error squared over k. This assumption can have some important implications for work with practical images as in many cases the degree of stationarity can be less than is normally found with many practical one dimensional signals.

$$E(\epsilon_k^2) = E(d_k^2) + \mathbf{w_k} E(\mathbf{x_k x_k}) \mathbf{w}_{K} - 2E(d_k \mathbf{x_k}) \mathbf{w_k}$$

We can define $E(\mathbf{x_k x_k})$ as the N dimensional correlation 'Tensor' of rank 2N, $\mathbf{R}$. Likewise the cross-correlation tensor $E(d_k \mathbf{x_K})$ denoted, $\mathbf{p}$.

$$E(\epsilon_k^2) = E(d_k^2) + \mathbf{w_k R w_k} + 2\mathbf{p} w_K$$

This shows that the mean square error is precisely a quadratic function of the weight values. Thus the error surface is unimodal with no local minima. It is this quality that will allow some form of gradient search to converge to the optimum solution.

Having shown the nature of the error or *Performance* surface the problem is now to find a search algorithm to find the minimum. Again the cue is taken from the 1D case of the derivation of the Widroff-Hoff LMS algorithm [1] and a simple gradient search is employed.

An adaptive algorithm may be formed by the following approach:

$$\nabla\epsilon_k^2 = \frac{\partial\epsilon_k^2}{\partial\mathbf{w}} = 2\epsilon_k\frac{\partial\epsilon_k}{\partial\mathbf{w}} = -2\epsilon_k\mathbf{x}_k$$

We can specify a steepest descent algorithm:

$$\mathbf{w_{k+1}} = \mathbf{w_k} - \mu\nabla\epsilon_k^2 = \mathbf{w_k} + 2\mu\epsilon_k\mathbf{x_k}$$

Where μ is the adaption coefficient. Explicitly in the case of 2D ADFs the update equation is:

$$w_{xy(k+1)} = w_{xyk} + 2\mu\epsilon_k x_{xyk}$$

The derivation of the LMS adaptive filter is independent of the dimensionality of the data, thus the same derivation can be used to lead to 2D,3D or ND ADFs. This derivation also shows us that the error surface is unimodal even for higher dimensionality data. The fact that 2D and 1D ADFs are based on the same theory also allows us to predict with reasonable confidence the behaviour of 2D ADFs by analogy with the 1D case, however care must be exercised in this practice as some of the problems encountered in 2D have no analogy in 1D; also the nature of practical image data may differ from practical 1D signals.

4.3 Image Registration System

The basic shift registration system is shown in figure 4:1. The reference image may in practice often be one of the previous frames of a series of image frames. The adaptive filter is run and its coefficients are updated. The filter will perform a gradient search to minimise the mean square error image. In the case of a shift this will occur when the output image has been correctly shifted by the filter to be registered with the reference image. The maximum shift that the system can cope with is given by the filter size and consequently its maximum spatial delay. The system is arranged so that there is in effect a spatial shift of half of the filter size between the two images so that the filter can cope with a positive or negative delay of magnitude less than m/2.

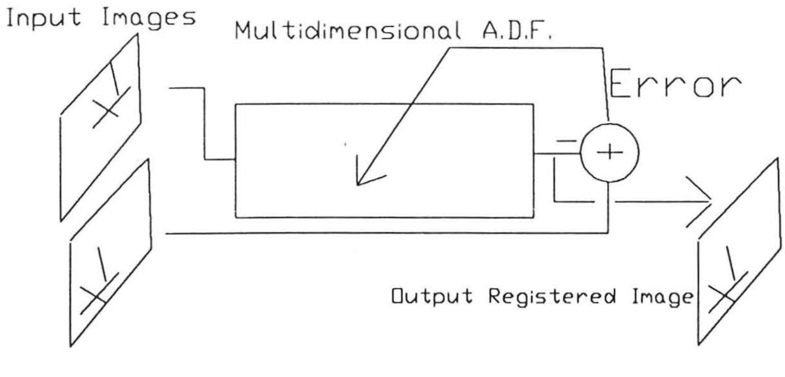


Figure 4.1: Adaptive Image Registration Using 2D Filters.

The particular path over the image that the filter runs can be important in the case of spatially varying shifts.If the variation is occurring more quickly in one of the two axis directions it would be advisable to track the filter along the other axis direction to allow the filter more iterations in which to update to the new shift. Consequently if images have a spatial variation which is mainly a function of the x coordinate it would be advisable to run the filter down the lines of constant y coordinate only advancing in x coordinate at the end of each y line.

It is possible to improve the performance of a given order of filter by ensuring that there is no large relative dc offset between the images. This may occur if the two images were captured under very different illumination levels. Although it may be expected that the filter could compensate for such an offset the process of doing this degrades the performance of the system as it is utilising zeroes of the filter that may be more usefully applied elsewhere. Correspondingly it is therefore often useful to high pass filter the images before registration if there is a possibility of large dc offsets.

4.4 The Adaption Coefficient

The value of the adaption coefficient is important in determining the mode of operation of the system. As a generalisation the nearer the adaption coefficient approaches the divergence value so the output image becomes sharper and the system can track spatially varying parameters more rapidly. However, smaller values of the adaption coefficient give higher noise immunity. One of the main problems in the practical implementation of adaptive image registration is the choice of a suitable adaption coefficient. In fact because of the nonstationary statistics of practical images it is necessary, in order to improve performance, to implement a self regulating adaption coefficient which varies with the image. A simple yet reasonably effective approach is to update the coefficient of adaption as a function of the input power to the filter. This approach, which is based on the work of Yassa [3], has been simplified to reduce computational load. A power estimate is obtained from the row of unseen data about to enter the filter, this is then filtered by a single pole filter whose feedback factor can be adjusted to control the smoothing of the power estimate.

Incident power:

$$P_k = \sum_{i=1,m} x_{i,1}^2$$

Smoothed power:$F_k = F_{k-1}g + P_k$

Where g is the feedback factor. The adaption coefficient is updated by:

$$\mu = \frac{\mu_{int}}{F_k}$$

Where μ_{int} is the initial adaption coefficient.

If it is desired to use the filter output image directly or to track spatially varying shifts a larger adaption coefficient may be needed. Even with the above adaption coefficient updator divergence may still occur on certain parts of some images so it is necessary to have a divergence detector. This may be done by checking that the output pixels do not exceed a certain value. If this should occur the μ_{int} value is immediately reduced by a factor (usually about 0.7) and the filter coefficients reset to either all zero, or all zero with a value one at the coefficient corresponding to a recent shift estimate.

4.5 Filter Solutions

As may be expected from the one dimensional analogy the performance of the two dimensional LMS algorithm depends on the eigenvalue spread of the input data, and the time taken to converge to the correct shift estimate is reduced as the whiteness of the data increases. In order to extract the shift estimate explicitly from the coefficients of

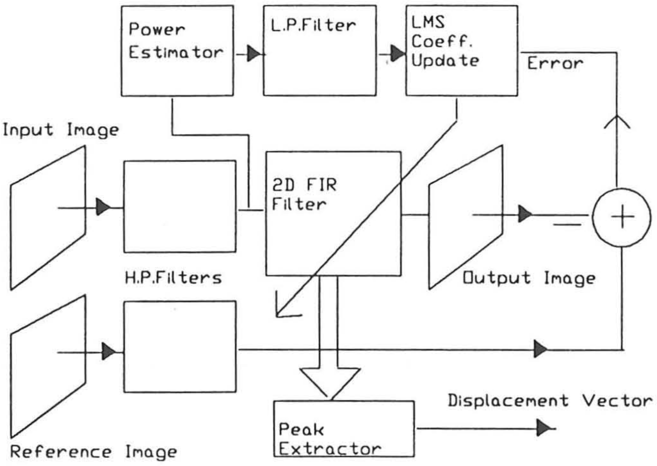


Figure 4.2: The Complete Registration System.

the adapted filter we must consider the filter solution. In the case of white input signals and integer shifts the filter converges to a Kronecker delta function at the shift delay as may be expected, and for non-integer shifts it converges to an interpolator. Almost all practical images do converge to solutions with peaks at the delay value. The dominance of this peak is however determined by the whiteness of the input image, whiter images giving a more dominant peak.

A simple peak detector may be used to extract the shift and interpolation of the peak may be used to obtain non-integer shift estimates. By using such a shift extractor at regular intervals a displacement vector map may be built up showing the displacement vector over the image.

4.6 Output Image

As a byproduct of the processing, a registered and shifted corrected image is produced at the filter output. However, apparently due to the sensitivity of the LMS algorithm to eigenvalue spread and the differing convergence rates for different eigenvalues, the solutions reached are near optimal but lead to subjectively slightly blurred images. These problems may be alleviated by using adaption coefficient values near to divergence, but this can lead to a risk of of divergence occurring. Another approach is to use the displacement vector field to 'manually' shift back the original image. This is currently done with many registration methods.

4.7 Performance

The filters were run on clean images with a spatially stationary shift. In practice a lock onto the correct shift is achieved in thirty or forty pixels. The adaption coefficient value may be small such that the output image is very blurred but yet give good results for the displacement vector. All the results shown were generated using images 80 by 80 pixels which were quantised to 6 bits.

The same experiment was repeated with the addition of equal levels of white uncorrelated noise to each of the images and the displacement vector field recorded. As may be expected the filters took longer to lock onto the shift, and in the case of high noise levels, the lock could be lost and regained throughout the picture. However, even at these high noise levels further increases in performance could be obtained by the utilisation of averaging of the displacement vector by use of the property of continuity of the displacement vector [12].

Finally the filters were run on an image pair in which the filter input image had been obtained by a spatially varying shift which was a function of the x coordinate.

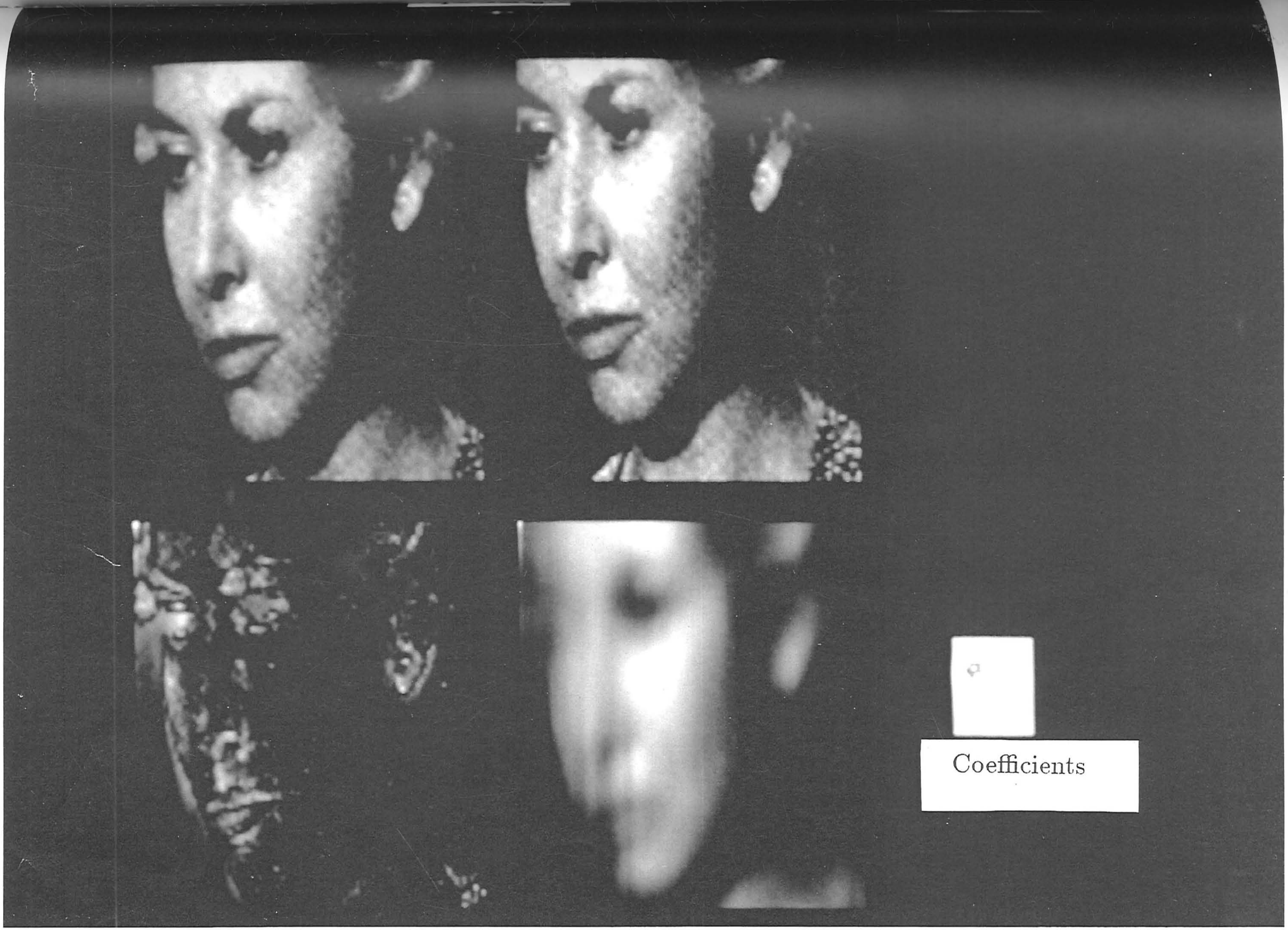
Photographs

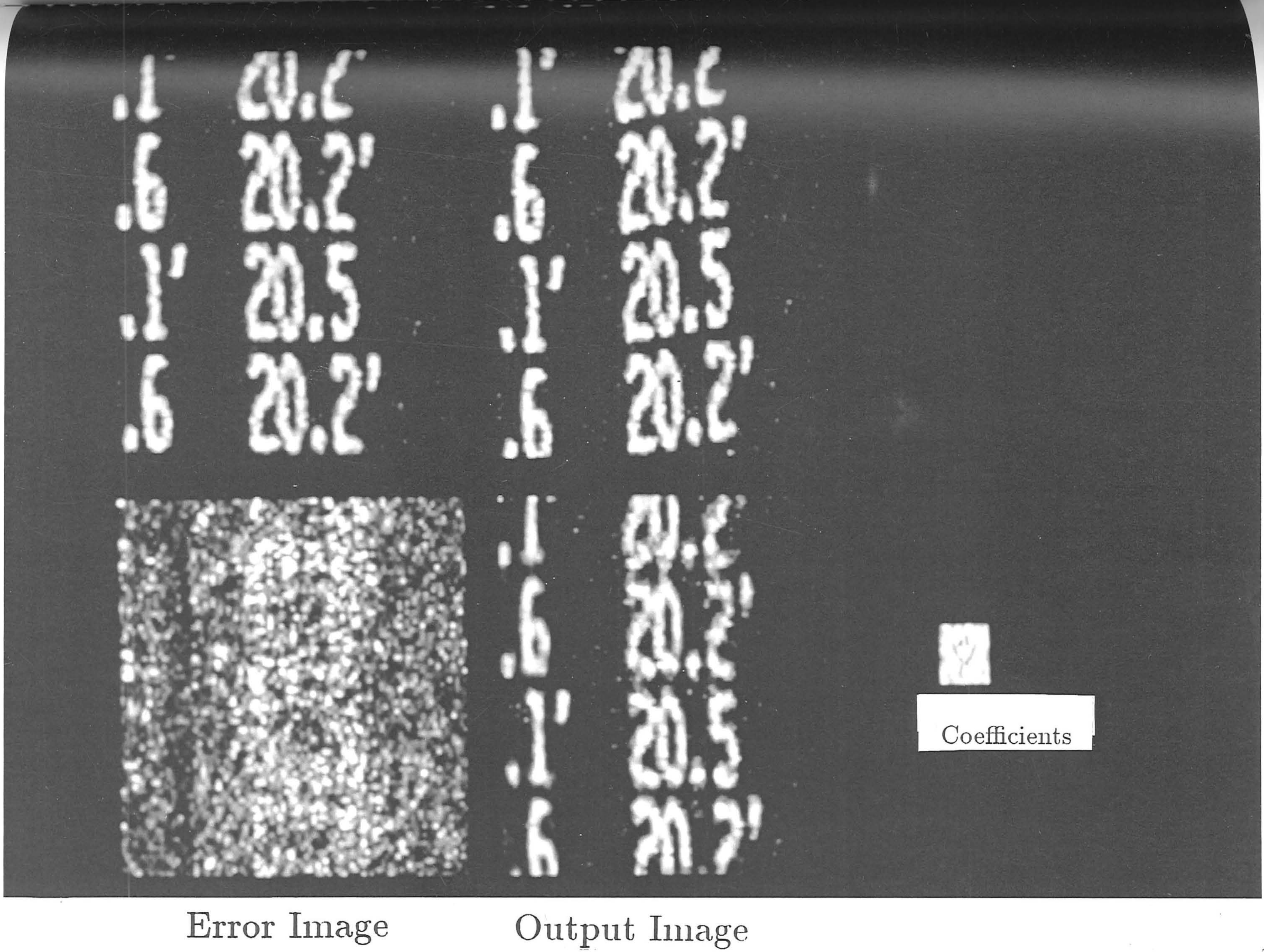
The photograph format is:

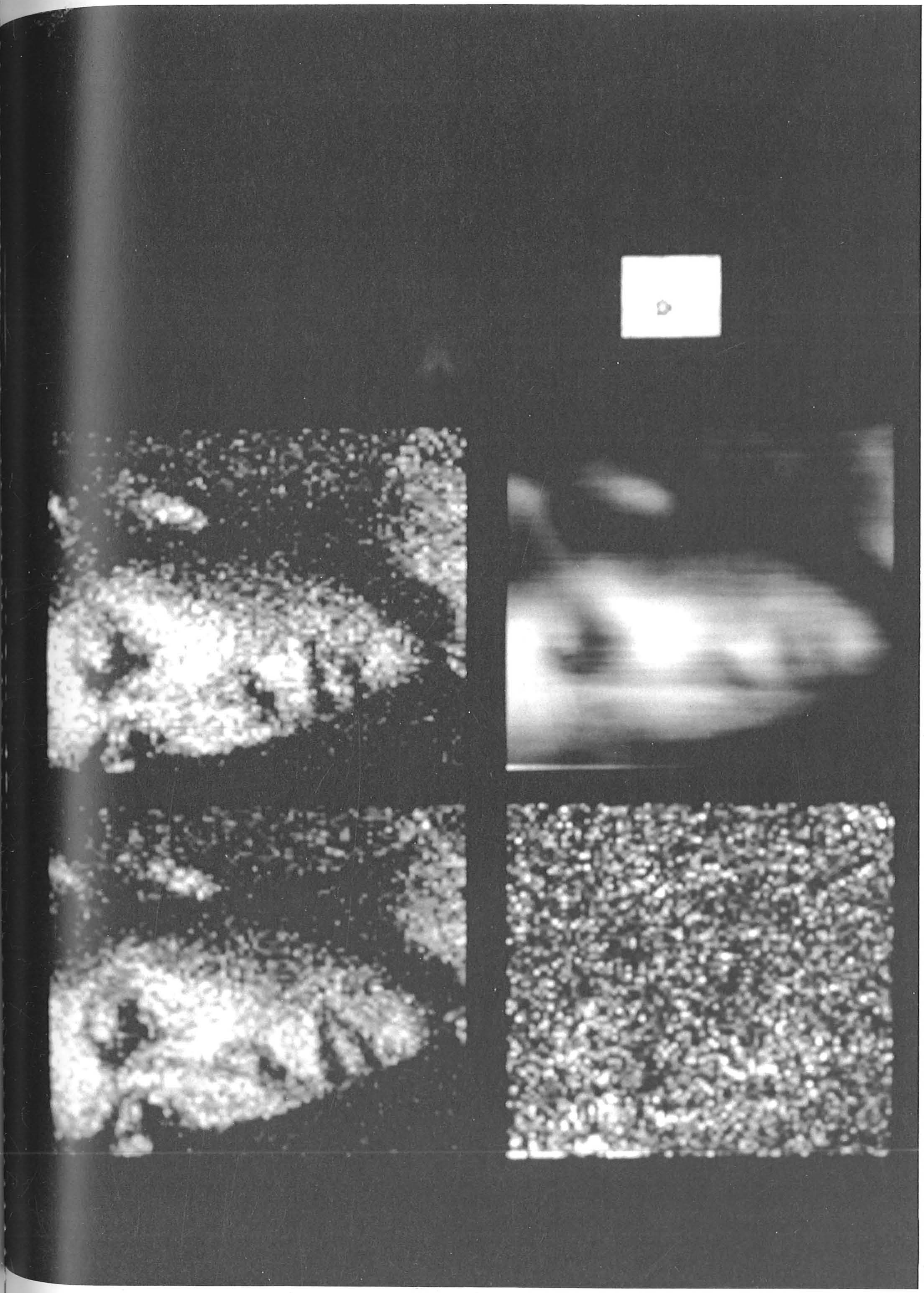
- Top left reference image.
- Top right input image.
- Bottom left error squared.
- Bottom right output image.
- Small Plot of final coefficient values.

The photograph and results are in the following order:

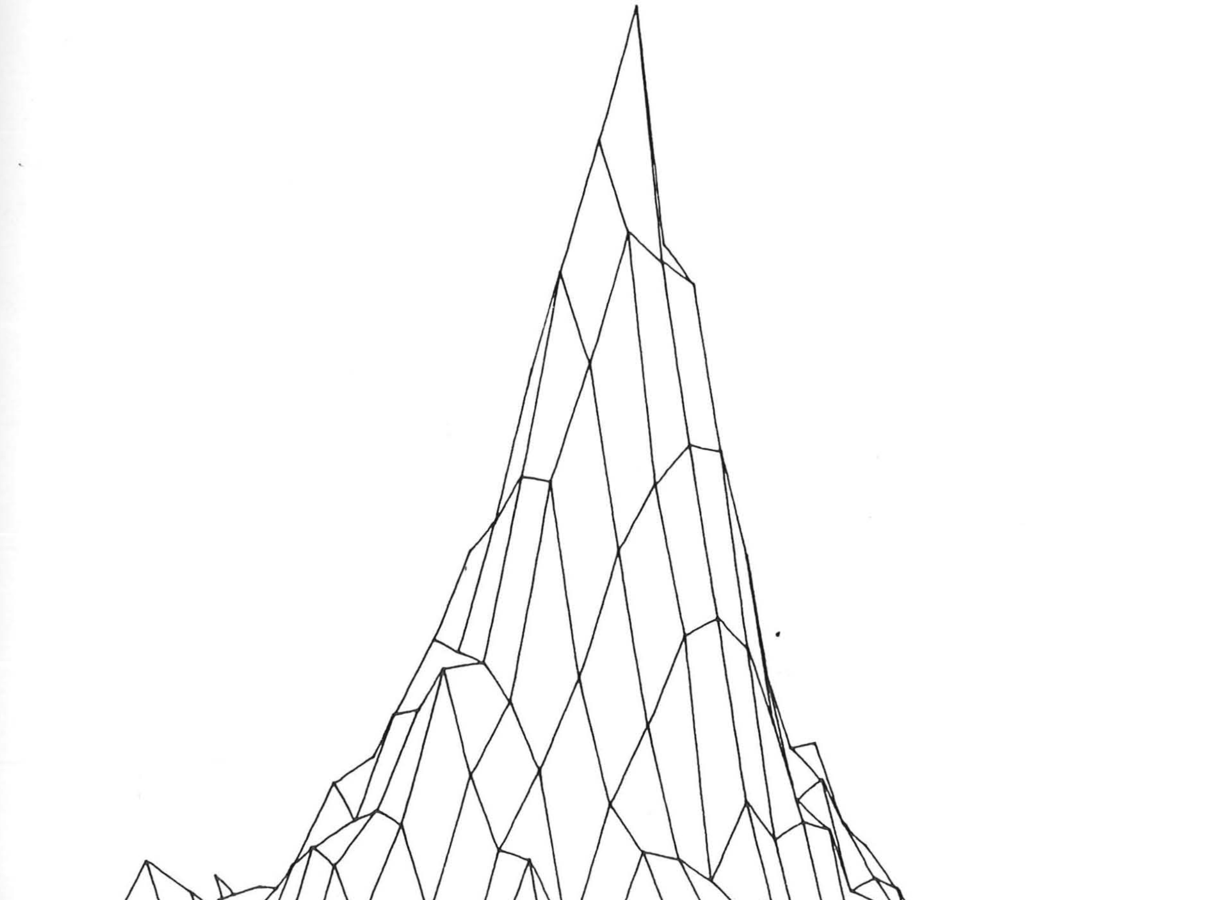
1. Results of constant shift case. Filter 20 by 20 with a (5,4) shift and the initial adaption coefficient set to .008.
2. Results of spatially varying shift case. Filter 12 by 12 with a shift $\delta y = 5 * \sin(.004 * x)$. Initial adaption coefficient set to .004.
3. Results of constant shift case with additive Gaussian white noise . Filter 20 by 20 with a (3,2) shift and the initial adaption coefficient set to .004.
4. Plot of final coefficient values for spatially stationary experiment.
5. Plot of final coefficient values for spatially varying experiment.
6. Plot of vector displacement output for spatially stationary experiment.
7. Plot of vector displacement output for spatially varying experiment.


Coefficients
Error Image
Output Image

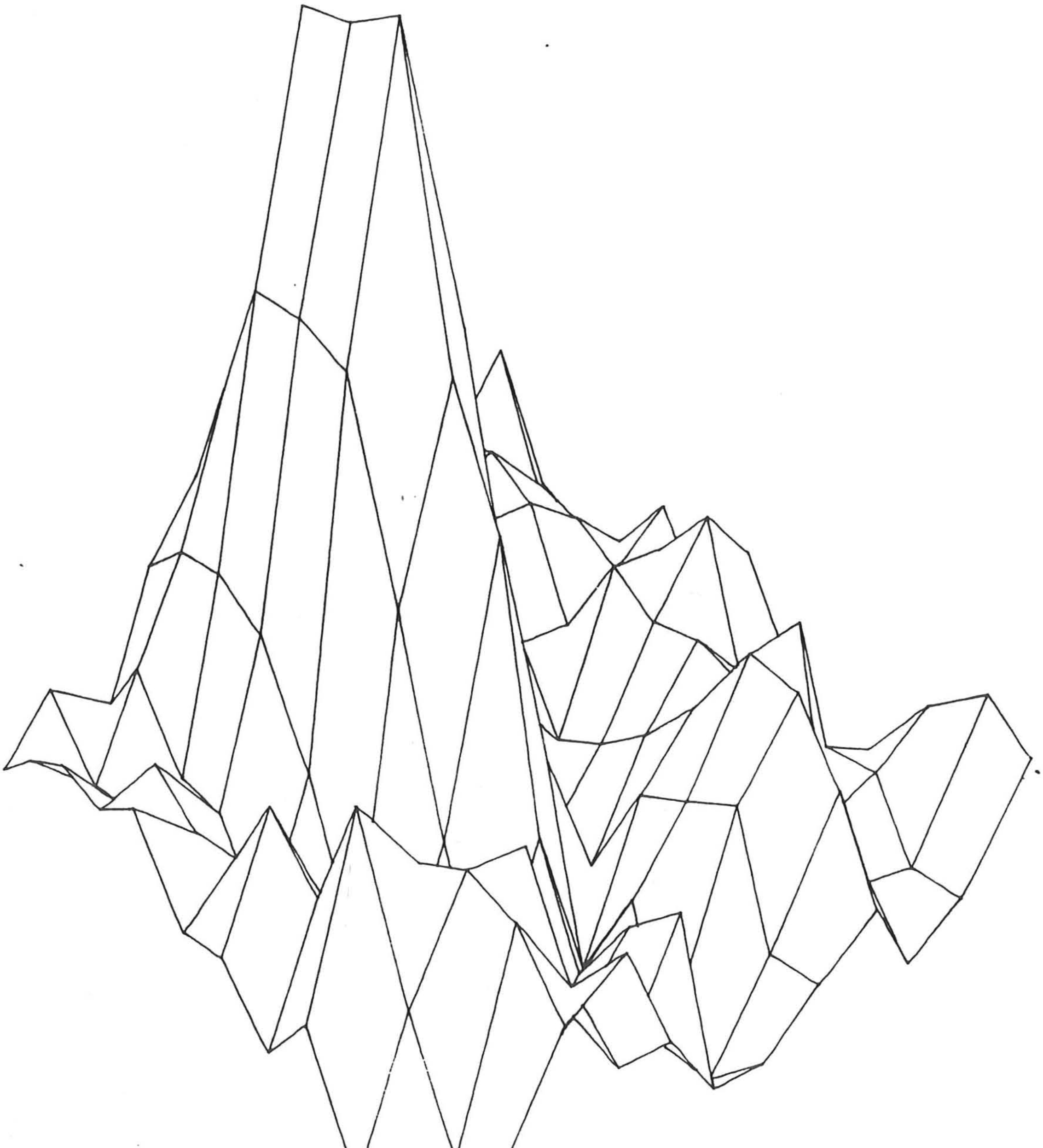

Coefficients
Error Image
Output Image



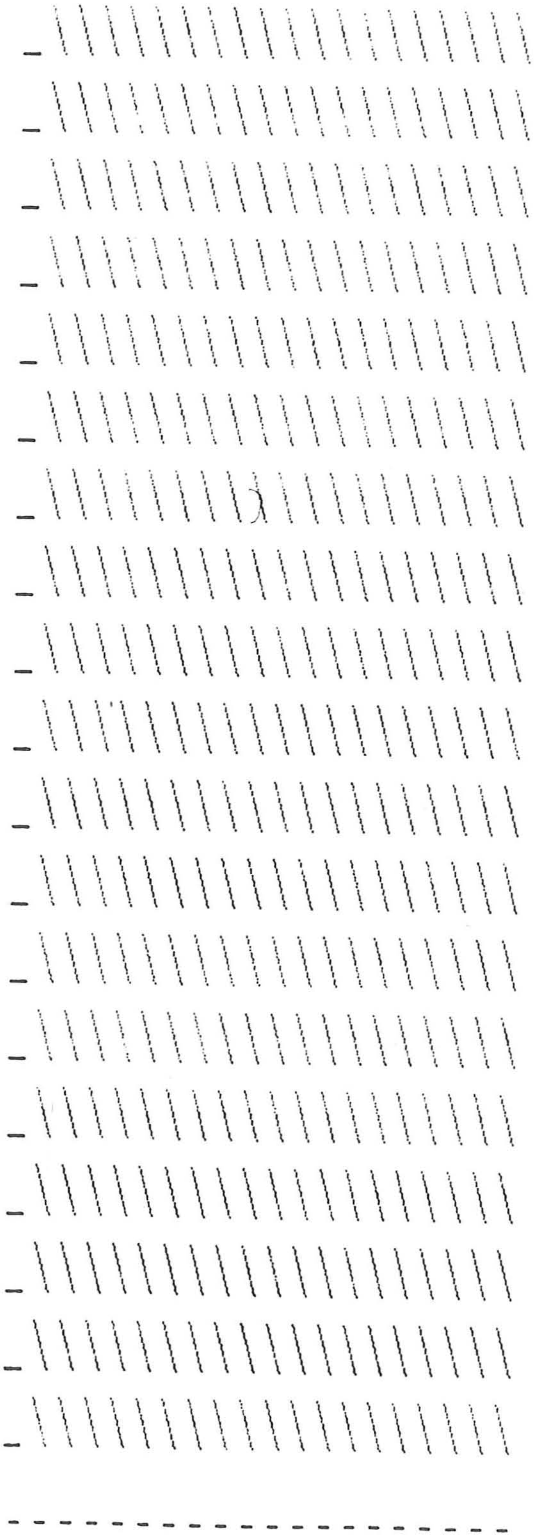
Final coefficients result 1



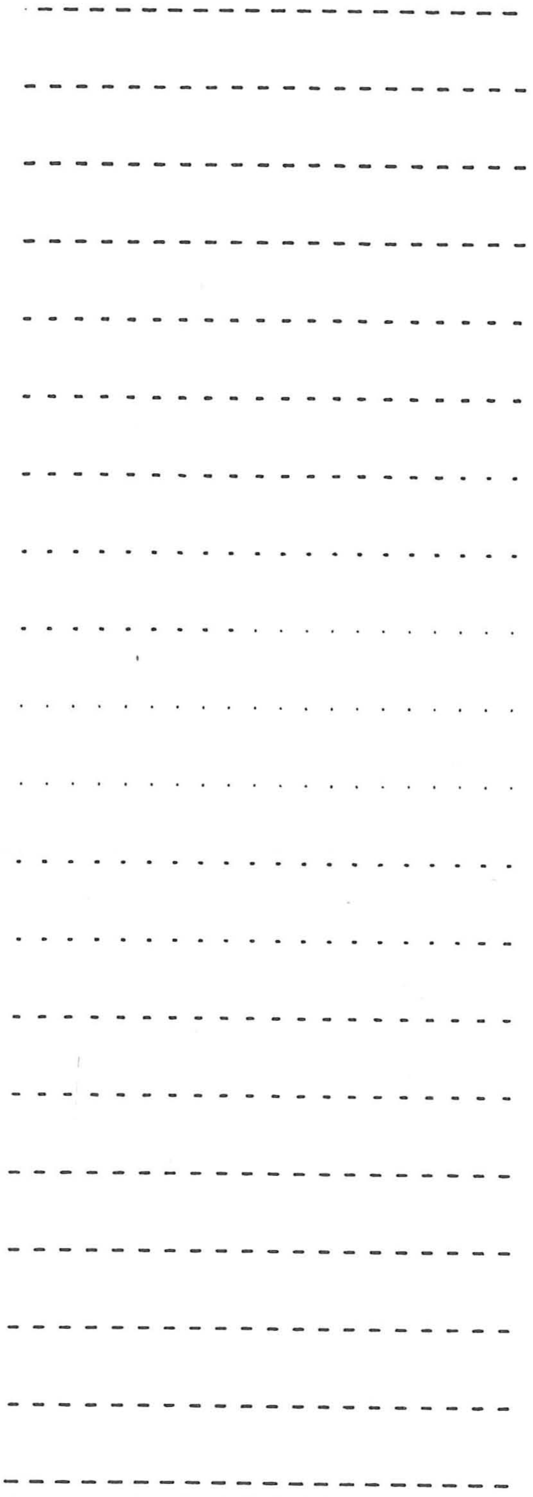
Final Coeffcients result 2



Vector Displacement result 1



Vector Displacement result 2



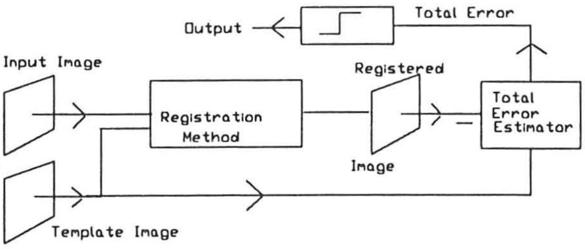


Figure 4.3: Template Matching System.

4.8 Computational Load

For spatially stationary shifts the filter need only be run over small portions of the image. The filter size is set by the largest shift to be dealt with, but this may be reduced by calculating displacement vector fields from a decimated image. The algorithm is inherently parallel and suited to the use of current adaptive digital filter integrated circuits.

4.9 Template Matching Possibilities

The following system may be used to implement the multidimensional adaptive filter as a template matcher by using the reference image of an object as the desired input and a scene as the filter input. The filter correlates the object in the scene and reference and shifts the scene. Thus the error image can be used as a measure of the match over the relevant region, thus providing a weak shift invariant template matching system. The use of higher order,for example 3D, adaptive filters may be utilised to register over a sequence of frames by the same principle as is outlined above.

4.10 Conclusion

The LMS multidimensional adaptive filter can provide a new approach to image registration which is well suited to parallel implementations. The system also has some useful properties, giving the noise immunity of correlation based methods but not requiring a search over possible shifts.It also has the ability to deal with weakly spatially varying shifts, unlike transform based methods. The adaptive filter can also handle weak image degradations as its frequency response is not constrained and may adapt to correct differences in the two images caused by sensors or uneven illumination and so on. The highly non-stationary nature of real images means that care must be taken to optimise the adaption coefficient continually if sharp output images are to be obtained. This appears to be related to the unequal convergence rates for the different eigenvalues of the image correlation matrix as may be expected from one dimensional work [1]. Setting the adaption coefficient close to the optimal can lead to streaking effects as the filter momentarily diverges since the optimal adaption coefficient value changes with the local statistics of the image. In general the method provides a useful and easily implemented approach which will perform better than the transform based methods for many practical cases due to its ability to overcome weak spatially varying effects.

Part II

Chapter 5

Pattern Recognition

5.1 Introduction

A standard application of connectionist networks is in pattern recognition. This half of the thesis attempts to define the abilities that a pattern recognition system requires, and to investigate the performance of the extended vector space connectionist model with respect to these abilities.

5.2 The Problem

In the pattern recognition problem as addressed in this thesis, a system is required to output a class index which reflects a particular class to which its input signal belongs. Consider a decision rule which partitions a space into regions Ω_i ,$i = 1, \ldots, N$, where N is the number of classes. An object is classified as coming from class ω_k if its corresponding vector representation $\mathbf{x}$ lies in region Ω_k. The boundaries between the regions are called decision surfaces. The task can be made more difficult by the effect of noise on the data and the variability of data for patterns belonging to the same class. The following is a list of desirable properties for a general pattern recognition system.

5.2.1 Adaptivity and Learning

A suitable pattern recognition system will need to be adaptive for most applications. The necessary rules to define most real-world recognition tasks are too complex to be analytically derived. This becomes more apparent when the effects of noise or class

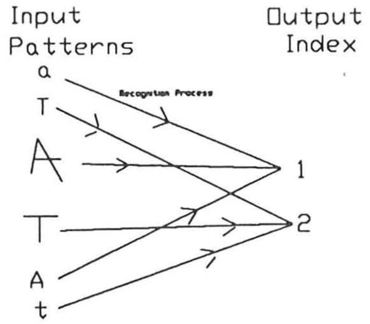


Figure 5.1: Pattern Recognition Problem

variation are considered. These problems may be circumvented by utilisation of a self learning system, which by experience of the recognition problem adapts itself to solve the problem.

5.2.2 Interpolation and Generalisation

Once a system has been trained using the members of a training set it is hoped that the system would correctly classify a new pattern which was not in the training set but a member of a class defined by it. This property is often referred to as interpolation or generalisation as it is the ability of the system to interpolate the classifier output between training patterns. There is an implicit assumption in the concept of interpolation that patterns belonging to the same class will cluster to a similar region of the decision space. This may not always apply, for example in the case of a capital and a small letter 'A'. However, from an analytical viewpoint this may be considered as two classes mapping to the same output value. It would be hoped that capital A letters will cluster and so on.

5.2.3 Learning Speed

It is obviously an advantage if the recognition system can adapt itself to a state of 'good' performance with as few pattern presentations as possible. The ability to adapt quickly is also of advantage as it also enables the recognition system to respond to changes in the noise or class statistics.

5.2.4 Certainty of Learning

It is impractical to have systems which may get stuck in their learning processes and not reach their optimal performance; we thus require a system that will not suffer from local minima problems. These problems would require the system to be supervised and reset, incurring multiple runs and consequently large computational expenditure.

5.2.5 Multiple Class Problems

Although most recognition tasks concerning multiple classes may be reduced to a series of one class recognition problems, this is in general highly inefficient in terms of computational load. An ideal system should be able to perform multiple class recognition.

5.2.6 Parameter Tuning

For a system to be practically implementable a method of tuning any of the system network parameters must be found to prevent the need to repeat training runs in order to optimise these parameters.

5.2.7 Parallelism

The system should be highly parallel to allow high speed operation. A parallel implementation may also allow the processing resources to be distributed and so it may be composed of many simple processing units.

5.2.8 Probabilistic Properties

A pattern recognition system must take account of the fundamental probability distributions relating the observations and the true classification of the object. An understanding of the importance of this may be gained from [17][26] and the following Bayes' recogniser section.

5.3 Traditional Approaches

Pattern recognition methods can be broadly classified into two distinct groups, statistical and syntactic. This thesis is primarily concerned with statistical pattern recognition rather than syntactic recognition. There are many methods which have been developed in the statistical pattern recognition field and an introduction to many of them may be found in [26] these include kernel methods, Nearest neighbour methods, Fisher discriminants and so forth. The Bayesian basis of pattern recognition and Fisher's approach are briefly described here.

5.3.1 Bayesian Classification

Using the notation of the introduction begin by assuming the probability that an object comes from class ω_i is a known $P(\omega_i)$. As this is an overall probability known before an observation vector $\mathbf{x}$ has been observed it is a *Prior* probability. Once an observation is made and an observation vector $\mathbf{x}$ is known we can compare the probabilities of belonging to each class for an observation $\mathbf{x}$ and classify according to whichever is larger.

$$P(\omega_k|\mathbf{x}) > P(\omega_j|\mathbf{x}) \Rightarrow \mathbf{x} \in \Omega_k$$

for all $j \neq k$, where Ω_k is the set of objects in the k th class.
This rule is known as *Bayes' minimum error rule.* The $P(\omega_j|\mathbf{x})$ are known as the *Posteriori probabilities* . Sadly these are not normally known and must be estimated. This estimation can be done by making use of samples of known classification as is pursued later in this thesis. In many cases however, Bayes' theorem is applied to give:

$$P(\omega_i|\mathbf{x}) = \frac{P(\mathbf{x}|\omega_i)P(\omega_i)}{P(\mathbf{x})}$$

Yielding:

$$P(\mathbf{x}|\omega_k)P(\omega_k) > P(\mathbf{x}|\omega_j)P(\omega_j) \Rightarrow \mathbf{x} \in \Omega_k$$

For all $j \neq k$. As before however $P(\mathbf{x} \mid \omega_i)$ are not often known.
It is often the case however that an incorrect classification may be of varying importance as a function of class. For example, in medical diagnosis it is often a minor problem in classifying a healthy patient as unwell but it is very dangerous to classify an unwell patient as well.

To build in this factor a cost function may be defined C_{ij} which gives the cost of misclassifying an object from class Ω_i as from class Ω_j. If $\mathbf{x} \in \Omega_i$ the expected cost is:

$$r_i = \sum_{j=1}^{N} C_{ij} \int_{\Omega_j} P(\mathbf{x} \mid \omega_i) d\mathbf{x}$$

The overall expected cost or risk is thus:

$$r = \sum_i r_i P(\omega_i) = \sum_i \sum_j \int_{\Omega_j} C_{ij} P(\omega_i) P(\mathbf{x} \mid \omega_i) d\mathbf{x}$$

This is minimised by defining Ω_k such that $\mathbf{x} \in \Omega_k$ whenever:

$$\left(\sum_i C_{ik} P(\omega_i) P(\mathbf{x} \mid \omega_i) \right) < \left(\sum_i C_{ij} P(\omega_i) P(\mathbf{x} \mid \omega_i) \right)$$

For all $j \neq k$.

This is the Bayes minimum risk decision rule.

5.3.2 Classical Fisher Statistical Pattern Recognition

Fisher in 1936 founded the classical approach to discriminant analysis with Fisher's criterion. The problem is one of finding that direction in the discriminant space along which the two groups to be classified are maximally separated. Fisher defined the separation between the two groups in a particular direction as the distance between the means of the two groups standardised for the within group variance in the specified direction. The importance of this standardisation may be appreciated by consideration of the following example.

Prior to standardisation the separation in the x_1 direction appears greater than that in the x_2 direction (Fig.5:2).

After standardisation it is however clear that the separation in the x_2 direction is greater. In general, standardisations may be performed in any general direction, and the problem is to find the direction $\mathbf{v}$ such that $(\mathbf{v}^t\mathbf{x}_1 - \mathbf{v}^t\mathbf{x}_2)$ is maximised relative to the standard deviation $(\mathbf{v}^t\mathbf{S}\mathbf{v})^{1/2}$ in that direction, where the $\mathbf{x}_i$ is the sample mean for the design set for class ω_i (i=1,2), and $\mathbf{S}$ is the assumed common sample variance-covariance matrix.

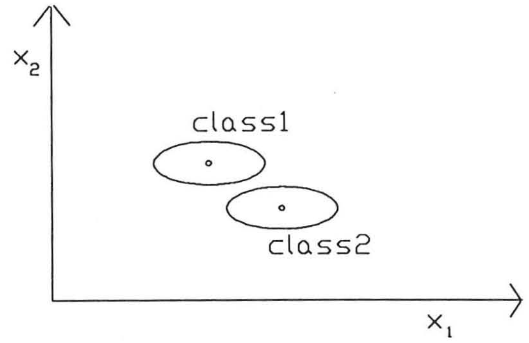


Figure 5.2: Prior to standardisation

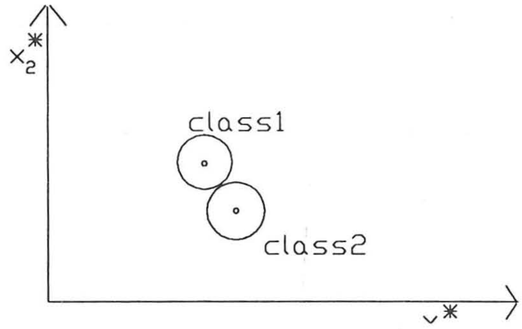


Figure 5.3: After standardisation

For each class if we have a set of observed sample vectors $\alpha_0 \ldots \alpha_n$ then:

$$\mathbf{x_i} = \frac{1}{n}\sum_{p=1}^{n} \alpha_p$$

and:

$$\mathbf{S} = \frac{1}{n-1}\sum_{p=1}^{n}(\alpha_p - \mathbf{x_i})(\alpha_p - \mathbf{x_i})^t$$

To find $\mathbf{v}$ maximise with respect to $\mathbf{v}$:

$$\frac{\mathbf{v}^t\mathbf{x}_1 - \mathbf{v}^t\mathbf{x}_2}{(\mathbf{v}^t\mathbf{S})^{1/2}}$$

Differentiating this with respect to $\mathbf{v}$ and equating it to zero gives:

$$\mathbf{x}_1 - \mathbf{x}_2 = \frac{\mathbf{v}^t(\mathbf{x}_1 - \mathbf{x}_2)\mathbf{S}\mathbf{v}^t}{(\mathbf{v}^t\mathbf{S}\mathbf{v})^{1/2}}$$

As only the direction is required:

$$\mathbf{v} \propto \mathbf{S}^{-1}(\mathbf{x}_1 - \mathbf{x}_2)$$

5.3.3 Limitations of the Classical Approaches

It is difficult to generalise over so many different approaches to the pattern recognition problem. However, in general the computational load required to perform multiclass multivariate recognition is prohibitive with many of the classical methods. The computational form is also often not suitable for parallel implementation, and it is often very difficult to create an adaptive method which will optimise itself easily as more data arrives, without a complete recalculation. Most of the most popular classical techniques are the parameterised distribution techniques which assume that the problem closely resembles a priori probability distributions (such as Gaussian) and although some methods allow the distribution parameters to vary as a function of class the distribution form is usually fixed. This assumption is often invalid as the probability distributions inherent in the problem are often highly anisotropic, class and observation dependent. For example, the variation found in handwritten characters arises not from purely random processes but the different observed vector directions , such as displacements or speeds, may correlate in the way they vary. The non-parametric methods such as nearest neighbour normally become computationally very heavy as they may require multidimensional search operations. These problems have led to the search for computationally lighter, adaptive, parallel methods which require fewer a priori assumptions. Connectionist models avoid these limitations as they are non-parametric and make weaker assumptions about the shapes of the underlying distributions than traditional statistical pattern recognisers.

Chapter 6

Connectionist Models and Neural Networks

6.1 Introduction

6.1.1 Limitations of the Conventional Syntactical Approaches

The modern computer has proved itself to be well suited to solving many problems and provides a powerful tool for aiding the solution of many other problems. There has, however, emerged a type of problem for which standard 'syntactic' or programming methods do not perform well. These problems generally have many input variables and many complex rules relating their outputs to their inputs. The rules that arise in many real world problems, such as visual object recognition under practical conditions, are in general too complex for a suitable set of rules to be worked out explicitly. Consequently, in general, syntactic methods of sufficient complexity cannot practically be constructed to solve these problems.

6.1.2 The Adaptive Approach.

The adaptive neural network can address this problem in that it does not require explicit statement of the rules of the problem, as it will implicitly learn them. Such networks are composed of many simple processing units linked together to create a system capable of performing complex tasks. It is also capable of learning statistical or probabilistic rules which may be fuzzy in nature.

6.2 Artificial Neural Networks

For a long time there has been an interest in reproducing in a machine the abilities of the mammalian brain. Attempts to develop detailed mathematical models began in the 1940s with work by McCulloch [44], and Rosenblatt[45]. In the 1960s Widrow's work [46] with linear perceptrons created much excitement which, as the limitations of the linear approach became apparent, was to fade again until the more recent work by Hopfield [47], Rumelhart and McClelland [48], and the sudden growth of interest that ensued.

The definition of what precisely constitutes an artificial Neural Network, or perhaps more scientifically, a connectionist model, is vague and to attempt to review all the methods under this general heading would be excessive. In this thesis the work relates to a particular class of connectionist model.

Networks may be divided into two classes:

1. Binary: These networks normally take binary input data and produce binary output data. However, internally continuous arithmetic may be used. An example of such a network is the Hopfield Net.

2. Continuous: These networks take continuous (or multilevel) input data and may produce continuous or binary output.

Each of these two classes may again be divided according to the following as a function of its learning abilities.

1. Supervised: The network is given a series of correct output values from some other problem defining system and may use these in conjunction with the input data to adapt itself to improve its performance.

2. Unsupervised: The network is not given correct outputs but is required to extract relationships in the data to allow it to form classification groupings which may in some manner classify 'like' patterns together.

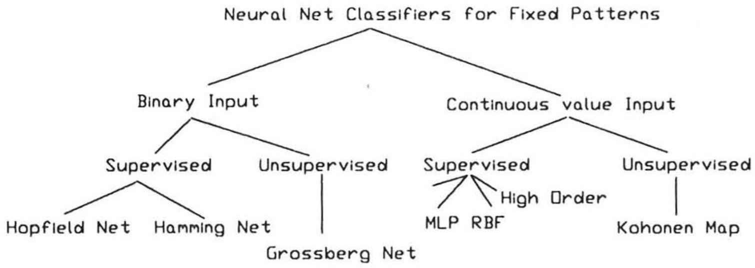


Figure 6.1: Classes of Neural Networks.

This thesis is only concerned with continuous input signal networks. This is the most general signal form and that which is found in most practical problems. The discussion will be constrained to deal only with supervised networks as these most closely resemble adaptive signal processing systems. It is also clear that currently the most general problems being addressed by Neural Networks are usually of the supervised continuous signal type.

6.3 ADALINE

The adaptive Linear Neuron or ADALINE is a simple linear combiner with an LMS feedback used to adjust its weights. The output is usually passed through a threshold device to produce a binary output. The linear nature of the decision region (line in the signal space which marks the transition of the outputs from classification -1 to classification +1) is easily shown by considering a 3 coefficient ADALINE [24].

For a threshold of zero:

$$y = x_1 w_1 + x_2 w_2 + w_0$$

This gives:

$$x_2 = -\frac{w_0}{w_2} - \frac{w_1}{w_2} x_1$$

Thus adaline can only differentiate signals into classes which are linearly separable.

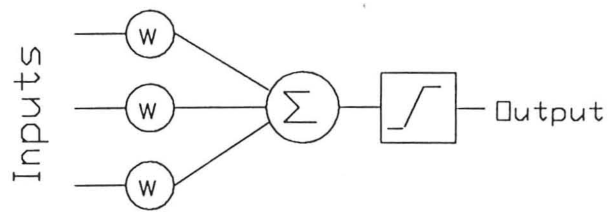


Figure 6.2: Hidden Layer Back propagation Neuron.

6.4 Hidden Layer Back-propagation

If the decision boundaries of systems composed of linear perceptrons are considered it is clear that the decision region is a line (in 2 Dimensions) or a hyper-plane (in N dimensions). This restricts the system from forming suitably flexible decision regions.

To avoid this restriction to linear systems it is necessary to introduce some non-linearity into the transfer function. This may be done by adding a non-linearity to the basic perceptron unit. Obviously there are many non-linearities which could be chosen. One in particular which has gained favour due to its similarity to some transfer functions measured in biological neurons, is the sigmoid, or hard limiter.

The hidden layer back propagation network is composed of neurons made up from a weighted linear combiner followed by a sigmoidal non-linearity. This non-linearity is normally based on a linear region around the origin. There is not a great deal of support from a signal processing point of view for this particular non-linearity, it however has become widely used as it bears a similarity to the transfer function observed in biological neurons. In fact even this justification has been questioned as the sigmoidal response is found in motor control neurons rather than information processing neurons.

The network is made up from sheets of such neurons connected in layers. The name 'hidden layer' comes from the fact that there may be a series of layers 'hidden' between the output and input layers.

The weights of the network are adapted by an algorithm which is very similar to the LMS algorithm, known as back-propagation.

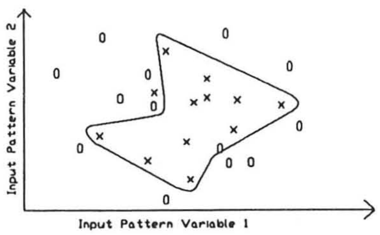


Figure 6.3: Decision Contour for Back propagation Multilayer Perceptron Network.

Consider the case of a sigmoidal non-linearity:

$$f(\alpha) = \frac{1}{1 + e^{-(\alpha - \theta)}}$$

Consider an input vector **x** and a desired output vector **d**.

From the output layer to the input layer adjust the weights by:

$$w_{ij}(t + 1) = w_{ij}(t) + \eta \delta_j x_j^*$$

$w_{ij}(t)$ is the weight from hidden node i or from an input node j at time t. x_j^* is either the output of node i or is an input, η is a gain term, and δ_j is an error term for node j. If node j is an output node then:

$$\delta_j = y_j(1 - y_j)(d_j - y_j)$$

where d_j is the desired output of node j and y_j is the actual output.

If node j is an internal hidden node then:

$$\delta_j = x_j^*(1 - x_j^*) \sum_k \delta_k w_{ij}$$

where k is over all nodes above node j. Internal nodes are adapted similarly by assuming that they are connection weights on links from auxiliary constant-value inputs.

The class regions are bounded by line segments the rounding of the nodes of which is a function of the hardness of the sigmoidal non-linearity. The number of line segments is related to the number of nodes.

6.4.1 Advantages and disadvantages of HLBP

Although perhaps currently the most popular form of connectionist model it is indeed difficult to find advantages for it over other networks. It is slow to adapt, can become stuck in local minima, has very limited mathematical justification, and is difficult to analyse. The analogy of its form with biological neurons is also of question. There tend to be a number of ad hoc parameters which must be optimised experimentally to allow the adaption algorithm to function.

6.5 Radial Basis Functions

These networks are based on the method of radial basis functions, a technique for interpolating in high dimensional spaces [16].

The RBF networks are of interest in that they have arisen from a careful and mathematically well founded assessment of the required tasks of a connectionist model.

It is clear that the classification problem may be viewed as a multidimensional interpolation problem. RBF networks are consequently a good method to produce a network capable of classifying input vectors into a series of classes. Define a set of m generally non-linear basis functions:

$$\phi(\| \mathbf{x} - \mathbf{y} \|)$$

Interpolating functions may be constructed from these basis functions with the form:

$$s(\mathbf{x}) = \sum_{j=1}^{m} \lambda_j \phi(\| \mathbf{x} - \mathbf{y}_\mathbf{j} \|)$$

These basis functions may be centred on any points $\mathbf{y}_\mathbf{i}$ including those not members of the training data set.

6.5.1 Advantages and Disadvantages of the RBF Approach

The RBF approach produces networks which, by the very nature of its derivation, tend to have good generalisation and interpolation properties. The most basic form of the RBF network is also unimodal and consequently avoids all of the problems encountered with HLBP networks due to local minima, and can consequently show quite fast adaption rates.

However, the necessity of choosing a priori the basis functions leads to a limitation of the performance of the method as it is assuming a priori the form of the pattern distributions. This assumption may not only be incorrect in general but also it is likely that the form of pattern clusters and the statistical properties of the patterns may be different for different regions of the recognition space. This can lead to highly variable results depending on the form of RBF and the RBF centre positions for a given problem.

The RBFs being localised may be expected to work moderately well for well clustered data.

Attempts to address the RBF preselection problems by some form of adaption of the RBF position and form have been attempted, but these suffer from the same problems of multimodality and slow convergence found with the HLBP [48].

From a theoretical viewpoint it is interesting to note that RBF networks are subsets of general higher order networks in which polynomial terms have related coefficients. This can be shown by expanding each RBF as a power series.

RBFs are however a well conceived approach and may be of great use for some types of problems in which there is good a priori information.

6.6 Group Method of Data Handling.

Ivakhnenko [36] in the sixties was among the first to suggest the possibility of using a polynomial type non-linearity in an adaptive system to solve generalised problems.

This approach has sadly been largely neglected by the neural network community. It is in fact a reasonably well conceived method with good performance.

A multilayer polynomial system was developed and this is the method normally referred to as Group Method of Data handling (GMDH). This method attempts to reduce the number of polynomial terms by using the outputs of smaller polynomials as the inputs to other polynomials in a layered structure. Whilst the approach is much better grounded in theory than other methods the multilayer approach leads to multimodality and problems in adaption if a normal 'on the job adaption' is to be done. The method is more suited to block processing on a predetermined training set. The ad hoc nature of some of the elimination rules also requires multiple runs and a priori knowledge to allow good performance.

Experience in signal processing shows that it is better to avoid the considerable problems encountered in multimodal system adaption by using a different method to create and select the useful polynomial terms.

Most of the work in this area has been done in prediction problems with surprisingly little work in pattern recognition and consequently very little has been done in optimisation of the method for this problem.

There is now a series of closely related approaches based in general on the following.

Firstly a set of first layer polynomials are calculated directly from the input data $x_1 \ldots x_m$ by taking pairs of input points giving $l = 1 \ldots m(m-1)/2$ new values:

$$y_l = A + Bx_i + Cx_j + Dx_i^2 + Ex_j^2 + Fx_ix_j$$

The coefficients of these equations are each optimised so that the error $\epsilon = d - y_l$ taken over the training set is minimised in the mean square error sense. Polynomials whose error after optimisation is highest are eliminated from the algorithm.

These y_l are now used as the inputs to another set of polynomials and so on.

This is continued until the minimum error from a polynomial in stage n is less than that for stage n+1.

The normalised RMS error over the training set is called the regularity criterion and is defined:

$$r_l^2 = \frac{\sum_{i=1}^{p}(d_i - y_{il})^2}{\sum_{i=1}^{p} y_i^2}$$

where there are p training set values d_i.

The multimodal nature of the surface leads to the need to apply a very slow adaption algorithm which is usually some form of random search with a modification. Such approaches called 'guided random search' are not dissimilar to simulated annealing or genetic algorithms [36] and as such are very slow to converge. Another problem with GMDH is that although two polynomials may each independently fail to lower output errors their cross products may lower the output error. Thus the independence of the terms selection procedure is questionable.

6.7 Higher Order Networks.

These are simple polynomial based networks. During the time of the work covered in this thesis work has been done in this area [24] mainly motivated by the ease of implementation of multiplication-like terms in optical methods.

Most of the higher order approaches used in neural networks are very simple and no attempt is made to optimise their required computational load as in GMDH. They have often been based on binary networks and only applied to small problems.

6.8 The New Approach

The system developed in this thesis is a higher order system which is developed and refined using a signal processing approach and is aimed at the pattern recognition problem. The new approach is related to GMDH in that it is also attempting to find a polynomial network . However, it is designed to be adaptive in operation both for training data and the real data and is optimised in performance by methods which are consistent with unimodal operation. It is also optimised with respect to the pattern recognition problem.

6.9 Other Continuous Supervised Connectionist Model Forms

There is now a series of proposed modifications of current networks, and some new types of network. These can be found referenced in [24]. Many of these other networks can be expressed as modifications of HLBP or radial basis functions. Some other methods of interest are Kohonen Self Organising Nets (unsupervised), and adaptive resonance theory (Unsupervised) [24].

Chapter 7

Generalised Adaptive Filters

7.1 The Signal Processing Approach

In the early days of neural networks there was a tendency to over-extrapolate known physiological information to provide networks. There was also a tendency to produce networks by empirical methods. This led to systems which could not easily be analysed or optimised. Recently the field of signal processing has advanced sufficiently to allow the mathematical derivation of a network form by using a well defined signal processing approach.

Firstly, it is necessary to consider a basic problem. A standard 'bench-mark' for neural networks is the exclusive-OR or parity problem. In this problem a series of binary inputs are input to the network and it is required to return the parity function value of the inputs.

We can consider this problem from a signal space approach. Consider a vector space created by plotting each of the input variables along a set of orthogonal axes. For a 2 input variable problem this space is a plane. For the binary exclusive-OR problem the input values (or 'input vectors') are (1,1) (0,0) (1,0) (0,1), the first two being in class one and the second in class two.

It can be seen from Fig.7.1 that a linear discriminator cannot be used to differentiate between the classes in the EX-OR problem. We can however transform the problem into an extended vector space in which the problem may be solved using a linear discriminator. We increase the dimensionality of the decision space by adding a dimension x_1x_2 . A linear plane may now be used to separate the classes.

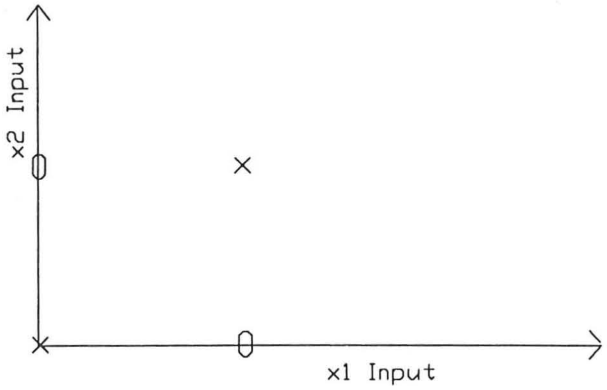


Figure 7.1: EXOR Problem.

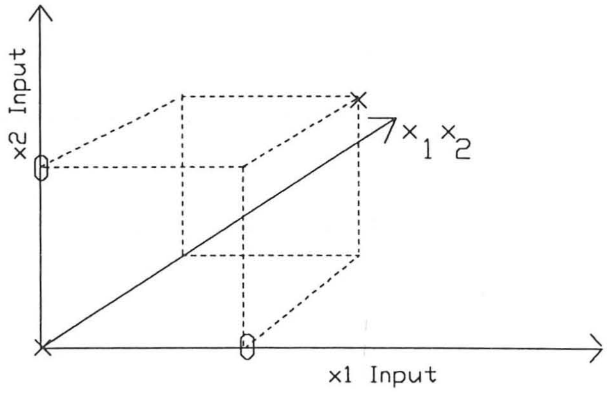


Figure 7.2: Extended space EXOR Problem.

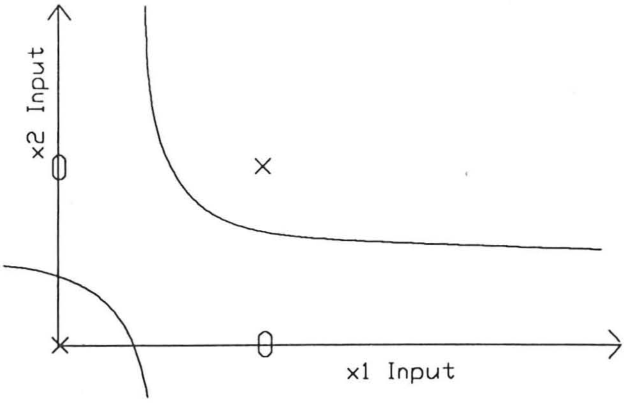


Figure 7.3: Extended space EXOR Problem decision contour .

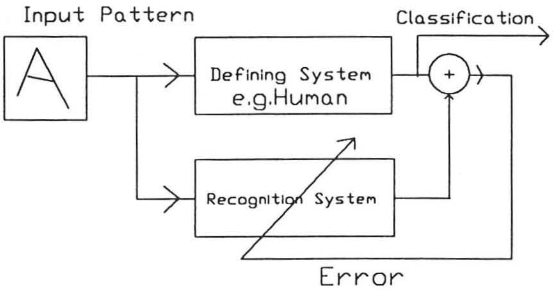


Figure 7.4: Pattern Recognition as an adaptive Modelling Problem.

If we collapse the decision plane of Fig 7.2 onto the 2D plane as in Fig 7.3 it can be seen that the decision contour is non-linear.

So by using a linear discriminator in an extended vector space it is possible to perform classification tasks requiring non-linear discriminators.

7.1.1 Non-linear System Modelling Approach

The pattern recognition problem may be thought of as a system modelling problem in which the adaptive pattern recognition is attempting to model the classification system; for example, a human defining the relationship between a set of character font images and a set of collating sequence numbers. The block diagram showing the recognition problem cast as a system modelling problem is shown in Fig 7.4.

It is clear that the relationship between the inputs and outputs of such a system is in general a non-linear one, thus the recognition system must be capable of generating a sufficiently close approximation to the non-linear relationship defining the system.

7.1.2 The Volterra Functional Analysis

In signal processing such non-linear systems may be modelled by use of the Volterra series. The form of the Volterra series was first studied by Vito Volterra, but Norbert Wiener was the first to apply it to non-linear systems theory. He used it to model the input-output relationship of non-linear systems. The continuous Volterra series may be written:

$$y(t) = \mathbf{H_0} + \mathbf{H_1}[x(t)] + \mathbf{H_2}[x(t)] + \ldots + \mathbf{H_n}[x(t)] + \ldots$$

where $y(t)$ is the output of a system and $x(t)$ is the input and in which:

$$\mathbf{H_n}[x(t)] =$$

$$\int_{-\infty}^{+\infty} \ldots \int_{-\infty}^{+\infty} h_n(\tau_1, \ldots, \tau_n) x(t - \tau_1) \ldots x(t - \tau_n) d\tau_1 \ldots d\tau_n$$

where $h_n(\tau_1, \ldots, \tau_n)$ is the Volterra kernel.

Stability of Volterra Functional

The stability analysis of a p_{th} order Volterra functional has not yet been done. However a sufficient but not necessary condition for the Volterra functional to be bounded-input, bounded-output stable (BIBO) is by inspection:

$$\int_{-\infty}^{+\infty} \ldots \int_{-\infty}^{+\infty} \mid h_p(\tau_1, \ldots, \tau_p) \mid d\tau_1 \ldots d\tau_p \langle \infty$$

where h_p is the pth order impulse response. [13] gives an example of a stable system not meeting the above.

Discrete Volterra Series

A discrete form of the series was developed.

In general this will need to be a heterogenous Volterra expansion in order to model systems composed of multiple orders of non-linearity. The heterogenous discrete Volterra series may be written as:

$$\sum_{i=1,N} \mathbf{H_n}$$

where $\mathbf{H_n}$ are the Volterra Kernels, and w are coefficients which define the model as follows :-

$$\mathbf{H_o} = \mathbf{wo} \text{ Zeroth Order term or `D.C.' term}$$

$$\mathbf{H_1} = \sum_{i} \mathrm{w_i x_i} \text{ First Order Linear term}$$

$$\mathbf{H_2} = \sum_{i}\sum_{j} \mathrm{w_{i,j} x_i x_j} \text{ Second order Quadratic Term}$$

$$\mathbf{H_3} = \sum_{i}\sum_{j}\sum_{k} \mathrm{w_{i,j,k} x_i x_j x_k} \text{ ....and so on.}$$

7.1.3 Adaptive Volterra Expansions

Normally the coefficients are fixed by use of analytical methods: however, they may be fixed adaptively. Consider the following system in which the inputs are combined to give the individual Volterra terms. These terms are then multiplied by their respective coefficients and the terms summed. It can be seen that the system can be decomposed into a vector expansion stage followed by a linear Finite Impulse Response (FIR) filter. The adaption of the coefficients may then be achieved by any of the methods used in FIR adaptive filter theory. The analytical basis of this approach allows us to calculate explicitly the form of the error surface. The Volterra Series is found to converge for most practical non-linearities, although large input levels or very high order non-linearities should be avoided. A very closely related expansion may be obtained by using the Gabor-Kolmogorov polynomial:

$$y = w_0 + w_1x_1 + w_2x_2 + w_{11}x_1x_1 + w_{12}x_1x_2 +$$

$$w_{22}x_2x_2 + w_{111}x_1x_1x_1 + w_{112}x_1x_1x_2 + \ldots$$

An often convenient notational method for such systems is to write the expansion terms as a series of Kronecker products.

In a system such as that shown the effect of joining an FIR filter to the Volterra expansion may be considered. The system has a set of input values $x_0 \ldots x_n$ which are input to the non-linear Volterra state expander to produce a set of output values $v_0 \ldots v_q$ with n being less than q. These values then form the input to an FIR adaptive filter. Consider the

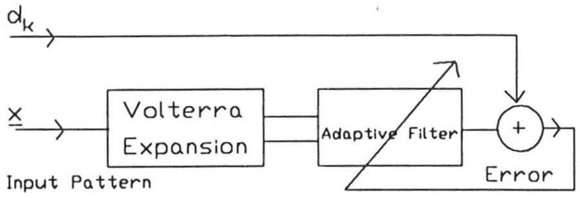


Figure 7.5: Adaptive Volterra System.

output values for the vector state expander. This set of values may be represented by the vector v. The filter coefficients may also be represented by the vector w.

$$\mathbf{w} = \begin{pmatrix} w_o \\ w_1 \\ w_2 \\ \vdots \\ w_n \end{pmatrix} \quad \mathbf{v} = \begin{pmatrix} v_o \\ v_1 \\ v_2 \\ \vdots \\ v_n \end{pmatrix}$$

Using the standard FIR equation we may write the error ϵ as:

$$\epsilon = d - \sum_i v_i w_i$$

in which d is the desired output from the filter, that is in our case the correct classification given the set of inputs to the whole system x_i. In vector form this may be written:

$$\epsilon = d - \mathbf{w}^{\mathbf{t}}\mathbf{v}$$

If we wish to consider the error function for the Least Mean Squares, or LMS, criterion with the mean being considered over the pattern presentations, the equation becomes:

$$E(\epsilon^2) = E[(d - \mathbf{w}^t\ \mathbf{v})^2] = \sigma^2 - 2\mathbf{wp} + \mathbf{wRw}$$

where R is the autocorrelation matrix of the data in the non-linearly extended vector space and similarly p is the crosscorrelation vector between the desired d signal and the non-linearly expanded data. σ is the variance of the desired response.

The objective is to find a set of weights **w** which will minimise the mean square error. The system then will output classifications which differ from the desired classifications with the minimum mean squared error possible for the system. The solution for the weights of the filter is obviously the same as that of a Wiener Filter operating in the extended space and so the methods used to find the Wiener solution explicitly may be applied. A more suitable approach for the recognition problem is to find the weight solution adaptively. This has the advantage of not requiring a priori knowledge of **R**, and does not require the inversion of a matrix.

It is clear from the above that the LMS error is solely a quadratic function of the weight vector. This shows that the performance surface is a hyperparaboloid and consequently always unimodal. This is a highly desirable property as it shows that the system will not exhibit local minima on the error surface. Such local minima cause the adaption performance of networks to be greatly reduced as the adaptive algorithm may get stuck in local minima and consequently not reach the optimal coefficient values. The property of unimodality also confers the ability to adapt much more quickly as there are no valleys in which to dither and the gradient is proportional to the distance from the solution. This last property prevents the adaption being greatly slowed or halted by regions of the error surface with small gradients. A stochastic gradient algorithm may be easily applied by following the derivation of the LMS algorithm. The usual gradient algorithm may be written by differentiating the mean square error with respect to the weights to give an iterative weight update equation.

$$\mathbf{w} = \mathbf{w} - \mu \nabla E(\epsilon^2)$$

where μ is an adaption coefficient which sets the speed of learning . A simple expression for $\nabla E(\epsilon^2)$ may be derived.

$$\frac{\partial E(\epsilon^2)}{\partial \mathbf{w}} = 2\epsilon \frac{\partial \epsilon}{\partial \mathbf{w}} = -2\epsilon \mathbf{x}$$

The analytic basis of the method combined with the property of linearity in the coefficients allows the application of the standard least squares methods. In general for the pattern recognition the problem is degenerate or rank-deficient. A single solution may still be obtained by the application of Singular Value Decomposition (SVD) which yields the minimum norm solution. Such analytical methods may also be used to study the network performance and obtain the eigenvalues for real recognition problems.

Parameter Tuning

There is only one parameter to be chosen in the new network: this is the adaption coefficient. However, this adaption coefficient is that of a standard adaptive filter. Consequently the large body of knowledge pertaining to adaptive filter theory may be immediately applied. Yassa[3] developed an expression for choosing the optimal adaption coefficient. No such method may be applied in the case of current networks as the nature of the error surface is complex and contains local minima. Consequently in current networks the parameters such as the adaption coefficient must in general be set by trial and error.

7.2 Multiple Level Classifications

The output of the network is well suited to multiple class classifications. This may be done by the use of different levels on the same output. For example the desired network outputs may be assigned to the integers. In this case an output within .5 of the desired classification can be considered as belonging to the classification.

If it is important for the network to only classify patterns correctly and the cost of an incorrect classification is high, a 'guard' band may be defined around each classification.

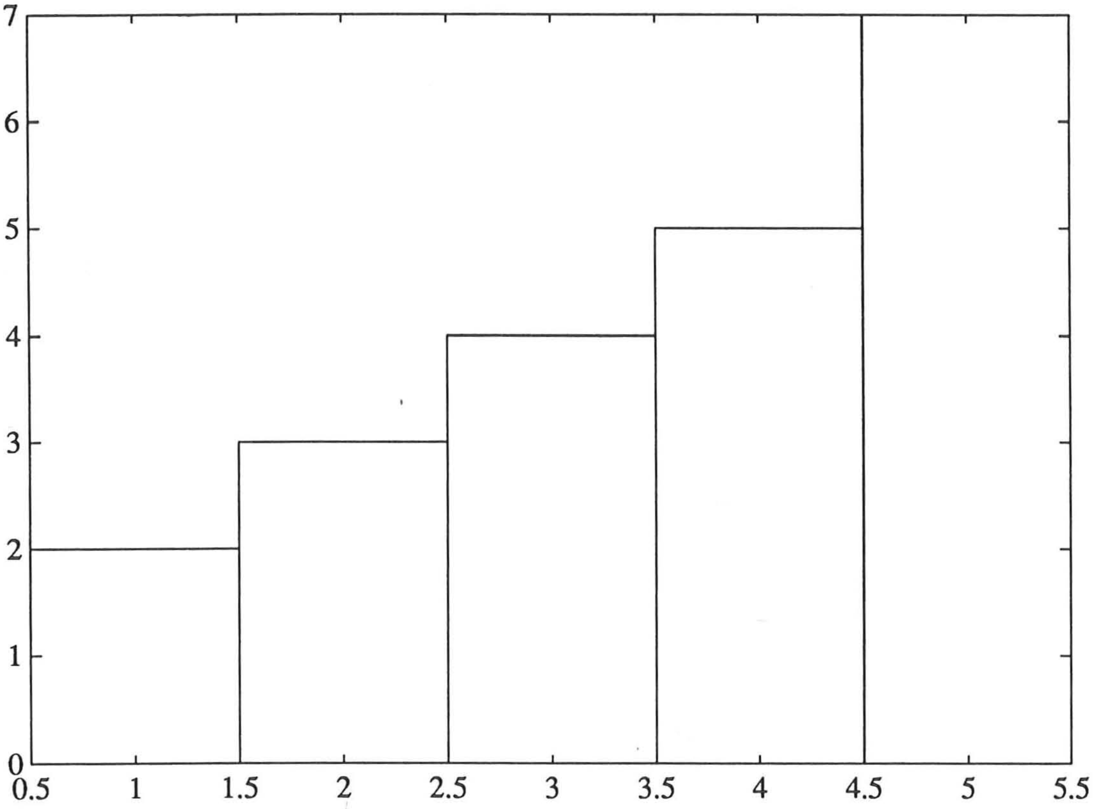


Figure 7.6: Classification against network output : Simple case.

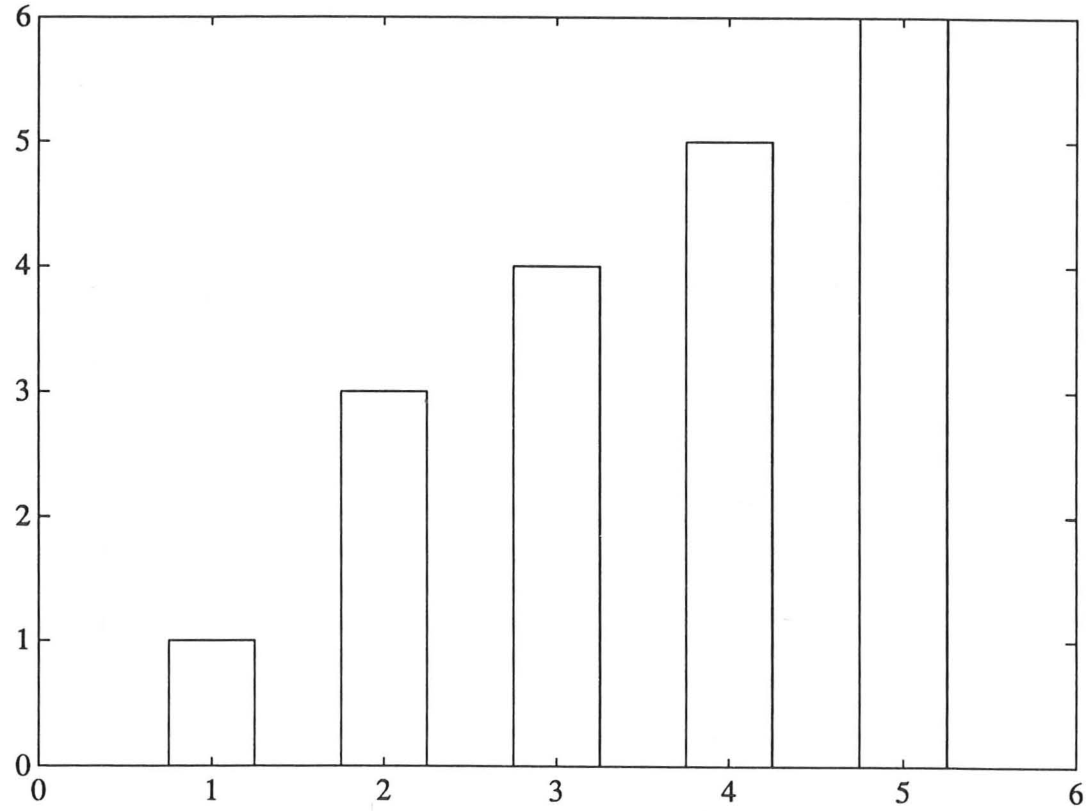


Figure 7.7: Classification against network output : Guard case.

Chapter 8

Network Performance

8.0.1 Generalisation Results

The ability of the network to form general decision contours is tested for small two-variable problems. In fact this is rather a harsh test as in practice if any hyperplane were as densely populated by patterns as in the examples, a further extension to separate the patterns would probably be used. The contours were obtained by forming the extended space correlation matrix over all the patterns and inverting it to solve the standard weight equation. The first class was assigned an index of one and the second an index of zero. The output threshold was set at 0.5.

The peanut cluster has proved a particularly difficult test for some types of network as it requires a larger number of decision line segments than the simple cluster problem.

As there was no noise in this experiment the function was not constrained in regions in which no patterns existed, and consequently used the freedom of these regions to allow itself to fit the patterned regions more accurately.

8.1 Network Comparisons

The new network's performance was compared with that of the current networks. The exclusive-OR problem was used as a test. The adaption times in terms of numbers of iterations were compared for the different network types. The times for the hidden layer back-propagation networks are taken from an American study [14]. These timings are for the best run, which was achieved by optimising the network parameters over a series of runs. If the HLBP network became stuck in a local minimum it was stopped and

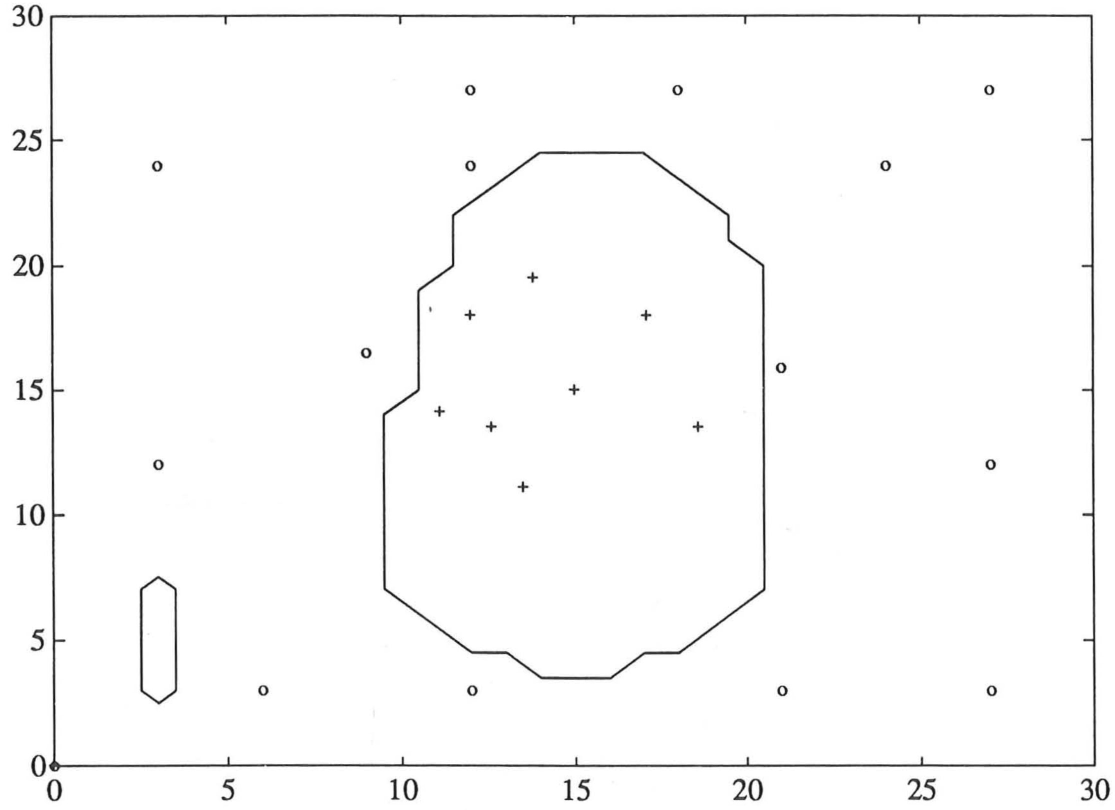


Figure 8.1: Localised Cluster: Patterns and Decision Contour.

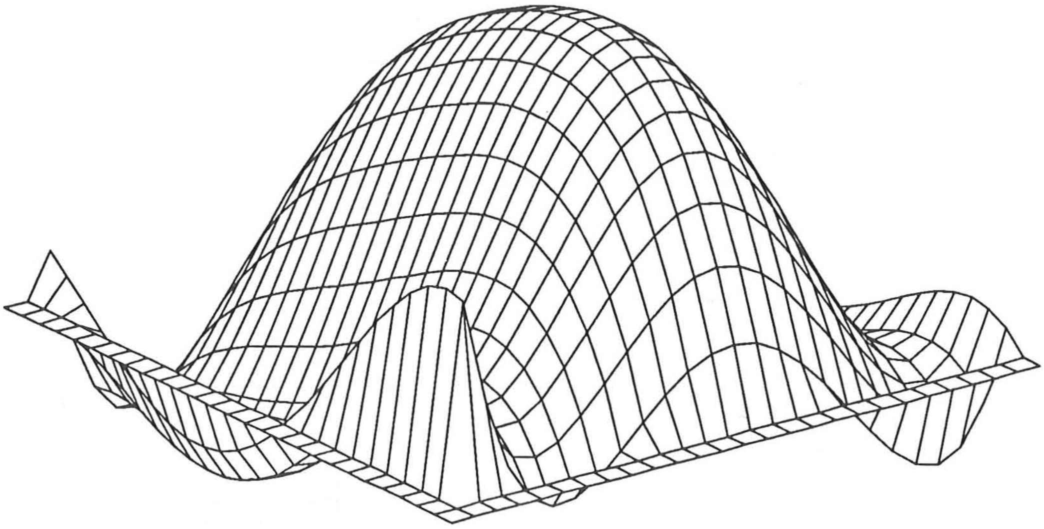


Figure 8.2: Localised Cluster: Discriminant Function Values.

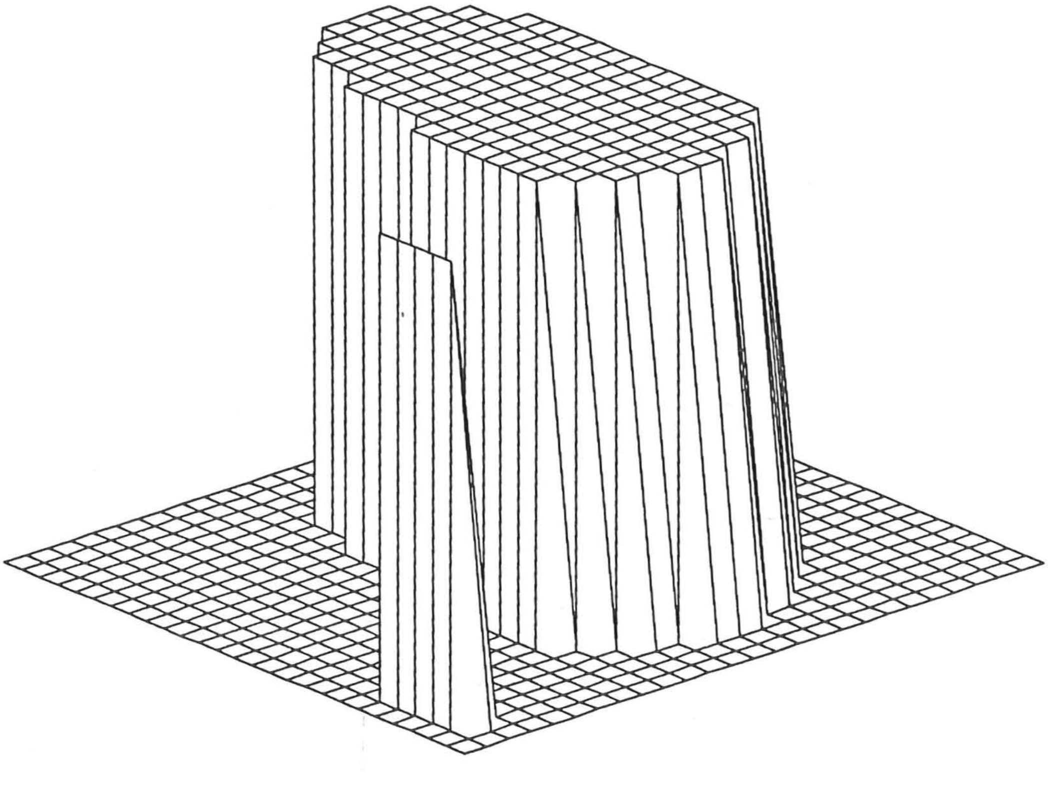


Figure 8.3: Localised Cluster: Thresholded Discriminant Function Values.

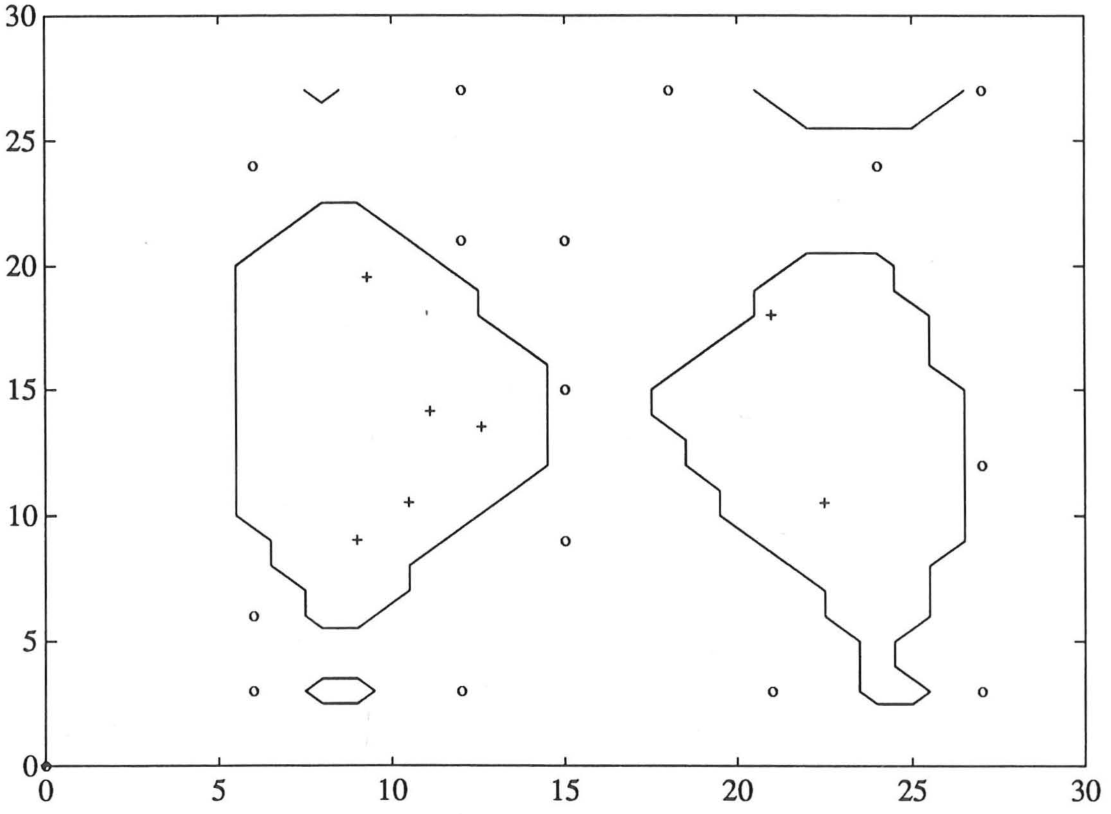


Figure 8.4: Dual Cluster: Patterns and Decision Contour.

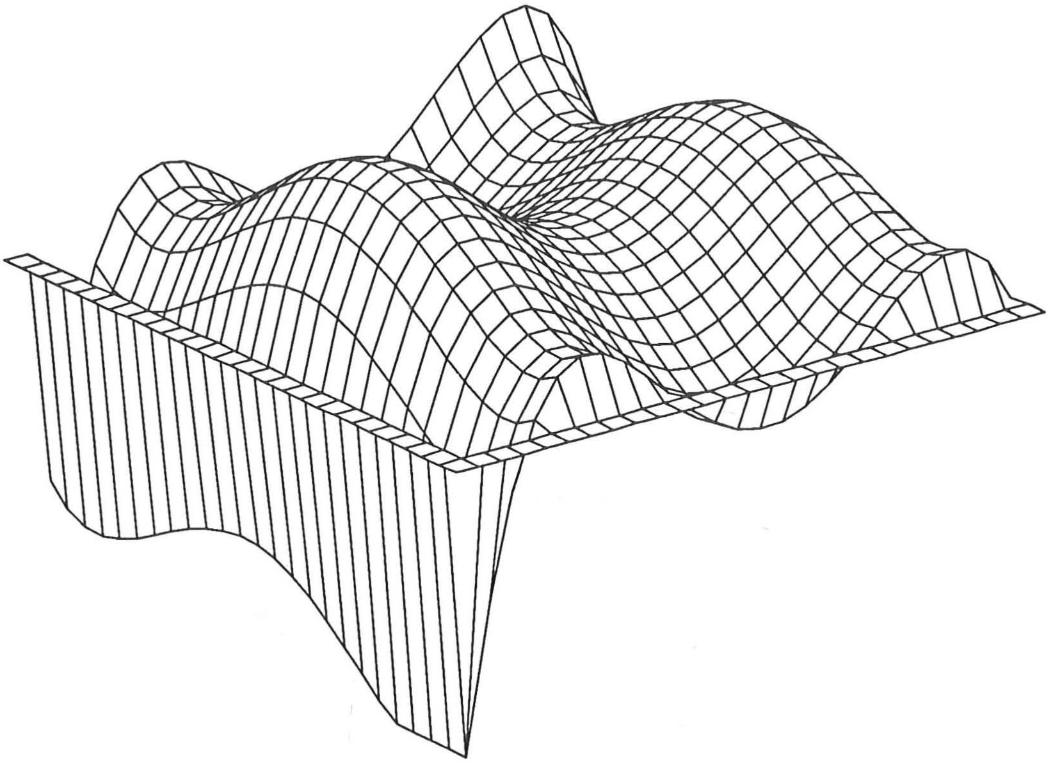


Figure 8.5: Dual Cluster: Discriminant Function Values.

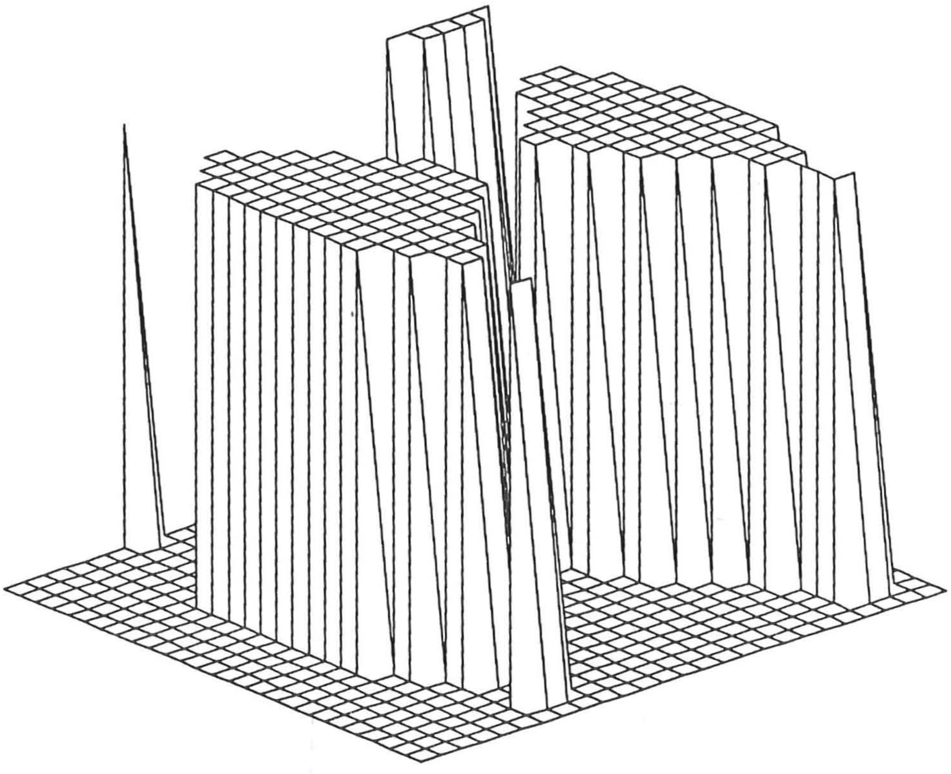


Figure 8.6: Dual Cluster: Thresholded Discriminant Function Values.

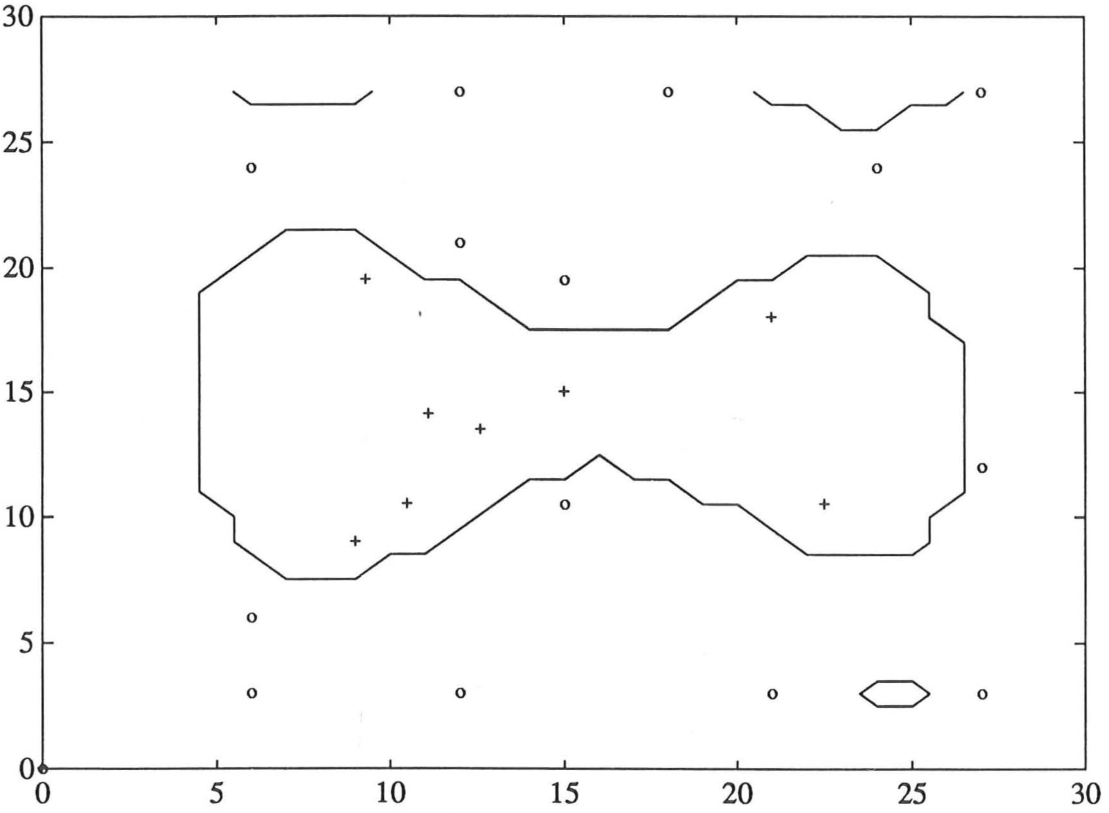


Figure 8.7: Peanut Cluster: Patterns and Decision Contour.

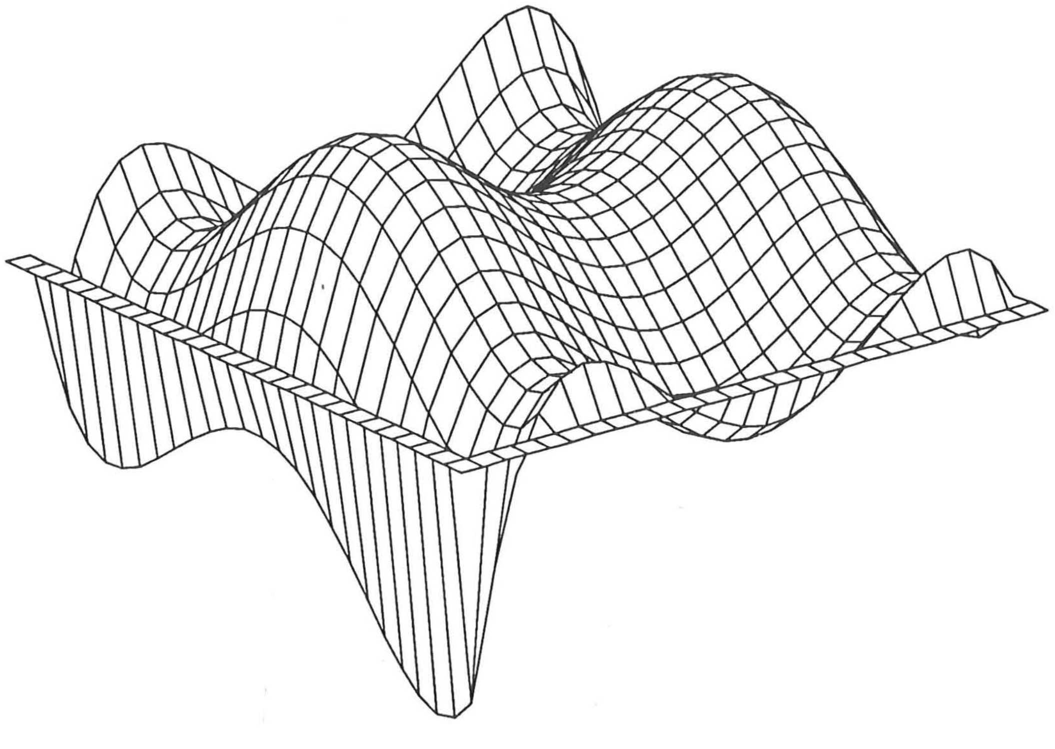


Figure 8.8: Peanut Cluster: Discriminant Function Values.

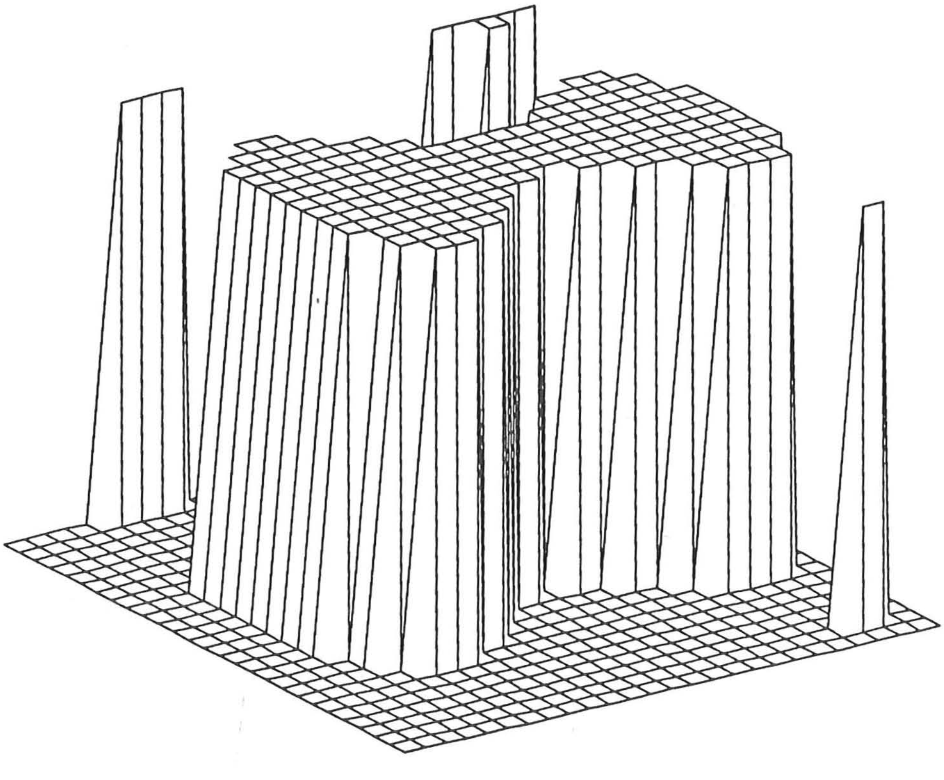


Figure 8.9: Peanut Cluster: Thresholded Discriminant Function Values.

restarted from different initial conditions. In the case of the new network the results are for the first time run as the adaption coefficient could be selected beforehand, and the local minimum problem did not exist. Thus when comparing the figures it is necessary to realise that in terms of total pattern presentation numbers the factor between the number of presentations to each network is in fact considerably greater.

Number of bits	Volterra	HLBP
2	15	95
3	17	265
4	62	1200
5	106	4100
6	292	20000
7	556	100000
8	1287	over 500000

If the figures in the above table are plotted for the two networks it is possible to see that the learning times for the current network rise quickly as a function of the number of input variables, whereas those for the new network rise at a lower rate. The results of the graph imply that the learning times for current networks are likely to be extremely large for complex problems with more than a few input variables.
In fact to see the Volterra results on the same scale a log plot is needed.

8.1.1 Interpolation Performance Comparison.

In this section the interpolative ability of two current network types Hidden-Layer and Radial Basis Function (RBFs)[42] is compared with that of the new network. The network is trained on the exclusive OR class points (0,0), (1,0),(0,1),(1,1) until full convergence. The

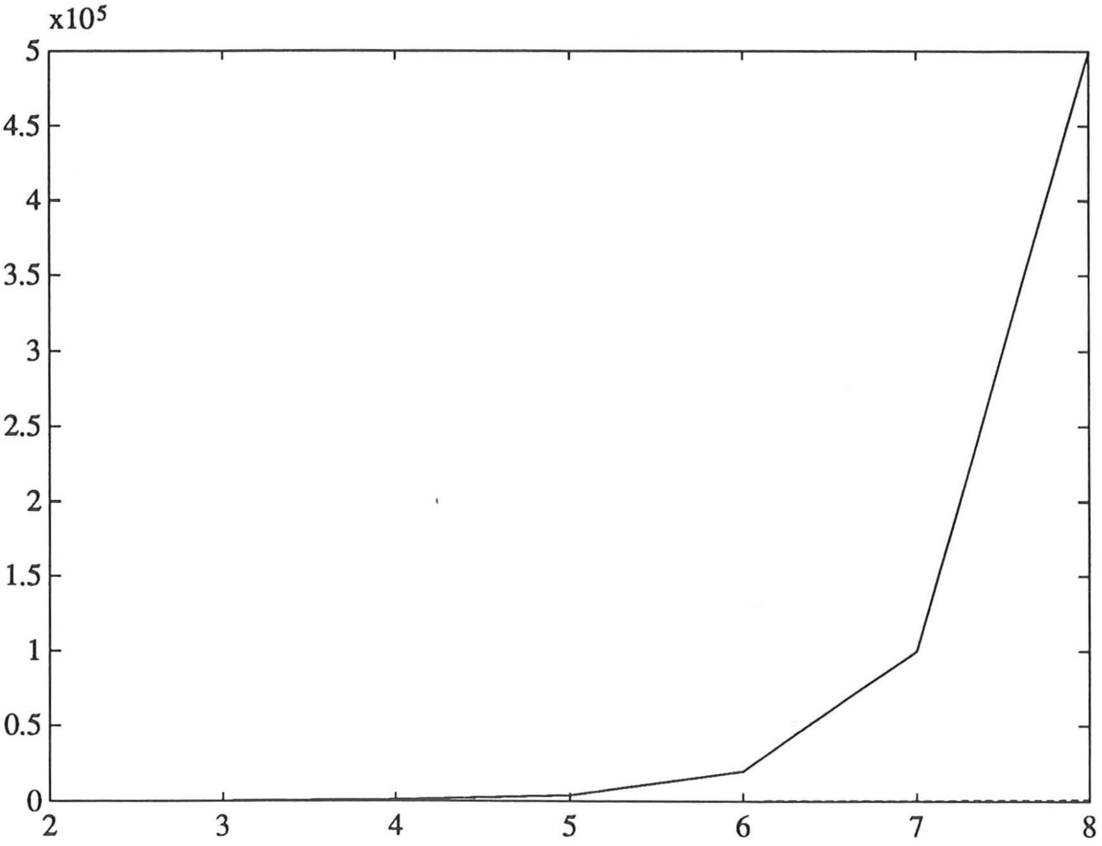


Figure 8.10: Presentations v Number of bits. HLBP solid, Volterra dashed line

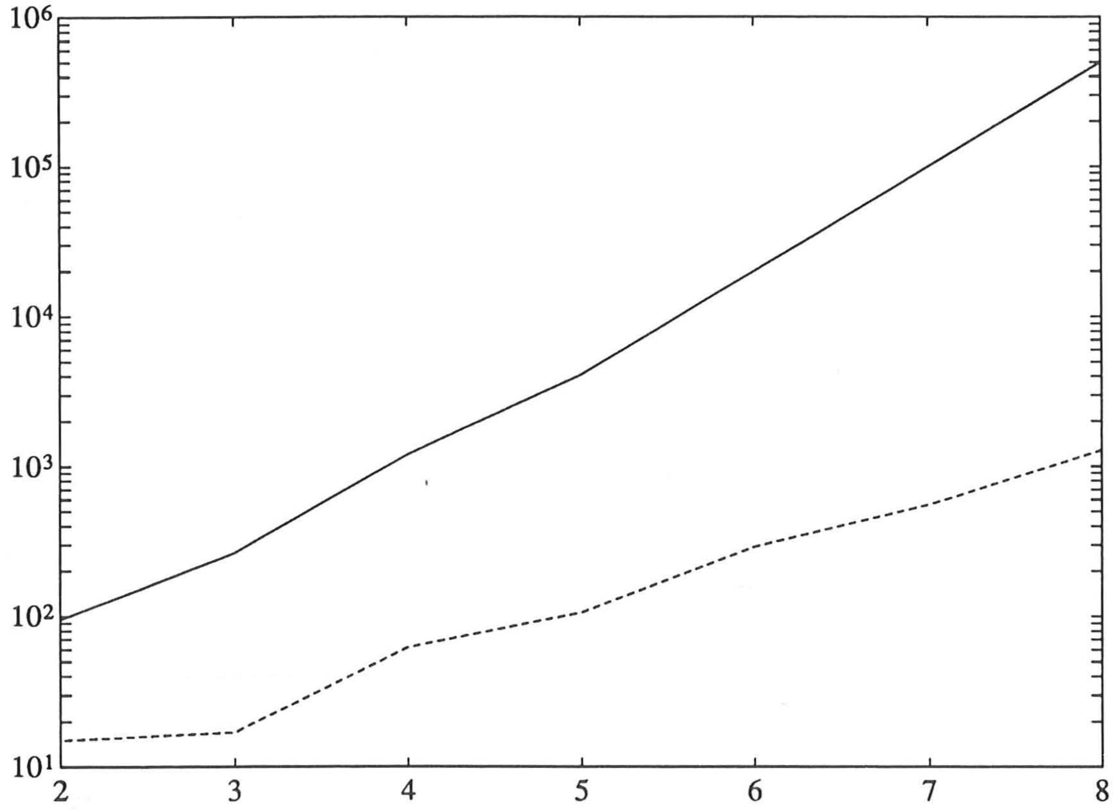
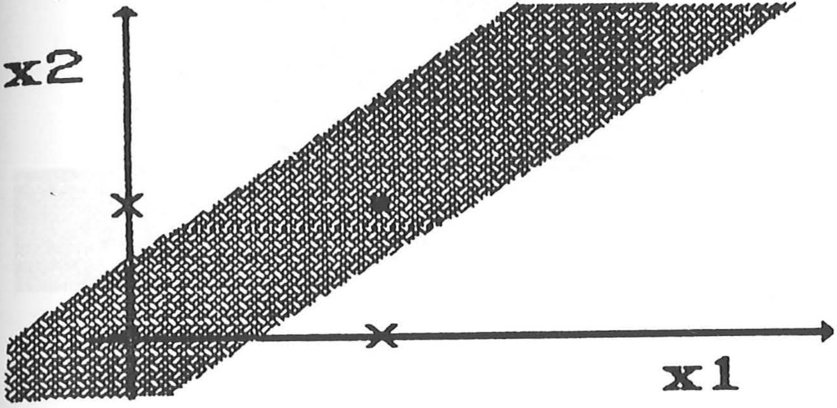
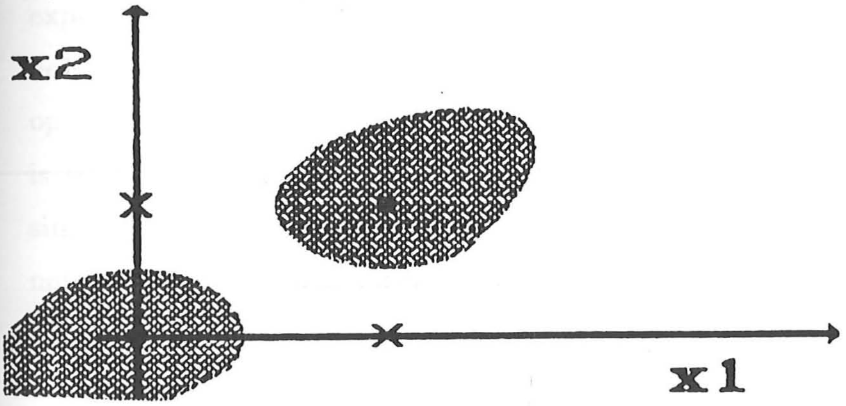
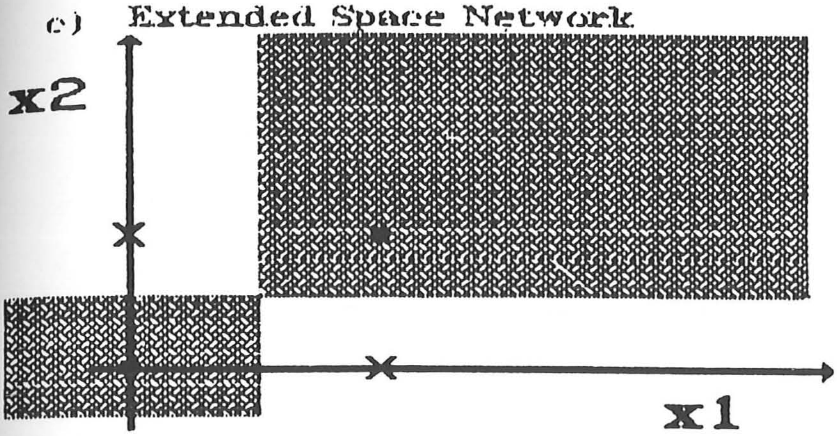


Figure 8.11: LOG Presentations v Number of bits. HLBP solid, Volterra dashed line

adaption is then stopped and the inputs are varied over the full signal space. The class decision is then plotted as a point for class one and no point for class two. Graphs a) and b) were provided by Niranjan[19].


a)Hidden Layer Back-Propogation


b) Radial Basis Function



The hidden layer network displays poor interpolative abilities which is to be expected as the decision region has linear boundaries. The Radial Basis Function network performs much better, producing suitably localised classes. This increase in performance is to be expected due to the relationship between RBFs and multidimensional interpolation. The new network converges to the best solution of the three, which is extremely near to the optimal solution, given no information other than at the class points. In fact, if the above is repeated for the network before it has fully converged, it displays results which are similar to those of RBF networks, that is the decision regions for a partially converged network is of a similar form to that of the fully converged RBF network. However the new network is not constrained to a set of *a priori* 'RBFs' and so can converge further. In fact this behaviour can be seen by following the decision contours as the network adapts.

8.1.2 Multiclass Performance

In practice it has not proved possible, in general, to use current networks for classification with more than 26 classes. In general, multiple class problems are addressed by using a series of networks. The flexibility of the decision surfaces of the new network, however, allows a single network to perform multiclass problems. One such problem is that of character recognition. The network was presented with an 8 by 8 pixel image of a character

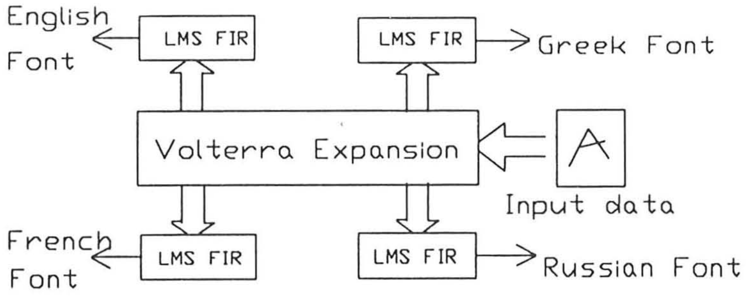


Figure 8.12: Multiple Mutliclass Problem.

and required to output the character's position in the collating sequence. The network was capable of learning to recognise all 80 characters in the font. A second order Volterra extension was used with all terms present. The coefficients were set by the LMS algorithm. It is also possible to increase the number of classes that a system can recognise by using the same vector expansion 'hardware' with multiple adaptive filter sections connected to it.

8.2 Determination of Network Capacities and Degeneracy

The capabilities of the network to differentiate inputs into a number of classes can depend on the degree of non-linearity of the relationships between the input patterns and output indices. A measurement of the 'fullness' of the network can be determined by considering the rank of the extended-space correlation network **R**. A rank lower than the number of expanded terms shows that the solution to the problem is degenerate and consequently the network should be able to perform well in that it can produce a number of solutions which will minimise the mean square error. However, Volterra network generalisation properties are found to be reduced for rank deficient problems as the polynomial form of the network can tend to over-model parts of the extended space. This can be avoided by application of the network size optimisers described later in this thesis. The extended-space correlation matrix is given by:

$$\mathbf{R}_{ex} = \mathbf{v}\mathbf{v}^t$$

where $\mathbf{v}$ is the extended state vector.

The rank of the matrix may be obtained by various methods, although Singular Value Decomposition can give good results when working with rank deficient systems.

Once the rank of the extended space correlation matrix is full this implies that there is now a unique solution. The ability of the network correctly to classify patterns now also depends on the regions over which the outputs are thresholded to within the correct class. Thus, for example, a correct output may be obtained if the network desired output is 1.00 but the network actually outputs 1.07, if the output thresholder operates between .5 and 1.5. This question of output region definitions is also relevant to the cost of incorrect classifications and more work may be done in future to relate the thresholded mean square error criterion to the cost and probabilities of incorrect classifications and hence set optimal output ranges.

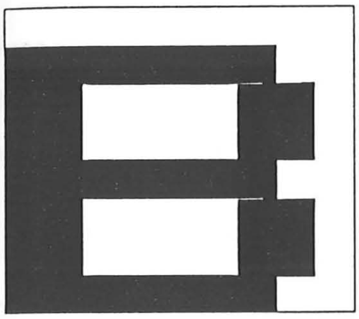


Figure 8.13: Example character.

8.3 Visual Recognition Results

8.3.1 Direct Recognition

The network was first applied to recognising binary font images of size 8 by 8 pixels. The sixty four pixels were input to the network and an error signal was generated by subtracting the network output from the value of the font member in its collating sequence[1]. The adaption coefficient was approximately set by using an estimate of the average power of the input image.

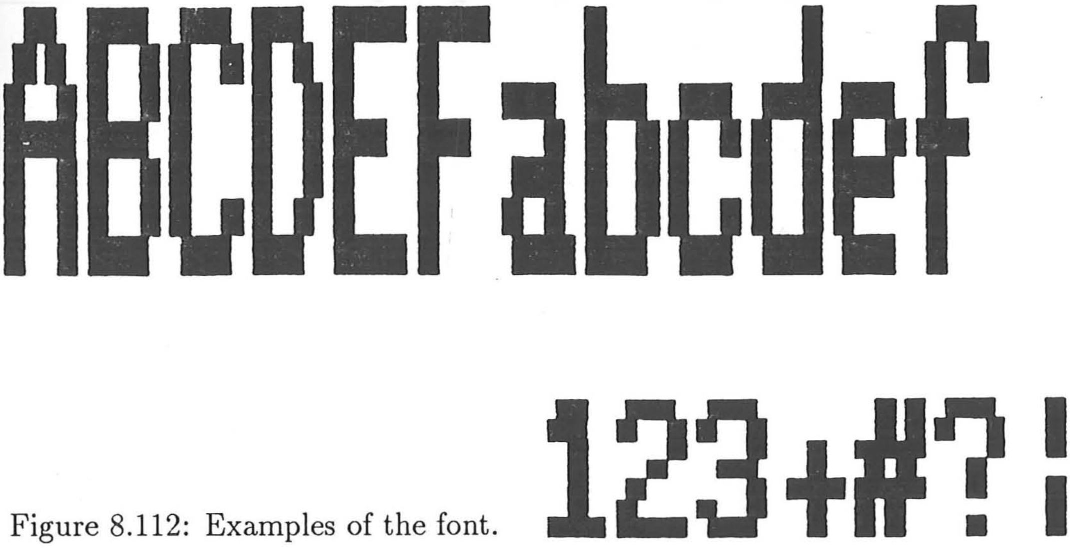


Figure 8.112: Examples of the font.

[1] The term 'collating sequence' means the index to the character in an ordered font

8.4 Invariant Recognition

A preprocessing section was used to produce patterns which were invariant to shift, scale or rotation. The recognition process was repeated using real camera images of simple handwritten capital letters.

8.4.1 Preprocessor

One of a series of capital letters was drawn with a marker pen onto a card. The image of the card was digitised by the use of a camera and framestore. The first stage performed by the preprocessor was an adaptive level binarisation; this was followed by a calculation of the centroid of the resulting image. The next stage was to use a simple edge finding algorithm to reduce the image to an outline image of the character. A line following algorithm was used to find sixty-four points in order around the outline. This algorithm was started from the leftmost point on the letter, and this process was repeated three times to take account of inner loops in the letters, for example, the letter B.

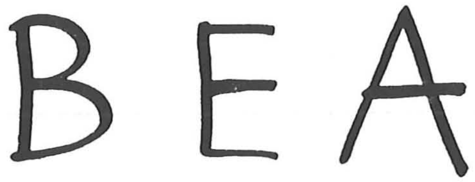


Figure 8.113: Example characters.

The euclidean distance from each point to the centroid was calculated. The referencing of the distance data to the centroid of the letter confers the property of positional invariance on the data. The power of the data is now normalised to be one. This confers the property of scale invariance on the data. A peak finding algorithm was used on the data and the data was shifted in a circular manner to put the peak value as the first data point. This operation gave the data the property of rotational invariance. Consequently the centroidal data patterns output by the pre processor should be shift, scale, and rotation independent.

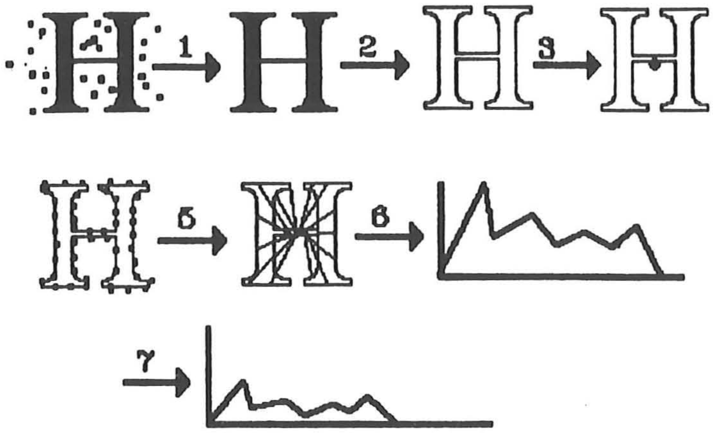


Figure 8.114: Preprocessor Stages.

Although in the majority of cases the preprocessor produces similar output patterns for an individual letter there are some letters which under certain circumstances will produce significantly different patterns. An example is the letter B. The first pass of the outline follower follows the outside of the letter, however the second pass could follow the top loop or the bottom loop and correspondingly generate significantly different patterns depending solely on which loop has the leftmost point. Consequently a connectionist model must be used which is powerful enough to cope with two significantly different clusters mapping to the same output index.

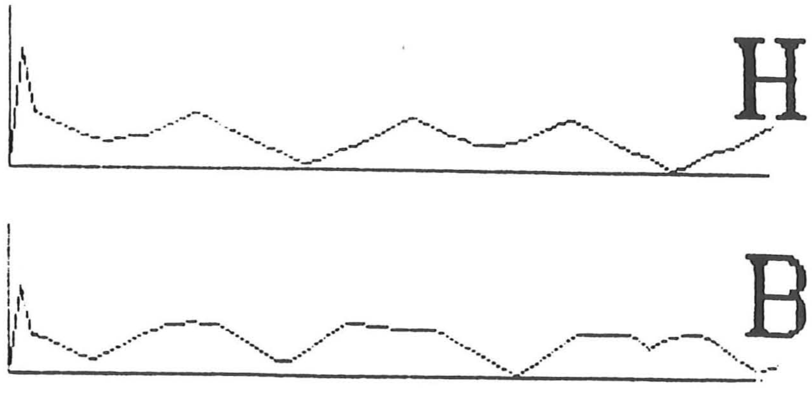


Figure 8.115 : Centroidal Data form.

8.4.2 Results

Direct recognition The network was found to recognise correctly all the characters in an eighty character font in around 7000 pattern presentations.

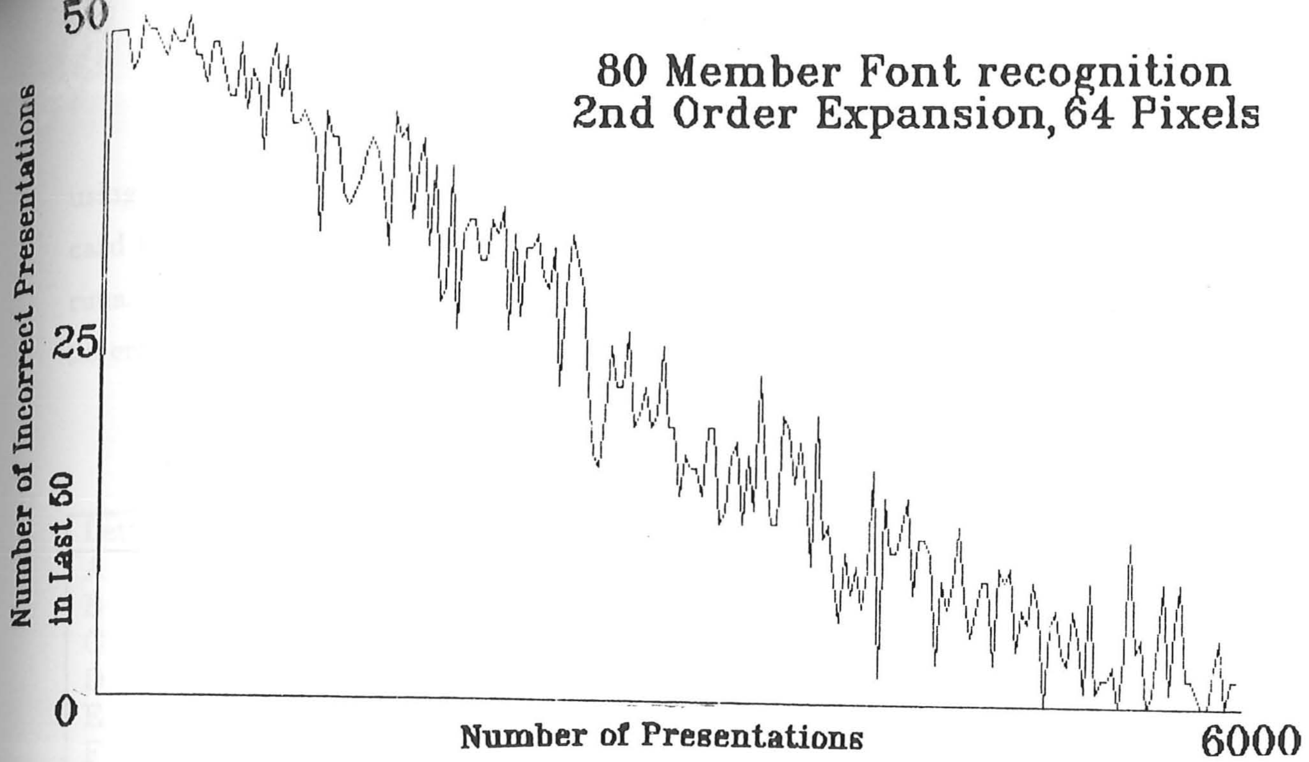


Figure 8.116: Plot of number of errors in the last fifty presentations against number of presentations.

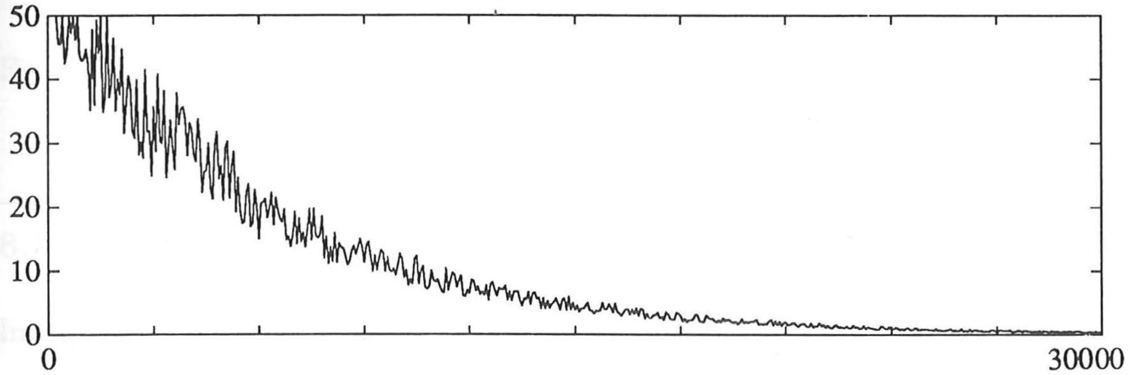


Figure 8.117. Plot of number of errors in the last fifty presentations against number of presentations with added noise.

Preprocessed Recognition The system was trained on twelve letters A.....L, initially using one card for each letter. Once the network was found to correctly recognise each card then gradually other cards with the same letters were introduced into the training runs. A new card was only introduced when the network had converged to a state giving no errors on the current cards.

Letter	Number of Cards Learnt
A	4
B	4
C	4
D	4
E	4
F	4
G	4
H	4
I	4
K	4
L	4

Figure 8.119 : Recognition Results

8.5 Conclusion

In the direct recognition case the network performed well, learning eighty separate class members: this is a higher number than usually reported for current networks. The convergence times were fast by comparison with current hidden-layer networks which have been reported to require many tens of thousands of presentations to learn fewer characters.
In the preprocessed case the network demonstrated its ability to cope with variation in the patterns, and consequently be applicable in practical recognition systems which require not only position invariance, but tolerance to character variation.

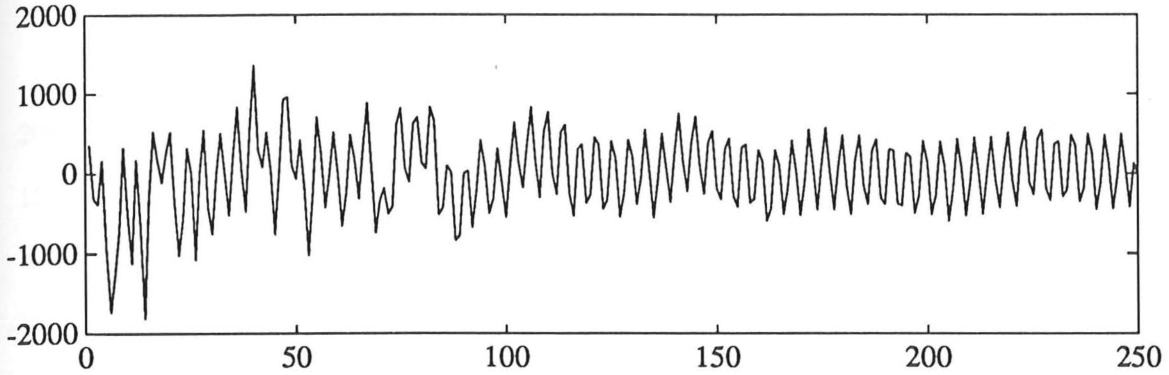


Figure 8.14: Example of Mug Data.

8.6 Audio Recognition

8.6.1 The Data

The following audio signals were generated by various acoustical means. For example, by hitting a cup with a pencil and so on. A,series of recordings were made of signals generated by the same means. No care was taken to produce identical sounds: for example, the cup was not always hit in the same place. The process was repeated for a series of different sound generating processes. These included a mug being hit, a click, a clap and so on. The plots also show the variation within one of the class sets.
The similarity of the frequency spectrum of different classes may also be seen.

8.6.2 Preprocessing and AR Model

As it is clear that the raw time series data is unlikely to produce clustering patterns for each class a preprocessor was used. The audio signals were preprocessed by over a range of forty samples finding the largest sample. The position of this sample was used as the starting position for ten samples to be taken. These ten samples had their power normalised and this pattern served at the network input. Ten point sections of data were taken from between the tenth and fortieth samples starting at the maximum sample in this range. This avoided scaling problems and reduce time origin variation effects.

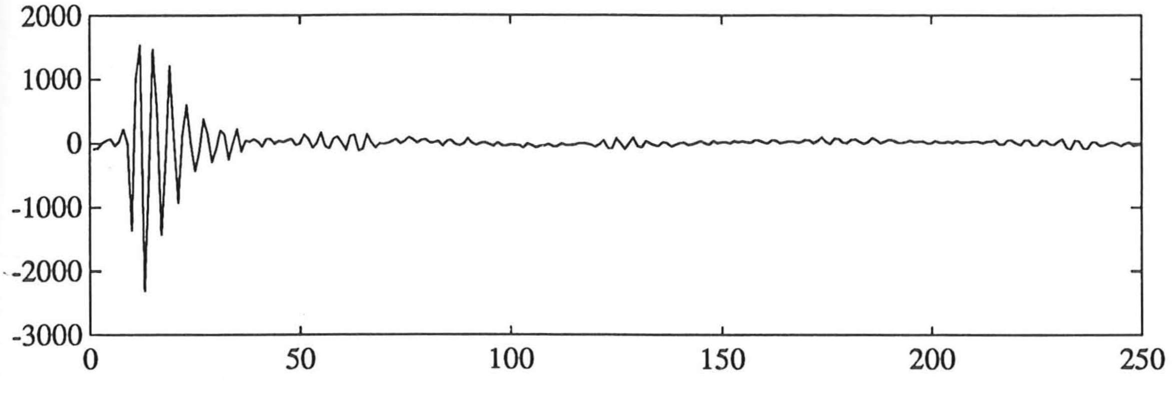


Figure 8.15: Example of Click Data.

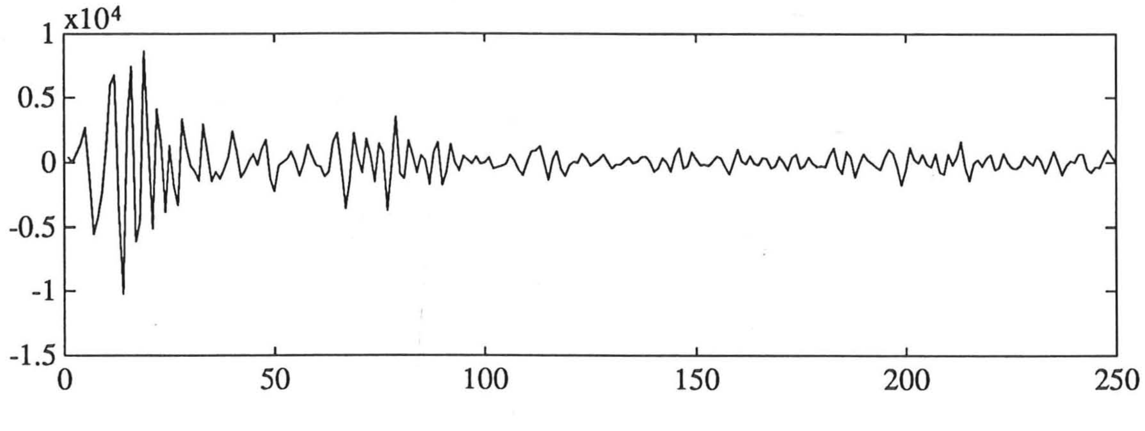


Figure 8.16: Example of Clap Data.

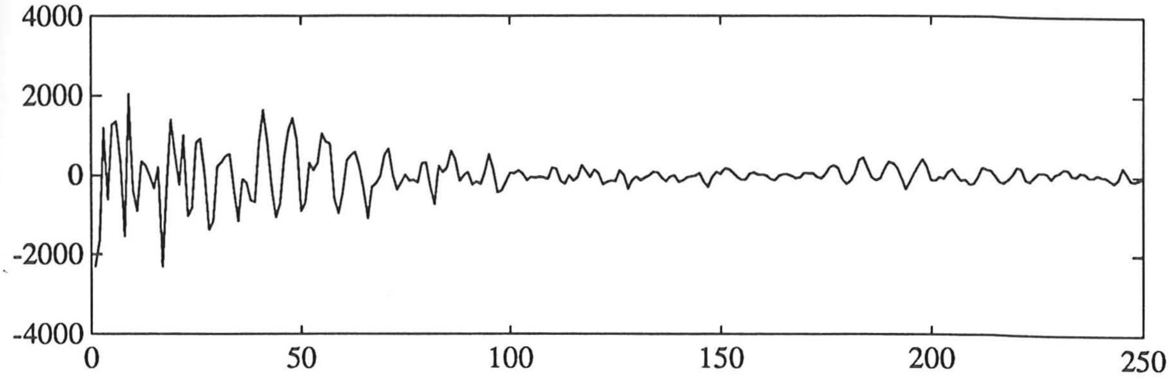


Figure 8.17: Example of Box Data.

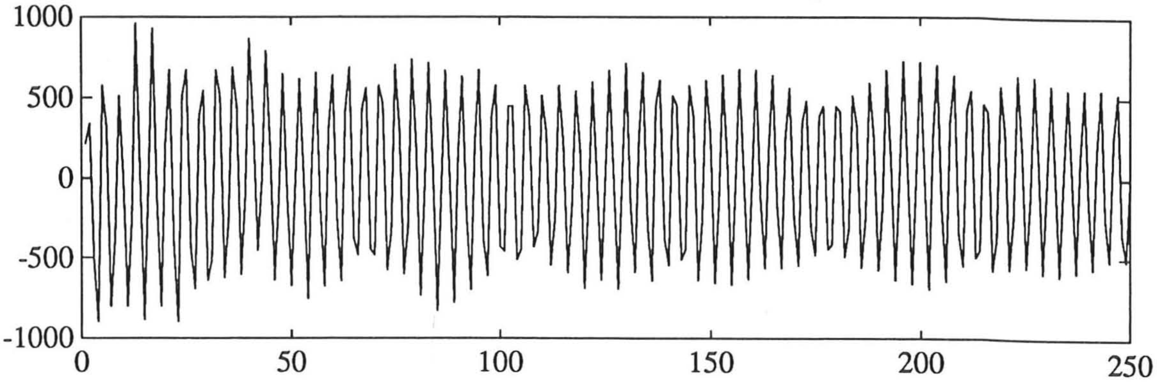


Figure 8.18: Example of Mug 0 Data.

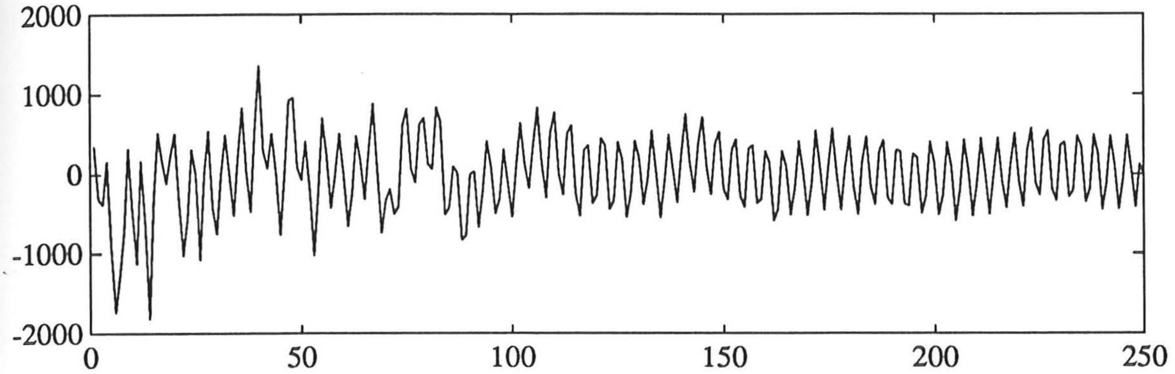


Figure 8.19: Example of Mug 4 Data.

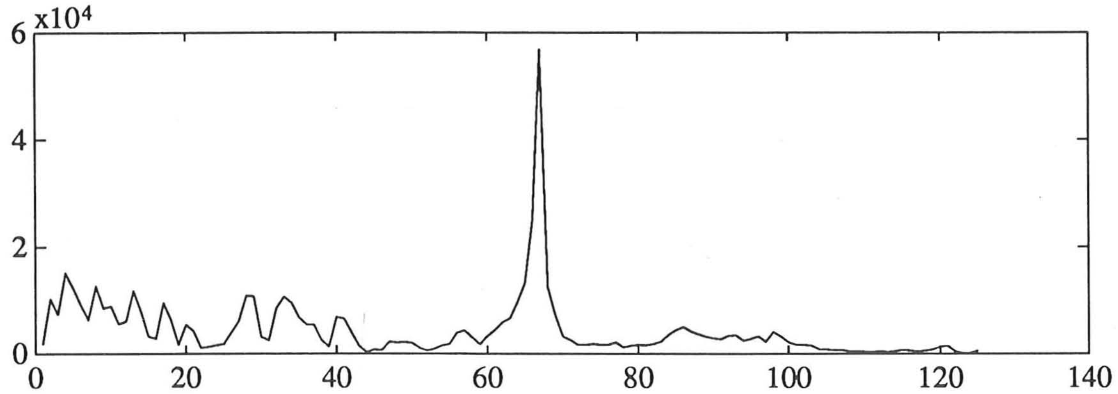


Figure 8.20: FFT of Mug 4 Data.

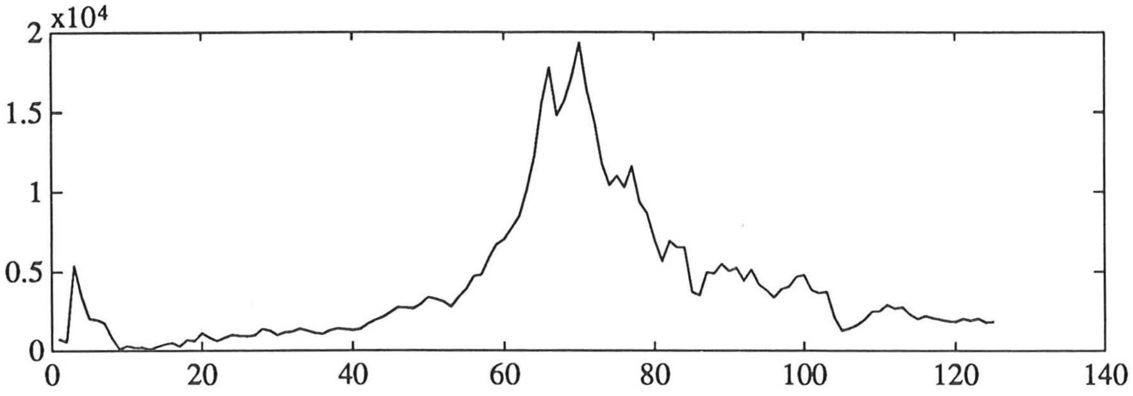


Figure 8.21: FFT of clik 0 Data.

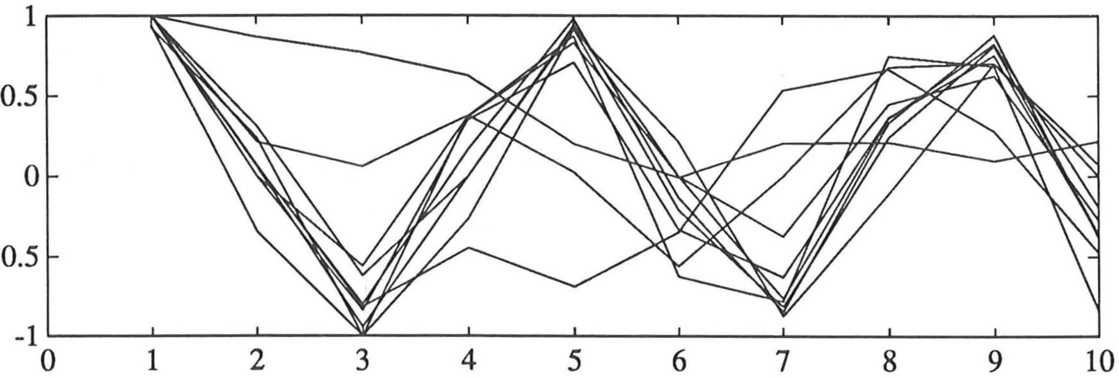


Figure 8.22: Processed mug data (10 examples).

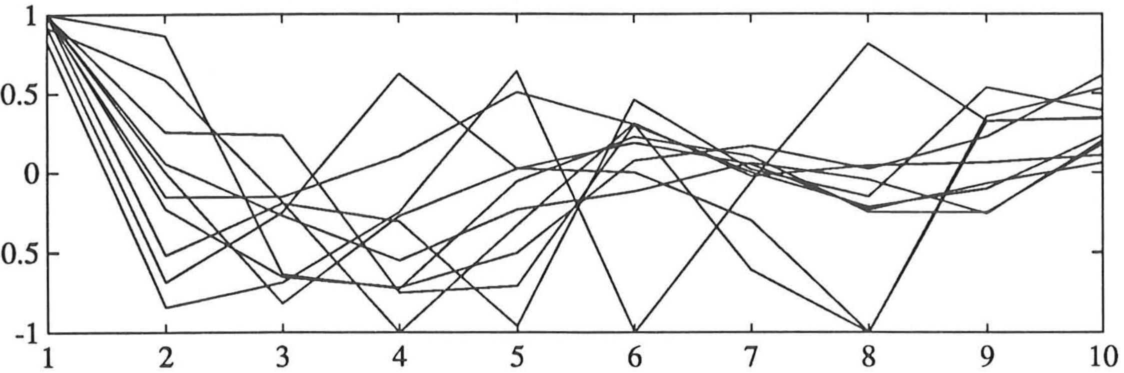


Figure 8.23: Processed Box data (10 examples).

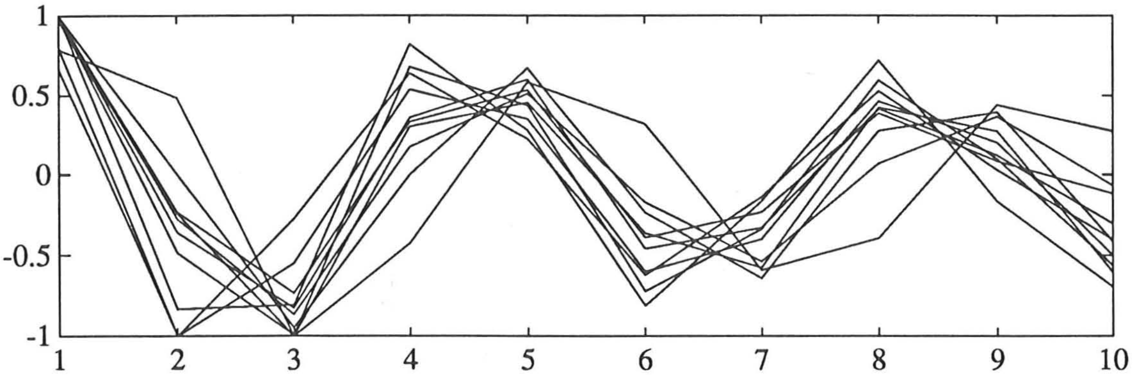


Figure 8.24: Processed click Data (10 examples).

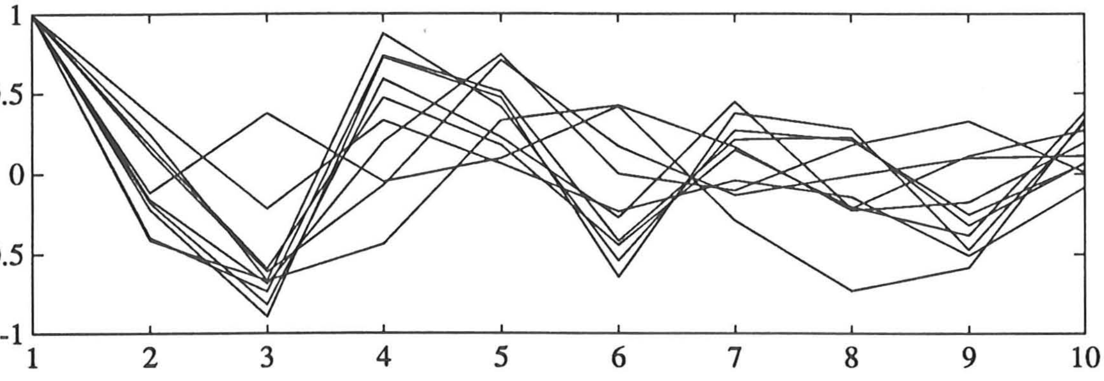


Figure 8.25: Processed clap Data (10 examples).

8.6.3 Results

The network was trained using 8 of the ten realisations of the data for all the classes and after learning to classify these correctly the 2 remaining examples were presented.

Sound	Desired Output	Actual Output	Error
Mug 0	1	1.01	.01
Mug 1	1	1.02	.02
Mug 2	1	.99	.01
Mug 3	1	.97	.03
Mug 4	1	1.02	.02
Mug 5	1	.99	.01
Mug 6	1	1.02	.02
Mug 7	1	1.00	.00
Mug 8	1	1.01	.01
Mug New 1	1	.96	.4
Mug New 2	1	1.3	.3
Click 0	3	3.01	.01
Click 1	3	3.00	.00
Click 2	3	3.00	.00
Click 3	3	2.98	.02
Click 4	3	2.99	.01
Click 5	3	3.01	.01
Click 6	3	2.84	.16
Click 7	3	2.96	.04
Click 8	3	3.02	.02
Click New 1	3	3.30	.30
Click New 2	3	3.12	.12
Clap 0	4	3.98	.02
Clap 1	4	3.99	.01
Clap 2	4	3.97	.03
Clap 3	4	4.02	.02
Clap 4	4	4.01	.01
Clap 5	4	3.99	.01
Clap 6	4	4.03	.03
Clap 7	4	4.17	.17
Clap 8	4	4.01	.01
Clap New 1	4	4.41	.41
Clap New 2	4	3.65	.35
Box 0	5	5.21	.21
Box 1	5	4.99	.01
Box 2	5	4.99	.01
Box 3	5	5.02	.02
Box 4	5	5.00	.00
Box 5	5	4.98	.02
Box 6	5	5.01	.01
Box 7	5	4.97	.03
Box 8	5	5.09	.09
Box New 1	5	5.39	.39
Box New 2	5	5.41	.41

8.6.4 Conclusion

The network performed well in the audio transient recognition task and required a comparatively light computational load to do so. The audio preprocessor could obviously be extended to form higher order correlations for systems with higher order statistics. The generalisation of the recogniser allowed it to recognise the previously unseen non-training patterns, although the error for such new patterns was considerably higher than for the training data.

Chapter 9

Output Mapping Problem and Principal Non-linearities

9.1 Output Mapping.

Consider a single input system with a series of output classes as shown below in figure 9:1.

This would require a high order non-linearity and consequently complex boundaries to the decision regions and so a larger network. If the output indices had been suitably chosen the whole problem could have been reduced to a linear one, figure 9:2.

Although the above is a simple case the problem becomes difficult to solve in multiple dimensions with a degenerate system. This question of output values has not been fully appreciated with current networks and a method for determining suitable indices would lead to far smaller networks. The approach being currently investigated involves a princi-

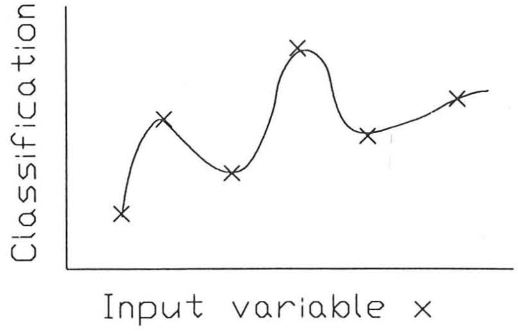


Figure 9.1: One Dimensional Problem

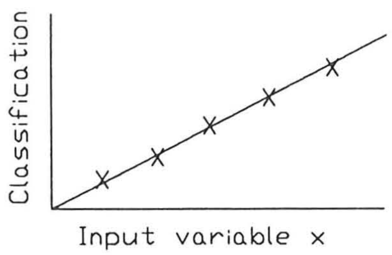


Figure 9.2: Linearised Problem

ple component analysis of the pattern difference vectors and has provided the ability to use smaller networks. It was hoped to develop an adaptive approach to network contraction.

9.2 Vector Space Approach

The problem of deciding on an optimal set of output indices may be approached from a vector space method.

The problem is to find a resolving vector, that is, the vector onto which the extended space class vector is projected, which maximises the difference in output value between the two classes with minimum separation.

Therefore it is necessary to find a projection vector $\mathbf{w}$ which maximises the difference of $\mathbf{x_i}.\mathbf{w}$ for the two input patterns $\mathbf{x_l}$ and $\mathbf{x_k}$ which give the closest values of $\mathbf{x_i}.\mathbf{w}$.

Thus it is necessary to maximise (Over all i,j such that i not equal to j) the minimum of:

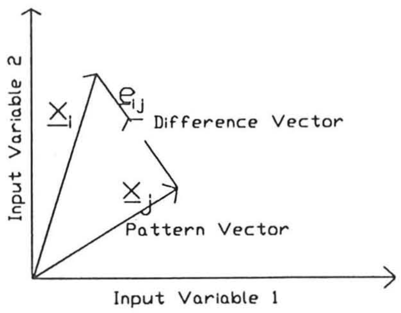
$$\mathbf{w}.\mathbf{x_i} - \mathbf{w}.\mathbf{x_j} = \mathbf{w}.(\mathbf{x_i} - \mathbf{x_j}) = \mathbf{w}.\mathbf{e_{ij}}$$

Where $\mathbf{e_{ij}}$ is the difference vector between the i th and j th input pattern vectors.

This may be addressed by a Lagrange multiplier approach. However, this led to a fruitless conclusion as the analysis gave an unusable result in terms of active and inactive constraints.

9.3 The Bayesian Approach

An attempt was made to apply Bayesian methods to the output mapping problem.



9.3.1 Bayes' Theorem

$$P(M \mid D) = \frac{P(D \mid M)P(M)}{P(D)}$$

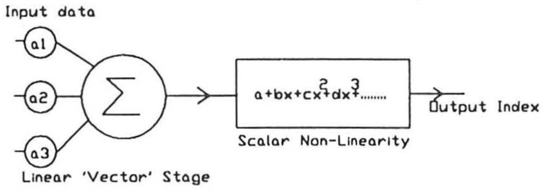
where $P(M \mid D)$ is the probability of the M given D, and P(D) the probability of D. Bayes theorem may be applied by using it to specify parameters in a model which is being tested against some data.

9.3.2 Application to Connectionist Models

Baye's Theorem may be applied in the connectionist model problem by using the following model. The operation of the network is separated into two stages. The first stage is the familiar vector operation. For the sake of simplicity the first stage will be limited to a linear operator (FIR filter). This first stage is followed by a scalar non-linearity. This non-linearity may be formulated in many ways but the simplest is a truncated polynomial. Bayes theorem is applied to estimate the coefficients of the power terms in this scalar non-linearity by integrating out the linear vector first stage as nuisance parameters. Thus Bayes theorem allows the separating of the decision boundary generation and the output mapping translation. This should lead to much smaller networks as the network's first stage will only have to converge to whichever solution separates the classes. The output mapping being a scalar non-linearity rather than a vector one is correspondingly far more efficient.

Consider a series of pattern vectors $\mathbf{x}_i$ with pattern classification output value c_i where i is the pattern index.

The proposed system modelling the relationship between the patterns and the output



values is assumed to be of the form shown .

The error is defined as usual as:

$$\epsilon_i = c_i - F[\mathbf{w}^{\mathbf{t}}\mathbf{x}]$$

where F[ ] is:

$$F[u] = \sum_{n=1,P} f_n u^n$$

f_n being the non-linearity coefficients to be determined. From this a likelihood function may be constructed for error values observed. Assuming ϵ to be distributed as a Gaussian variable with zero mean:

$$P(\epsilon_1 \ldots \epsilon_N \mid f_1 \ldots f_n, \mathrm{w}) = \frac{1}{\sqrt[N/2]{2\pi\sigma^N}} e^{-\frac{1}{2\sigma^2}\sum_{i=1,N}\epsilon_i^2} = P(D \mid M)$$

M referring to the model and D to the data. Bayes theorem may now be applied:

$$P(M \mid D) = \frac{P(D \mid M)P(M)}{P(D)}$$

$$P(F[], \mathrm{w} \mid \epsilon) = \frac{1}{\sqrt[N/2]{2\pi\sigma^N}} e^{-\frac{1}{2\sigma^2}\sum_{i=1,N}\epsilon_i^2} P(F[], \mathrm{w})$$

The coefficients $\mathbf{w}$ of the linear combiner may now be integrated out:

$$P(F[]) = \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} \frac{1}{\sqrt[N/2]{2\pi\sigma^N}} e^{-\frac{1}{2\sigma^2}\sum_{i=1,N}\epsilon_i^2} P(F[], \mathrm{w}) d\mathrm{w}$$

Assume $P(F[], \mathrm{w}) = A$ a constant. The analytical solution of this integral is difficult, so an attempt was made to perform numerical integration.

9.3.3 Conclusion

Although the Bayesian approach is an interesting attempt to investigate the output mapping problem the impracticality of the calculation of the integrals by numerical means for systems with more than a few coefficients is at present sufficient to prevent the use of the method. If an analytical solution or a good approximation to the solution could be found this approach could yet prove fruitful.

9.4 Optimisation of Output Map Order

A constrained optimisation approach was developed to allow the optimisation of the output index labels.

9.4.1 Derivation

The following approach to suitable selection of these indices has been shown to give good results. It uses a constraint to ensure that the output indices derived are separated from one another and consequently takes the form of a constrained optimisation.

Suppose that there are M classes and in the training phase there are N_i (i=1,M) presentations for each class. Let d_i be the classification index for each class. Let $\mathbf{x_n(i)}$ be the extended space pattern vector for the ith class and the nth example of it. Let $\mathbf{w}$ be the network weight vectors.

$$\epsilon = \sum_{i=1}^{M} \left[\sum_{n=1}^{N_i} \left[d_i - \mathbf{w}^t \mathbf{x}_n(i) \right]^2 \right]$$

The d_i must be constrained. A suitable constraint is found to be:

$$\sum_{i=1}^{M} d_i^2 = c$$

where c is an arbitrary constant.

The method of Lagrange multipliers may now be applied. Set up the objective function:

$$Q = \sum_{i=1}^{M} \left[\sum_{n=1}^{N_i} \left[d_i - \mathbf{w}^t \mathbf{x}_n(i) \right]^2 \right] + \lambda \sum_{i=1}^{M} (d_i^2 - c)$$

$$\frac{\partial Q}{\partial \mathbf{w}} = \sum_{i=1}^{M} 2 \left[\sum_{n=1}^{N_i} \left[d_i - \mathbf{w}^t \mathbf{x}_n(i) \right] \mathbf{x}_\mathbf{n}(i) \right] = 0 \ldots 1$$

$$\frac{\partial Q}{\partial d_k} = \sum_{n=1}^{N_k} 2 \left[d_k - \mathbf{w}^t \mathbf{x}_n(k) \right] + \lambda 2 d_k = 0 \ldots 2$$

where k=1,M.

From 1:

$$\sum_{i=1}^{M} \left[d_i \sum_{n=1}^{N_i} \mathbf{x}_n(i) - \left(\sum_{n=1}^{N_i} \mathbf{x}_n(i) \mathbf{x}_n^t(i) \right) \mathbf{w} \right] = 0$$

$$\sum_{i=1}^{M} d_i \sum_{n=1}^{N_i} \mathbf{x}_n(i) - \left[\sum_{i=1}^{M} \sum_{n=1}^{N_i} \mathbf{x}_n(i) \mathbf{x}_n^t(i) \right] \mathbf{w} = 0$$

Define:

$$\sum_{n=1}^{N_i} \mathbf{x}_n(i) = \mathbf{p}(i)$$

and:

$$\sum_{i=1}^{M} \sum_{n=1}^{N_i} \mathbf{x}_n(i) \mathbf{x}_n^t(i) = \mathbf{R_{xx}}$$

Then:

$$\sum_{i=1}^{M} d_i \mathbf{p}(i) - \mathbf{R_{xx}} \mathbf{w} = 0 \ldots 3$$

From 2:

$$(\lambda + N_k) d_k - \sum_{n=1}^{N_k} \mathbf{w}^t \mathbf{x}_n(k) = 0$$

$$(\lambda + N_k) d_k - \mathbf{w}^t \sum_{n=1}^{N_k} \mathbf{x}_n(k) = 0 \ldots 4$$

4 becomes:

$$(\lambda + N_k) d_k - \mathbf{w}^\mathbf{t} \mathbf{p}(k) = 0 \ldots 5$$

$k = 1, M$. Multiply 5 by d_k and sum over k (change to i):

$$\sum_{i=1}^{M} (\lambda + N_i) d_i^2 - \sum_{i=1}^{M} \mathbf{w}^\mathbf{t} \mathbf{p}(i) d_i = 0$$

For convenience, assume $N_i = N$ for $i = 1, M$:

$$(\lambda + N)\sum_{i=1}^{M} d_i^2 - \sum_{i=1}^{M} \mathbf{w}^{\mathbf{t}}\mathbf{p}(i)d_i = 0$$

But :

$$\sum_i d_i^2 = c$$

so:

$$\lambda + N = \frac{1}{c}\mathbf{w}^{\mathbf{t}}\sum_{i=1}^{M} d_i\mathbf{p}(i)\ldots 6$$

Substitution (5,3):

$$\sum_{i=1}^{M} \frac{1}{\lambda + N}\mathbf{w}^{\mathbf{t}}\mathbf{p}(i)\mathbf{p}(i) - \mathbf{R_{xx}w} = 0$$

$$\frac{1}{\lambda + N}\mathbf{P_{xx}w} = \mathbf{R_{xx}w}$$

where:

$$\mathbf{P_{xx}} = \sum_{i=1}^{M} \mathbf{p}(i)\mathbf{p}^{\mathbf{t}}(i)$$

Assuming $\mathbf{R_{xx}}$ non-singular then:

$$\mathbf{R_{xx}^{-1}P_{xx}w} = (\lambda + N)\mathbf{w}$$

That is, $\mathbf{w}$ is an eigenvector of $\mathbf{R_{xx}^{-1}P_{xx}}$.

$$\mathbf{w} = \alpha\mathbf{e}$$

where $\mathbf{e}$ is an eigenvector. Multiply 3 by $\mathbf{w}^{\mathbf{t}}$:

$$\mathbf{w}^{\mathbf{t}}\sum_{i=1}^{M} d_i\mathbf{p}(i) - \mathbf{w}^{\mathbf{t}}\mathbf{R_{xx}w} = 0\ldots 7$$

Substitute 6 into 7:

$$c(\lambda + N) = \mathbf{w}^{\mathbf{t}}\mathbf{R_{xx}w}\ldots 8$$

Substituting 8 into 5 gives:

$$d_k = \frac{c\mathbf{w}^{\mathbf{t}}\mathbf{p}(k)}{\mathbf{w}^{\mathbf{t}}\mathbf{R_{xx}w}}$$

From 8:

$$c\mu = \alpha^2\mathbf{e}^{\mathbf{t}}\mathbf{R_{xx}e}\ldots 9$$

where μ is the eigenvalue corresponding to $\mathbf{e}$, i.e. $\lambda + N = \mu$. Now, to decide on which eigenvalue to choose, consider the error:

We have:

$$\epsilon = \sum_{i=1}^{M} \left[\sum_{n=1}^{N_i} \left[d_i - \mathbf{w}^t x_n(i) \right]^2 \right] \ldots 10$$

and:

$$d_k = \frac{c\mathbf{w^t p}(k)}{\mathbf{w^t R_{xx} w}} \ldots 11$$

and from 9:

$$\alpha = \left[\frac{\mu c}{\mathbf{e^t R_{xx} e}} \right]^{\frac{1}{2}}$$

Expanding 10 gives:

$$\begin{aligned} \epsilon &= \sum_{i=1}^{M} \sum_{n=1}^{N} \left[d_i^2 - 2d_i \mathbf{w^t x_n}(i) + (\mathbf{w^t x_n}(i))^2 \right] \\ &= Nc - 2\sum_{i=1}^{M} d_i \mathbf{w^t p}(i) + \mathbf{w^t R_{xx} w} \end{aligned}$$

Substitute 7:

$$\epsilon = Nc - \sum_{i=1}^{M} d_i \mathbf{w^t p}(i) \ldots 12$$

From 11:

$$\mathbf{w^t p}(k) = \frac{d_k}{c} \mathbf{w^t R_{xx} w}$$

Substitute in 12:

$$\begin{aligned} \epsilon &= Nc - \sum_{i=1}^{M} \frac{d_i^2}{c} \mathbf{w^t R_{xx} w} \\ &= Nc - \mathbf{w^t R_{xx} w} \\ \epsilon &= c(N - \mu) \end{aligned}$$

where μ is the eigenvalue corresponding to $\mathbf{e}$. Hence the error is a minimum when the largest eigenvalue is selected. Substitution shows:

$$d_k = \frac{\mathbf{w^t p_k}}{\mu}$$

By use of this expression a set of 'optimal' output indices may be found. These indices are in general not integers and in fact a convenient approach to practical implementation has been found. The original integer indices are simply reordered for the classes in the same order as the optimal ones. This is found to work well as may be expected if the connectionist model is viewed as a multidimensional interpolator.

9.4.2 Results

The output mapping scheme was first applied to a two input variable problem with class clusters as shown below in figure 9.3.

These clusters and assignments gave the following outputs for each class. The output mapping system was now run to give the following optimal indices and the original indices reordered. The plots show the desired and actual outputs against pattern number. The solid line is the desired output and the dotted line the actual output.

It can be seen that the output error has been greatly reduced by optimal ordering of the output indices.

The output mapping approach was next applied to the problem of optical character recognition, using a 80 font member set with 8 by 8 pixels for each character. The desired output was the position of the character in the collating sequence 0 to 79.

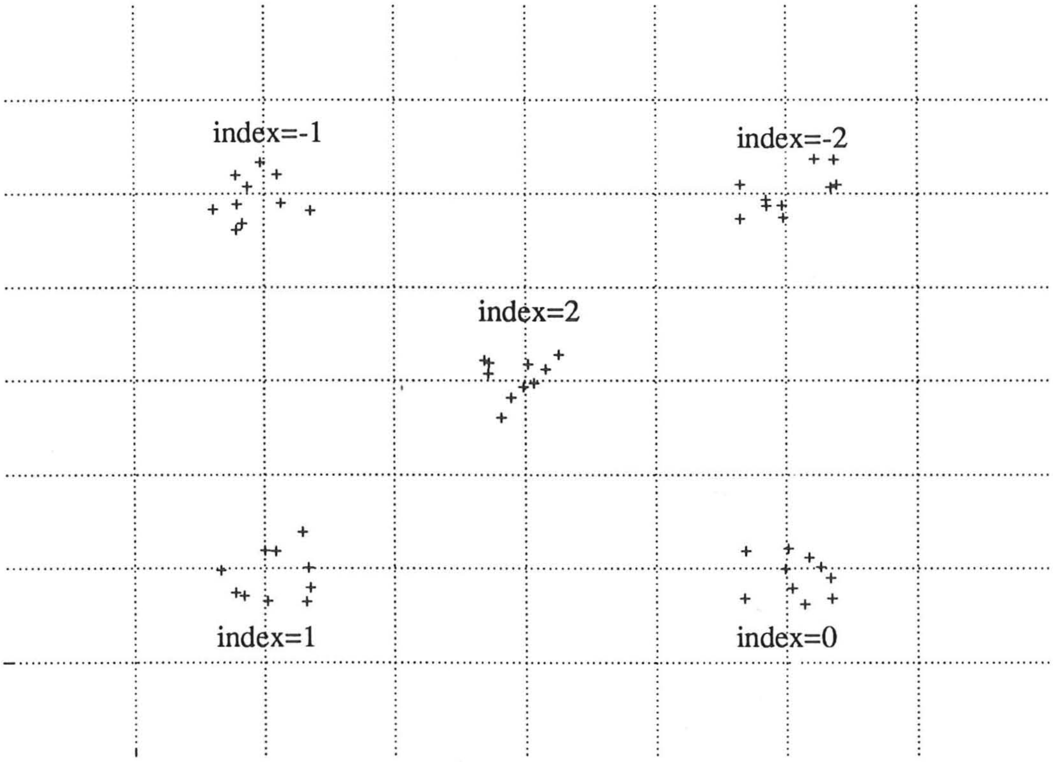


Figure 9.3: Input pattern clusters.

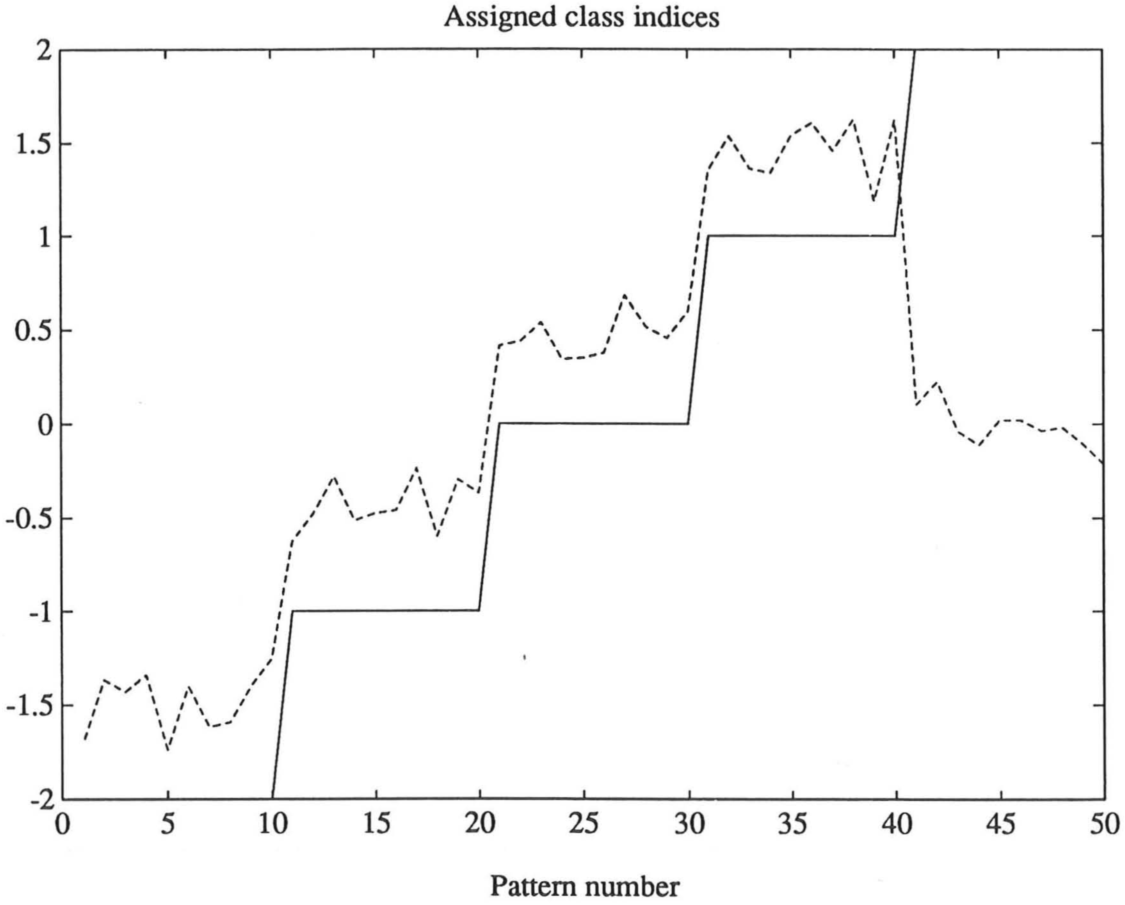


Figure 9.4: Plot of output with original indices. Solid line is desired output and dotted is actual.

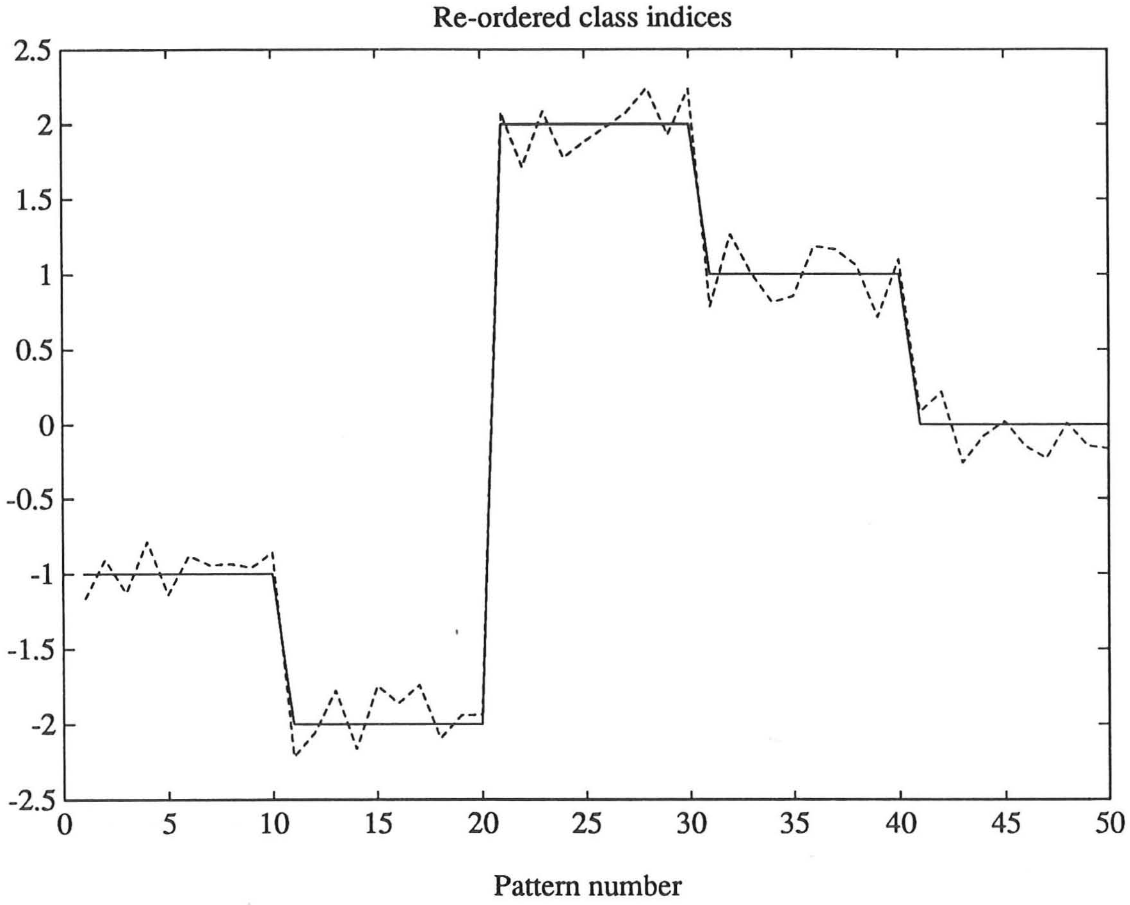


Figure 9.5: Plot of Outputs with reordered indices. Solid line is desired output and dotted is actual.

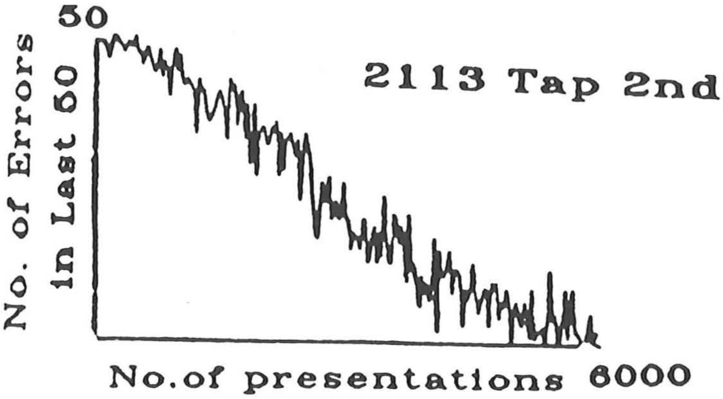


Figure : Results for 80 member font OCR with first and second order terms without optimisation of indicies.

With the arbitrary indices the linear system is not capable of solving the problem. However, if the indices are optimised a linear system can now solve the problem.

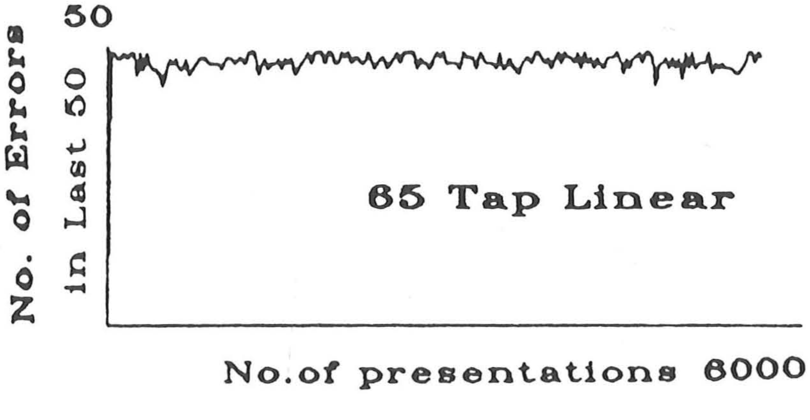


Fig. OCR results using linear first order terms without optimisation of output indices.

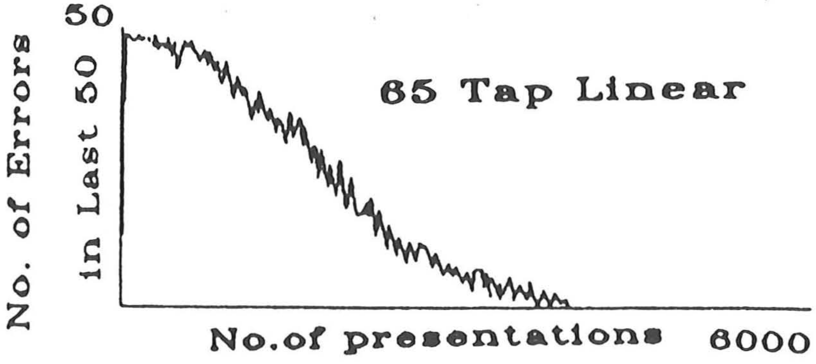


Fig. Results for 80 member font OCR with first order terms but optimised indices

9.4.3 Conclusion

The Volterra connectionist model has already been shown to have significant advantages over other systems in terms of analytical basis, speed of learning, scaling properties, and generalisation. The output mapping method demonstrates an approach which makes the Volterra connectionist model highly efficient computationally, by comparison with current neural networks. Although in the above results the system was applied to yield linear recognisers, exactly the same approach may be used to reduce a higher order non-linear recogniser to a lower order one. The network is no longer being constrained to fit arbitrary indices and so may utilise all of its degrees of freedom to improve recognition performance.

9.4.4 Analogy with Generalised Karhunen-Loeve Transform

The eigenvectors created by application of the output mapping may themselves be used for creation of a signal space. Consider the K.L. transform. This transform creates a signal space in which the signal energies are optimally concentrated in as few of the space defining vectors as possible. This is of course of interest in communications as it allows transmission of a given number of the new space vectors to transmit as much of the signal energy as possible. However, it is not necessarily a good thing for pattern recognition in which we are more interested in the class differences than the classes themselves. The space created by the output mapping eigenvectors also has some interesting properties which must be investigated in future work. It appears to offer a space in which the class separation for a limited selection of the eignevectors is maximised. So two classes which cannot be separated by projection on to the first eigenvectors may be separated by using a multidimensional output space composed from subsequent eigenvectors. More work is to be done on this subject in the future.

9.5 New Network Using Adaptive Output Mapping

Above, an output mapping scheme was developed which, in the pattern recognition problem, minimised the order of non-linearity required, to relate the input data and output indices, and thus enabled much smaller networks than usual to be applied to pattern recognition problems. The method was based on a constrained optimisation. The error defined over M classes in the training phase was defined as:

$$\epsilon^{2} = \sum_{i=1}^{M} \sum_{n=1}^{N_i} [d_i - \mathbf{w}\overset{t}{\mathbf{x}}_{\mathbf{n}}(\mathbf{i})]^2$$

where N_i is the number of presentations for each class and $\mathbf{x_n}(\mathbf{i})$ is the extended space pattern vector for the ith class and nth example of it. The d_i are constrained :

$$\sum_{i=1}^{M} d_i^2 = c$$

where c is an arbitrary constant. In the original derivation the method of Lagrange multipliers is now applied to give, by the solution of a generalised eigenvalue equation and the selection of the largest eigenvalue, a set of optimal output indices.

Such a block based method is not directly applicable to an adaptive approach as it requires the indices to be calculated a priori. Consequently an effort was made to derive an adaptive approach to optimisation of the output mapping.

An adaptive approach may be derived as follows:

If the original error equation is considered it is clear that the mean square error is a quadratic or unimodal function of the index d_i. This makes it suitable for a gradient search approach. This gradient based optimisation must, however, take account of the constraint equation to restrict the locus of search points to the values dictated by the constraint equation.

This constraint is most easily done by performing the index update in spherical polar coordinates: the weight update may still be performed in cartesian coordinates.

In SP coordinates the constraint equation becomes a constant radius constraint, with the various angular coordinates being updated by gradient search. The actual radius is an irrelevant gain factor which simply scales all the indices. Rewriting the d_i as $\cos\theta_i$ in SP, then :

$$\sum_{i=1}^{M} d_i^2 = c$$

becomes:

$$\sum_{i=1}^{M} \cos^2\theta_i = 1$$

As expected the constraint equation is a hypersphere in SP coordinates. The error is defined as before:

$$\epsilon = d_i - \mathbf{w}^{\mathbf{t}}\mathbf{x}$$

Or in SP coordinates for the output d_i space:

$$\epsilon = \cos\theta_i - \mathbf{w}^{\mathbf{t}}\mathbf{x}$$

To optimise the outputs a gradient search is performed to minimise ϵ with respect to the indices, but steps are only allowed around the hypersphere because of the constraint equation, that is, the component of the steps in the radial direction is zero. The constrained gradient search then becomes:

$$\theta_{i+1} = \theta_i - \mu_{om}\nabla(\epsilon^2)$$

where μ_{om} is a step size constant parameter. Now:

$$\frac{\partial E(\epsilon^2)}{\partial \quad \theta_i} = 2\epsilon\frac{\partial\epsilon}{\partial \quad \theta_i} = -2\epsilon\sin\theta_i$$

therefore the algorithm is:

$$\theta_{i+1} = \theta_i + 2\mu_{om}\epsilon\sin\theta_i$$

$$\epsilon = \cos\theta_i - \mathbf{w}\overset{\xi}{\mathbf{x}}$$

The classification attached to each class may be discovered by running a known training pattern through the system as the adaption proceeds. It is hoped to continue work on this approach in the future.

9.6 Principal Non-linearities.

As is appreciated in the GMDH [36] approaches not all of the polynomial terms may actually play an important role in network performance for a given problem. Some terms may be redundant. It is a shame to have to calculate these terms each time. Methods however which use a priori information to remove random terms cannot perform well in general for on-the-job applications as the redundancy in the terms may change. The remaining terms which are required for good network performance may be called the principal non-linearities.

9.7 Adaptive Identification of Principal Non-linearities

It is clear that in order to perform a totally general transfer function a very large number of terms from the Volterra series may be needed. Past experience has shown that many real problems may, however, be solved with limited order networks of order one, two or three. This is especially true in the field of pattern recognition where the above output optimisation has been applied to reduce the order of required transfer function . It has also become apparent that many problems may be completely independent of some of the terms in the general polynomial. This independence or irrelevance of certain terms leads to larger, less efficient networks, but it is very difficult to predict in advance which terms will be irrelevant and remove them. It has also been demonstrated that networks composed of random terms will in general operate well (shown later in this thesis). This frees the network from the constraint of having complete Volterra expansions. An alteration to the adaption algorithm is derived here which allows the network to organise itself to eliminate some of these unhelpful terms to decrease network size. Alternatively the computational load decrease incurred by the network when a term is lost may allow another term to be added. Such a network which removes terms to optimise its structure may be referred to as a self collapsing network (SCN). If pattern class densities in the multidimensional space are considered, an interesting theory may been proposed that suggests that for complex pattern recognition problems limited non-linearities should be required, for well preprocessed problems. Consider an image, which for human purposes may be considered as 256 by 256 pixels. This means our signal space has 256 by 256 dimensions and it is likely that in such a large space a low order network could separate many classes if they were correctly output mapped.

9.7.1 Degeneracy

The problem in the SCN is to detect redundant terms and neutralise them. Consider a degenerate problem in which the extended space correlation matrix $\mathbf{R_{ex}}$ is singular. For such cases there is in fact a series (quite often an infinite number!) of possible solutions. The standard block approach to such a problem is to employ Singular Value Decomposition to yield the unique minimum norm solution. SVD is however not suitable for an adaptive approach and incurs a relatively heavy computational overhead. In a minimum norm solution the weights associated with any redundant terms will be set to zero.

9.7.2 Adaptive Minimum Norm Solution.

Attempts were made to develop such an algorithm: however it became apparent that many methods failed with such rank deficient problems and the only approach found was to use Singular Value Decomposition.

Although there are recursive implementations of SVD these are computationally heavy and would be impractical for many connectionist model applications. In order to obtain a minimum norm solution the following adaptive approach was derived with a new error measure:

$$\eta_k^2 = (d_k - y_k)^2 + \alpha \mathbf{w}^{\mathbf{t}} \mathbf{w}$$

In which α is a constant setting the relative importance of term elimination to reduction of mean square error. Consequently if a term is truly redundant the value of its weight will be arbitrary. The above addition to the error term adds a penalty for setting such a weight to a non-zero value. The square error may be differentiated with respect to the weights to lead to a stochastic gradient update equation:

$$\mathbf{w}_{k+1} = \mathbf{w}_\mathbf{k} - \mu_{scn}\nabla(\eta_k^2)$$

Where μ_{scn} is a step size constant. The derivative may be found as follows:

$$\nabla(\eta_k^2) = \nabla(\epsilon_k^2) + \nabla(\alpha \mathbf{w}_\mathbf{k}^\mathbf{t}\mathbf{w}_\mathbf{k})$$

Where $\epsilon_k = d_k - \mathbf{w}_\mathbf{k}^\mathbf{t}\mathbf{x}_\mathbf{k}$ as before.

The ϵ_k^2 derivative can (as in the LMS case) be shown to give:

$$\nabla(\epsilon_k^2) = -2\epsilon\mathbf{x}_\mathbf{k}$$

The $\mathbf{w}_\mathbf{k}^\mathbf{t}\mathbf{w}_\mathbf{k}$ term gives:

$$\nabla(\mathbf{w}_\mathbf{k}^\mathbf{t}\mathbf{w}_\mathbf{k}) = 2\alpha\mathbf{w}_\mathbf{k}$$

put $\alpha_2 = 2\alpha\mu_{scn}$
$\beta = 2\mu_{scn}$

Hence the update becomes:

$$\mathbf{w}_{\mathbf{k}+1} = \mathbf{w}_\mathbf{k} + \beta\epsilon\mathbf{x}_\mathbf{k} - \alpha_2\mathbf{w}_\mathbf{k}$$

It can be seen that the norm minimiser adds only one multiplication and one subtraction per weight update.

Now that the minimum norm solution is being preferred by the adaption algorithm, terms may periodically be eliminated. To give a constant size network terms may also be randomly added each time. Through this process the network may perform self structuring. Consequently, the irrelevant polynomial terms are selected and eliminated.

Similarly, the total number of terms may be set as a function of the classification error rate. It should, however, be noted that the optimal solution of the weights (in the mean square error sense) is perturbed by the extra term and consequently α is normally kept small to minimise this perturbation. The effect of α being small is for the network to initially adapt its weights irrespective of the norm constraint. Only when the M.S.E is minimised will the α term begin to have more effect. The setting of α is determined by the relative importance of network collapsing and minimising output error.

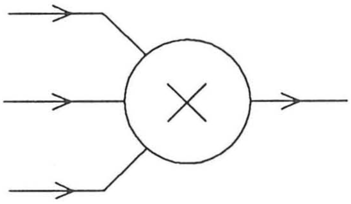


Figure 9.6: Type I Neuron.

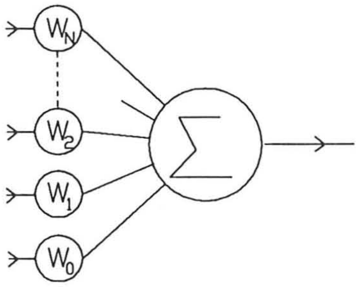


Figure 9.7: Type II Neuron.

9.7.3 Perceptron Analogy

As will be discussed in chapter 10 the Volterra network can be composed from randomly connected type I and II perceptrons (Figure 9.6 and 9.7).

The second type of neuron is the output neuron. This is the same as the linear perceptron with weighted input and one is needed for each multiclass output.

In perceptron terms the performance of each perceptron is being investigated and if it is found to be of little use to the system its inputs connections are destroyed and new ones 'grown' to random connection points.

Terms may also be added to keep an optimised constant size structure. This has been done by random selection, although a host of other rationales could be investigated using higher order correlations.

It should be noted that the weight energy penalty term tends to slightly distort the normal weight solution and thus leads to a misadjustment giving a slightly higher mean square error. The effect can be avoided by only including the penalty term during periods of training or retraining rather than during the recognition phase.

9.7.4 Adaptive Self Collapsing Network Examples.

As a simple example, a two pattern problem is addressed. A second order extension is used:

$$y = w_0 1 + w_1 x_1 + w_2 x_2 + w_3 x_1 x_2$$

The two patterns are alternately presented and the smallest coefficient after 50 presentations is shown:

Pattern 1	Pattern 2	Redundancy	Eliminated Weight
(2,-1)	(-1,2)	$x_1 x_2$	w_3
(2,-1)	(-2,-1)	x_2	w_2
(-1,-2)	(-1,2)	x_1	w_1

The results demonstrate that the SCN correctly eliminates unnecessary terms and so can solve the problem with a smaller network. Unlike SVD the effect of noise of the apparent degeneracy of a degenerate problem is not of importance as the SCN method does not require only exact degeneracy to work but will assume the minimum norm solution for near degenerate problems.

9.8 Conclusion

The output mapping approach leads to much smaller networks and consequently significant decreases in computational load with no degradation in performance. A full understanding of the implications of the new output mapping approach will require more work in order to investigate its behaviour.

The new network has the ability due to its derivation to model general non-linearities and maintains high speed convergence without local minima problems, due to its unimodality. It will also function well for multiclass problems and will work with computational loads far lower than other networks by minimising the non-linearity of the problem and then concentrating on the principal non-linearities of the problem.
The self collapsing adaptive mapping final network is for input vector **h** :

$$\mathbf{x} = F(\mathbf{h})$$

where $F()$ is a random selection of polynomial terms.

$$y_k = \mathbf{w}^t\mathbf{x}$$

$$\epsilon = \cos\theta_i - \mathbf{w}^{\mathbf{t}}\mathbf{x}$$

Weight update:

$$\mathbf{w}_{\mathbf{new}} = \mathbf{w}_{\mathbf{old}} + \beta\epsilon\mathbf{x} - \alpha_2\mathbf{w}$$

Index update:

$$\theta_{newi} = \theta_{iold} + 2\epsilon\sin\theta_{iold}$$

After a preset number of presentations, terms with a weight value of modulus less than a threshold are eliminated from $F()$ and a new random polynomials term added.

Chapter 10

Implementation

10.1 Introduction

In order to facilitate the understanding of the new network it has so far been discussed as two sections: firstly the vector state expansion and secondly the adaptive filter section. The same network may be implemented in a more distributed form composed of a series of identical 'neurons' by factorising the expansion polynomial. Once this has been done it is possible to find factorisations composed of identical units. Consequently a network may be created using a network of simple terms. Another approach which is currently under investigation is to define two types of 'neuron'. The first is composed of a simple multiplier and the second a standard linear perceptron.

An interesting form of the second implementation is that in which the network elements are randomly connected. This generates a connectionist model containing random terms from the non-linear state expansion. However, provided that the network is large enough, the inherent degeneracy allows the network to find possible solutions without the missing terms.

10.2 Exact Implementation

The first computational reduction in implementing the network may be found by realising that terms such as x_1x_2 and x_2x_1 are numerically the same and thus only one of them is needed. Computational load may also be saved by storing the result of each term's calculation before it is multiplied by its coefficient so that the term is not recalculated for the update equation. The network may be directly implemented with each term being calculated from scratch. This is inefficient but easy to code. A more efficient approach is to use previous lower order terms to calculate higher order ones. The full application of this idea is in the nested approach. The vector space expansion equation may be factorised to give more efficient implementations of the network by utilisation of nesting. Consider the simple expansion:

$$y = w_{00} + w_{10}x_1 + w_{01}x_2 + w_{11}x_1x_2 + w_{20}x_1^2 + w_{02}x_2^2 + w_{21}x_1^2x_2 + w_{12}x_1x_2^2 + w_{30}x_1^3 + w_{03}x_2^3$$

This may be nested in the form:

$$y = p_0 + x_1(p_1 + x_1(p_2 + x_1(p_3)))$$

Where:

$$p_0 = w_{00} + x_2(w_{01} + x_2(w_{02} + x_2w_{03})))$$

$$p_1 = w_{10} + x_2(w_{11} + x_2(w_{12}))$$

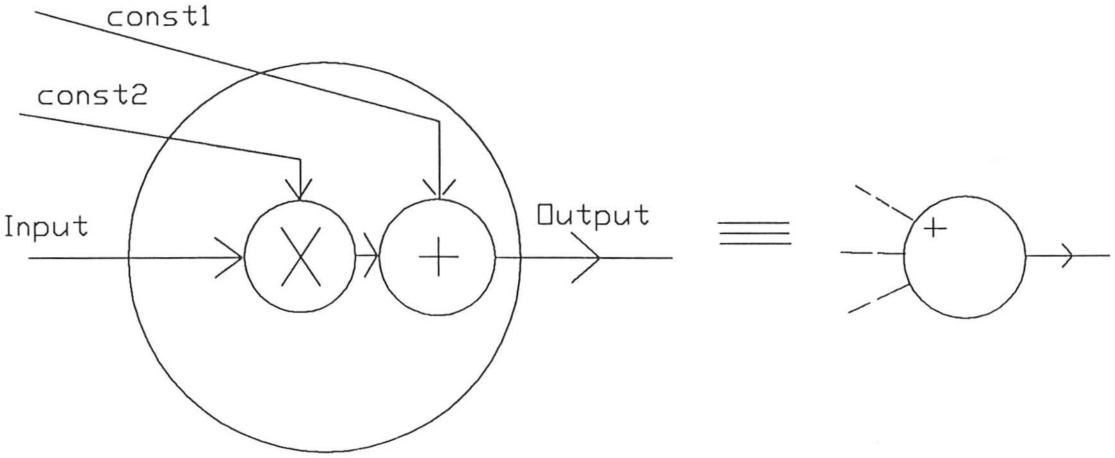
$$p_2 = w_{20} + x_2(w_{21})$$

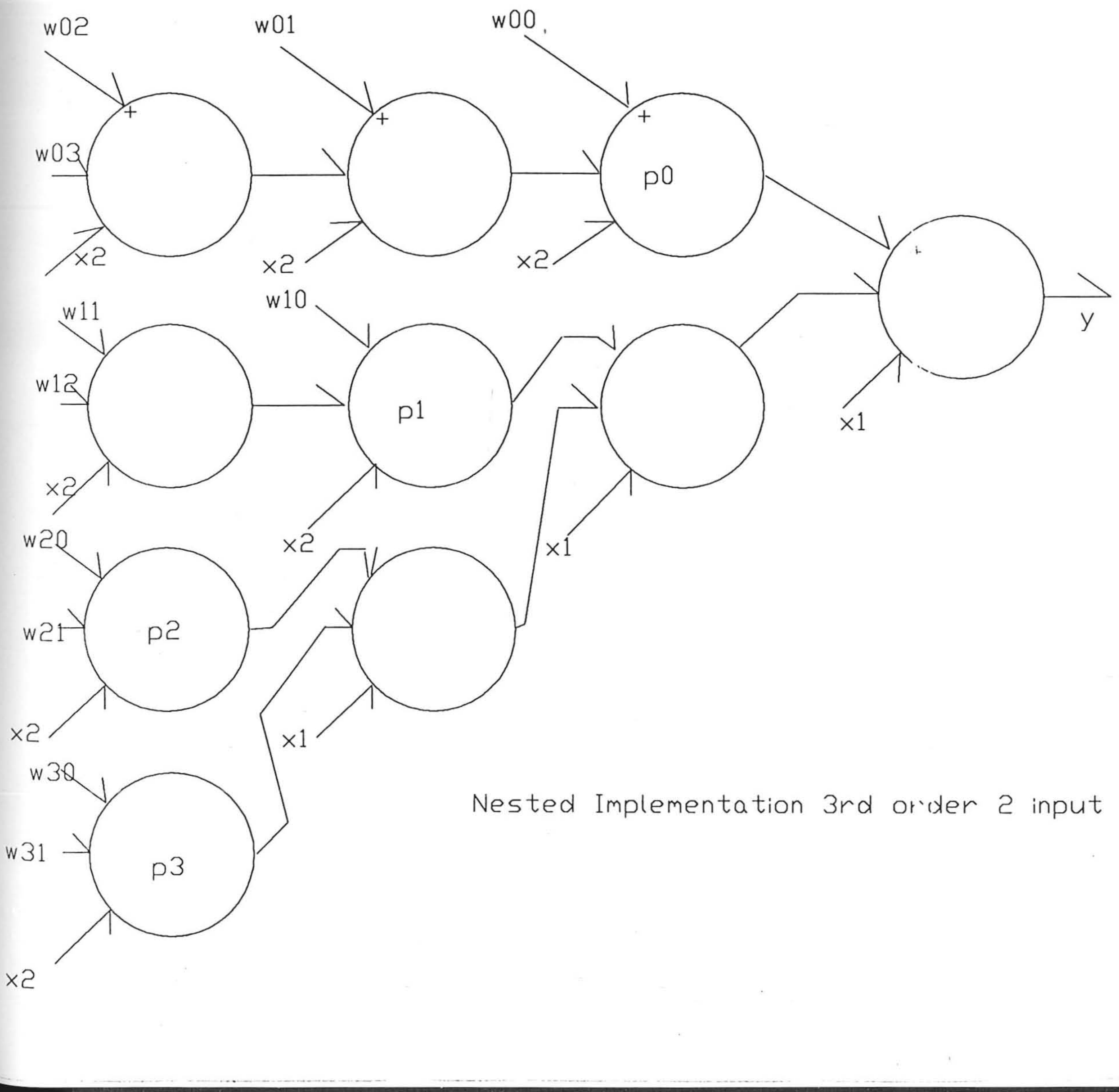
$$p_3 = w_{30} + x_2(w_{31})$$

That is, a simple recursive application of:

$$out = const1 + in * const2$$

Other implementation approaches are to separate the expansion and summation operations to allow multiple problem solving and to allow the random internal connections of self structured networks. The connections of such random or self structuring networks can be easily implemented by a two input quasi-multiplicative element (or 'neuron'!) whose inputs can be arbitrarily connected. This can easily and efficiently be implemented for simulation on a digital computer by storing the input positions for all the elements on a stack and using indirect addressing to fetch the inputs.





Nested Implementation 3rd order 2 input

10.3 Random Interconnection

It is of interest to consider the effects of incorrect or even random connection between the units of the new network. This is of interest as many possible future implementations of large networks may not be able to guarantee 100 percent fabrication accuracy. Such future implementations using light or analog VLSI may use tens of thousands of perceptrons and hundreds of thousands of interconnections. To produce such systems in a totally accurate manner may be very difficult. It is also considered unlikely that biological information processing systems can grow and maintain themselves exactly to a predetermined 'wiring' diagram . Thus it is of interest to consider the effects of faulty or missing connections.

In the extreme case it is interesting to consider the performance of large versions of the network composed of perceptrons connected in a random fashion determined by some probability generating function. This would be of interest as it would allow self-organising networks. One scenario has each perceptron growing out a random connection to another perceptron, and there are variations on this theme.

10.3.1 The Two Neuron Random Model

The network may be decomposed into two types of neurons. The first is a neuron which performs a vector space expansion. This may be as simple as a multiplication of two inputs, or a neuron taking in multiple inputs to give the product of them all.

The second type of neuron is the output neuron. This is the same as the linear perceptron with weighted inputs and one is needed for each multiclass output.

The network is made up by the random connection of layers of the type I neurons. Random connections are made from the inputs of the type II neurons into the type I layers.

10.3.2 Results for Random Connection

The random network was now applied to the optical character recognition problem and the number of errors in the last fifty presented characters was plotted against the presentation number.

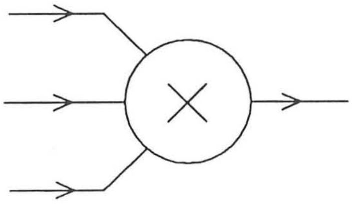


Figure 10.1: Type I Neuron.

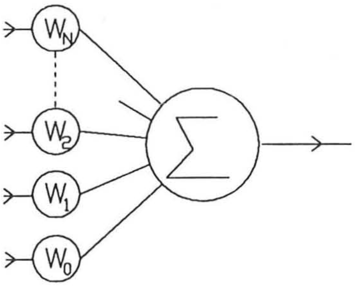


Figure 10.2: Type II Neuron.

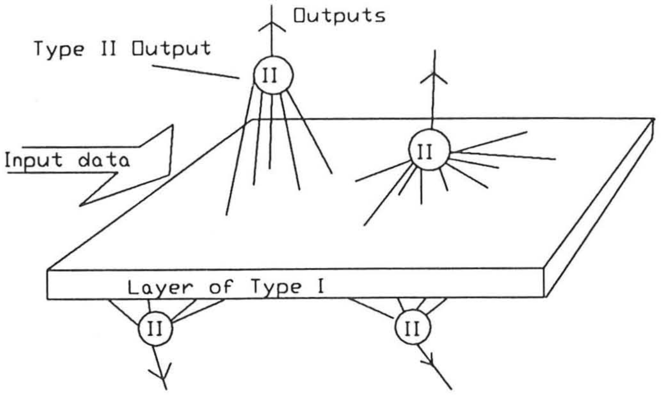


Figure 10.3: Random Network.

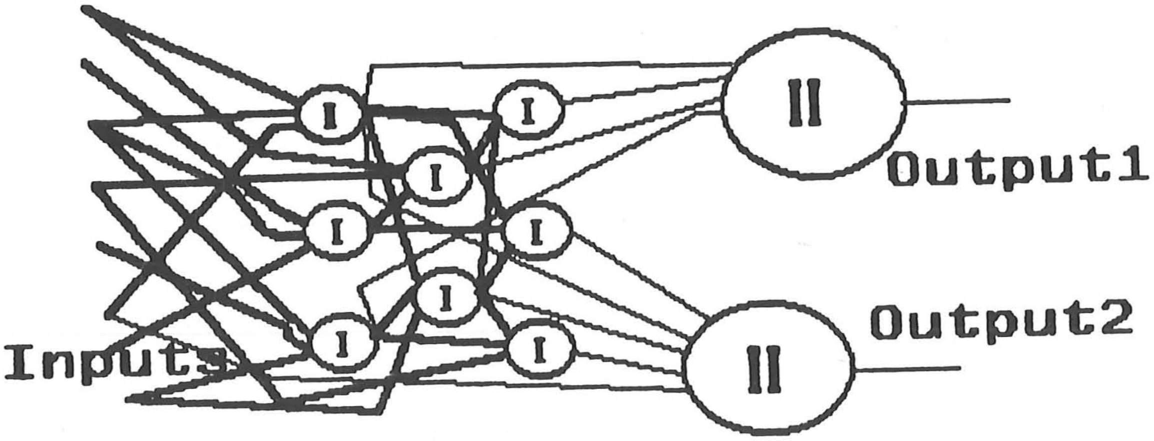


Fig. Example Random Network

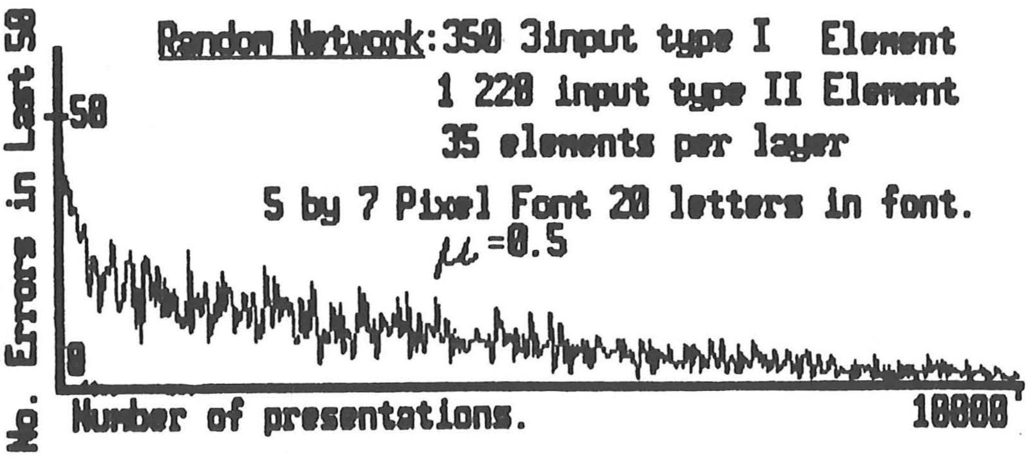


Fig. Results for Random Network

10.3.3 Conclusion

The random network works well and will converge to give a viable operation; as may be expected the computational load is heavier than the non-random network.

10.4 Limited Dynamic Range and Imperfect Multiplication

In the previous section the effect of imperfect or random interperceptron connections was considered. In this section the effect of imperfections in the perceptrons is considered. Such imperfections are considered as those expected to arise in an analog implementation such as light, analog electronics and so on.

Practical analog systems can only work over a limited dynamic range: if the signal is too small the signal becomes lost in noise, too large and it will exceed the range of the system (for example, hard limiting in an amplifier). In order to avoid this problem the network should be able to function with all internal calculations giving signals within some modest dynamic range.

In implementing of the new network it is clear that the effect of multiplications implemented by analog multipliers should be considered. The exact form of these imperfections will be specific to the exact implementation used and as such is beyond the scope of this discussion. However, the effect of general degradation may be considered. This degradation may be modelled as low level white noise added to the multiplier transfer function.

10.4.1 Results

A two dimensional problem was attempted using a network for which the expansion stage multiplications were limited in dynamic range by a hard limiter for product results over 100 in magnitude. The input values were normalised over the range 0-10.

The extension was next calculated using a noise corrupted multiplier, the noise being uniformly distributed over the range of 0 to 0.1 and fixed as a function of time. The network inputs were normalised to the range -0.5 to +0.5.

$$c = a * b$$

Was replaced by:

$$c = a * b + 0.1 * random((a + 0.5) * 5, (b + 0.5) * 5))$$

Where *random* is a matrix of uniformly distributed numbers ranging from 0 to 1.

1. Fig.10.4:Output against two inputs for perfect two input multiplier.

2. Fig.10.5:Output against two inputs for static noise corrupted two input multiplier.

3. Fig.10.6:Output against two inputs for limited dynamic range two input multiplier.

4. Fig.10.7:Decision regions for limited dynamic range network case.

5. Fig.10.8:Discriminator surface for limited dynamic range multiplier network.

6. Fig.10.9:Decision regions for static noise corrupted multiplier network.

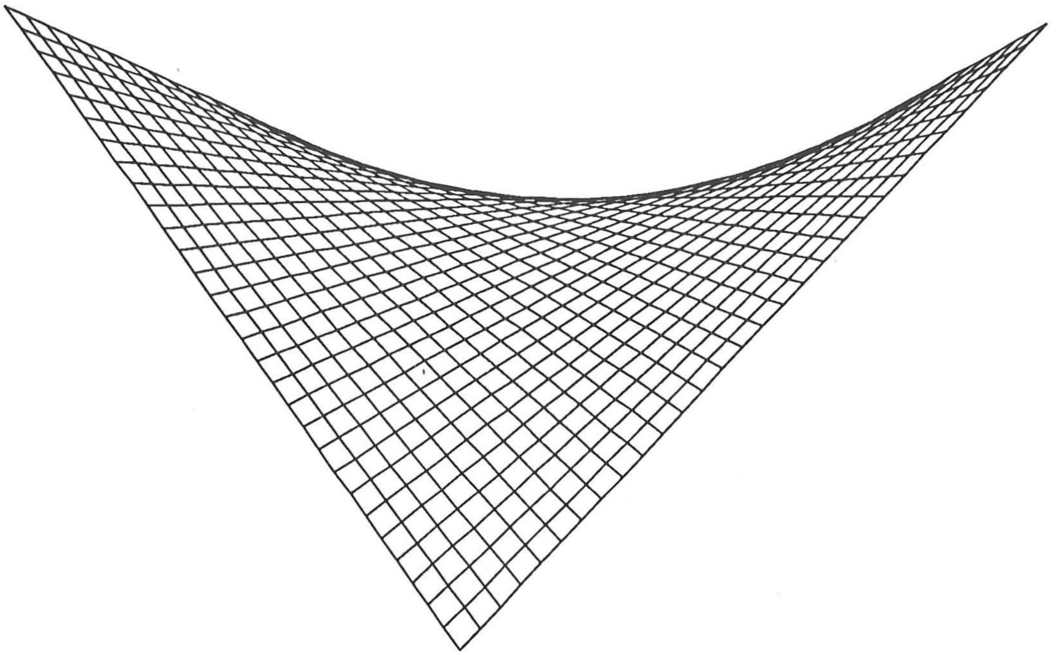


Figure 10.4: Ideal Multiplier

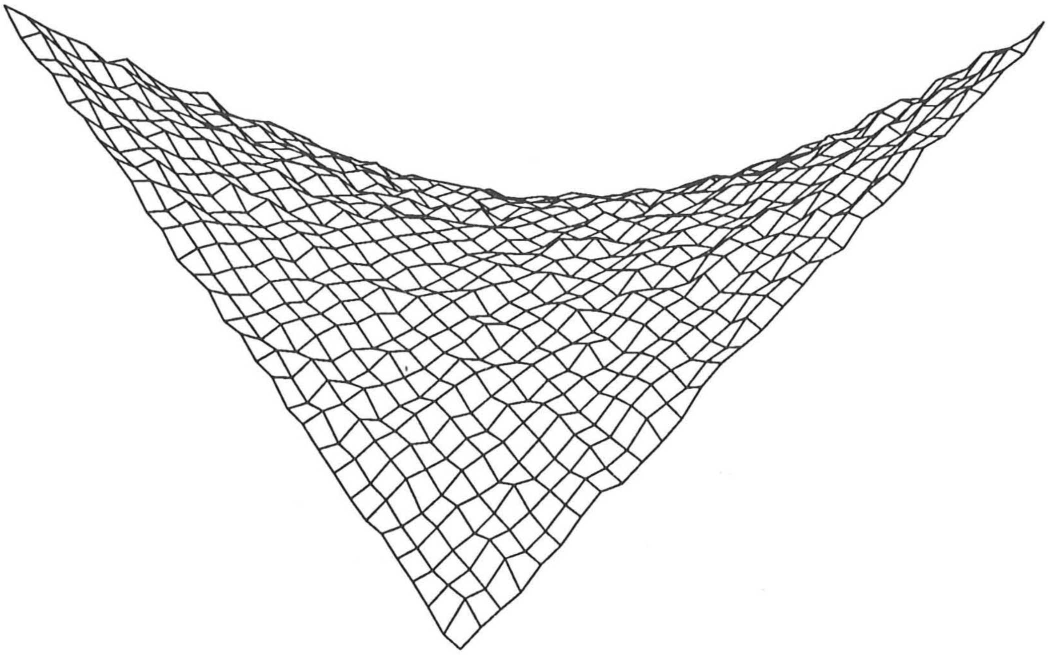


Figure 10.5: Static Noise Corrupted Multiplier.

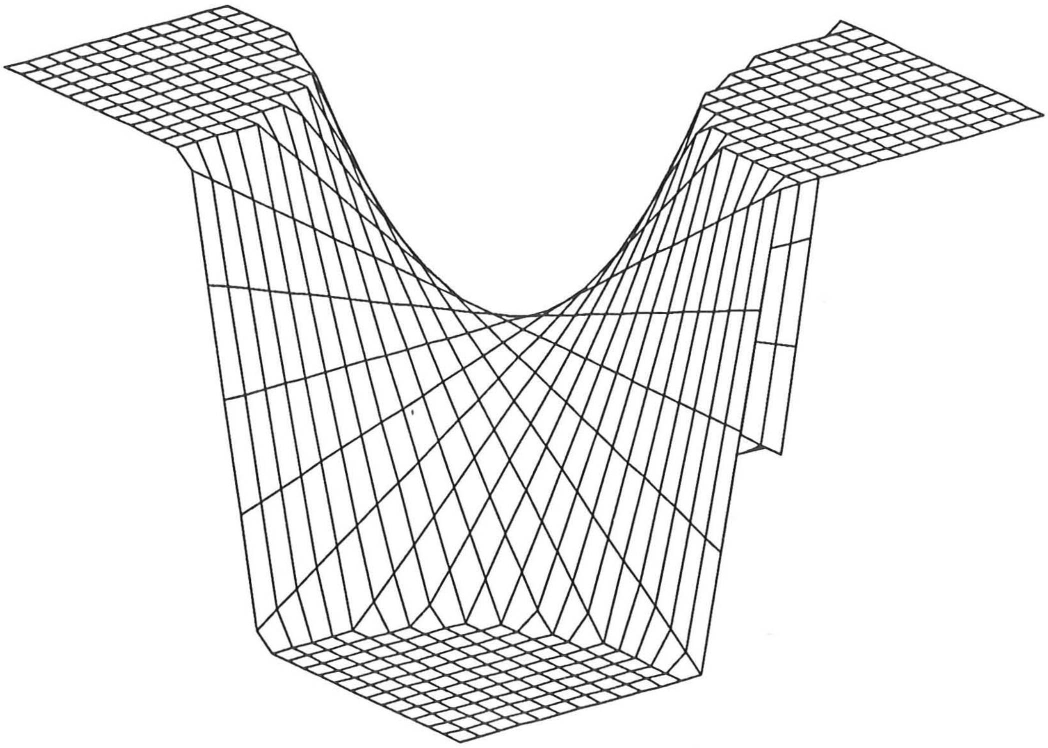


Figure 10.6: Limited Dynamic Range Multiplier.

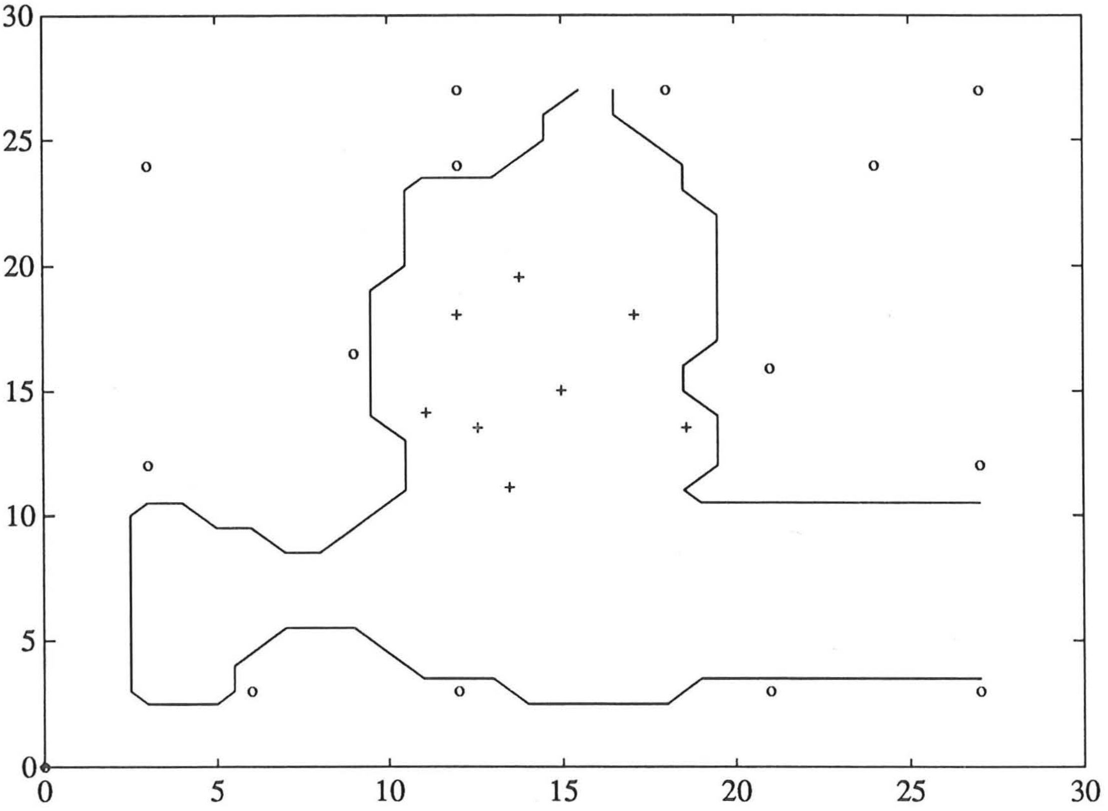


Figure 10.7: Limited Dynamic Range Multiplier Results.

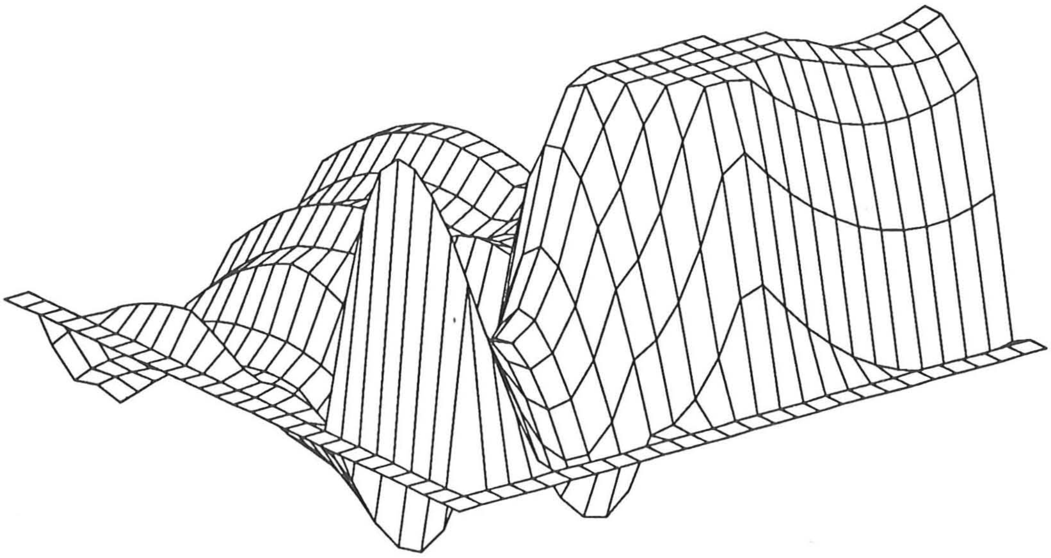


Figure 10.8: Limited Dynamic Range Multiplier Discriminator Surface Results.

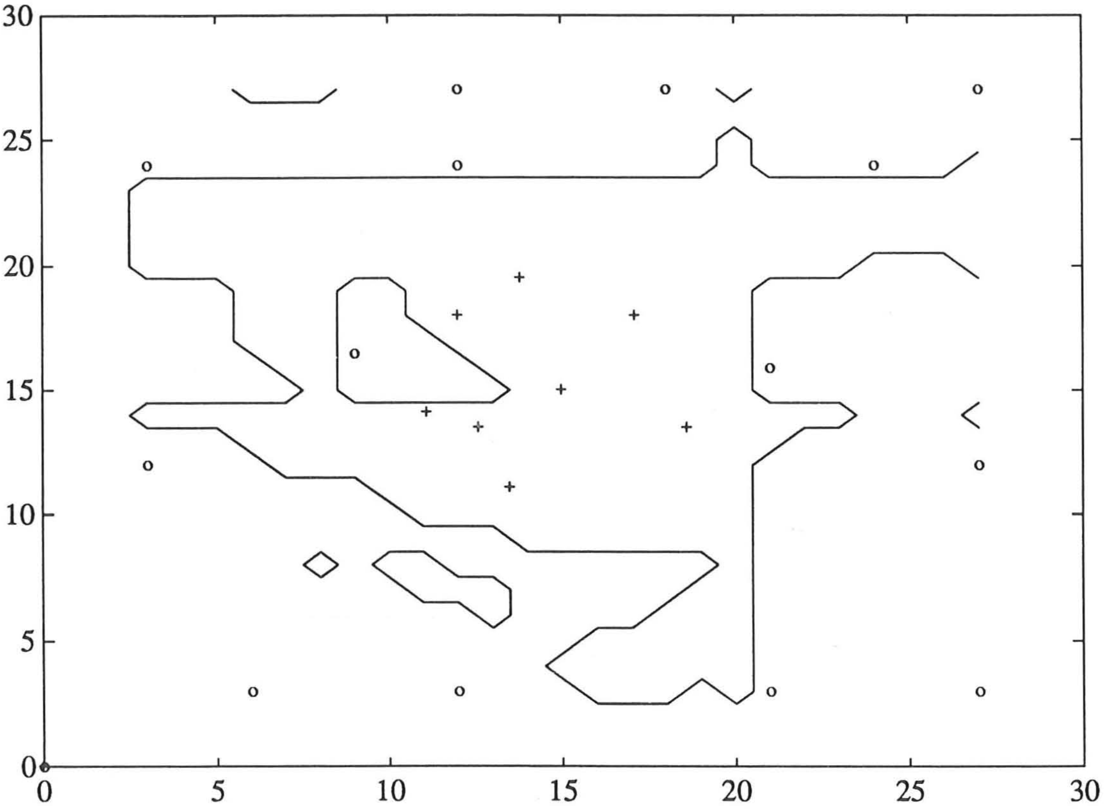


Figure 10.9: Static Noise Corrupted Multiplier Results.

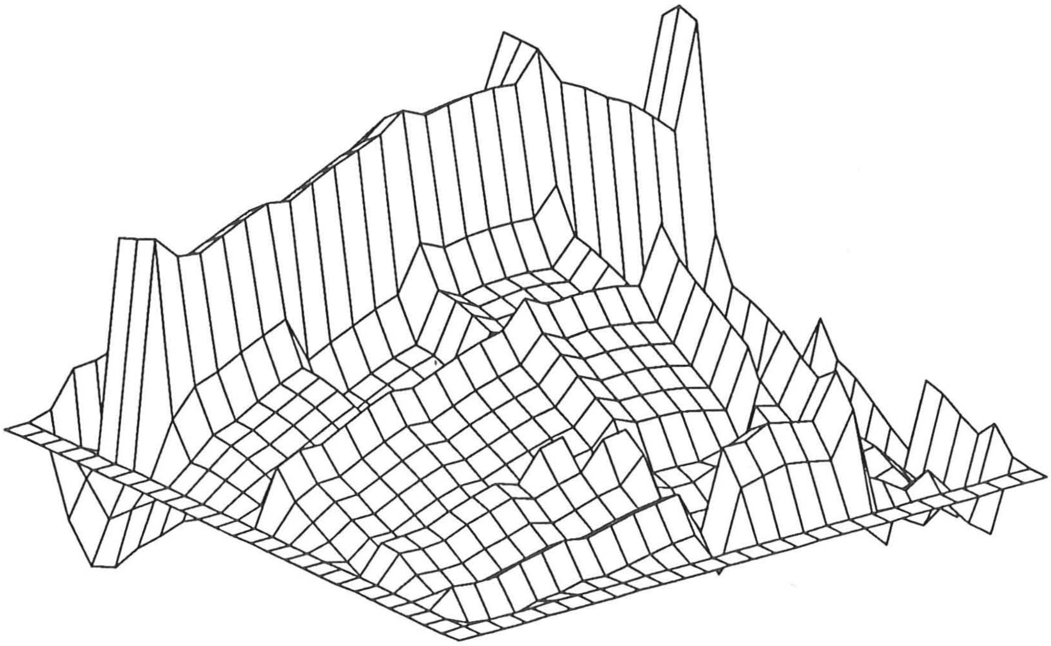


Figure 10.10: Static Noise Corrupted Multiplier Discriminator Surface Results.

7. Fig.10.10:Discriminator surface for static noise corrupted multiplier network.

10.5 Conclusion

The network still has the ability to function for limited dynamic range calculation and for imperfect calculation. Its performance is however in general degraded for the limited range operation. The network will correctly operate with quite severe noise corruptions on its multiplication operations. It may be possible in future work to analyse the effects of such degradations by expansion of the transfer function by power series and replacing each network multiply by a power series expansion of the imperfect multiply. By considering the imperfections in this manner in the light of the network random connection results it is possible to explain why the network can still operate with imperfect multiplies, as these imperfect multiplies can be written as the addition of a series of perfect multiplies. The effect of time varying multiplication noise errors has been left for further work.

Chapter 11

Conclusion and Future Research

11.1 Future Research

The quadratic error surface lends itself ideally to the use of a more efficient adaption algorithm, such as that of conjugate gradients [2]. A variation of this method is currently being developed, which is giving considerably faster convergence times than those discussed earlier without requiring the storage of very large matrices.

A form of the network based on the Wiener G-functionals is being considered, as these form an orthogonal set. This may lead to advantages in adaptive performance for some applications and form another basis from which to analyse the network behaviour.[13] Likewise other non-linear expansions such as the method of gating functions are being considered. This may produce compact networks for problems which contain many saturating elements.[13]

The use of a least squares criterion for multiclass problems is not a particularly good one as it can weight the error likelihood on a class as a function of its output value, so other error functions are being investigated.

Ideally, a pattern recognition system should construct an estimate of the underlying probability density functions which are generating the input vectors given an element from a particular class. Although, due to the expectation operator in the mean square error equation, the probability of a pattern vector occurring is taken into account by the network, it may be possible for the network to make far more complete estimations of underlying probability processes by using Bayesian methods.

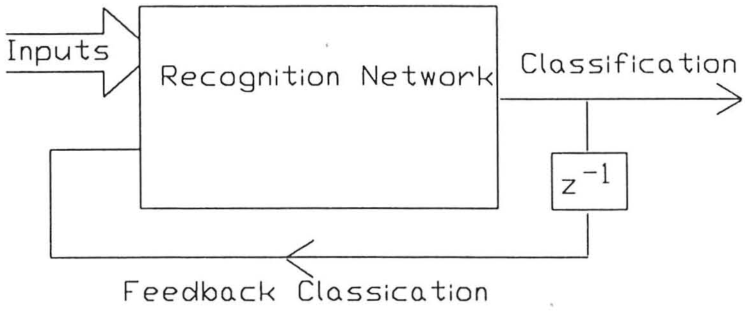


Figure 11.1: Recursive Networks

11.1.1 Time Series Network Incorporation

Many pattern recognition problems require a large amount of contextual information. Such a problem is found in connected speech recognition in which the recognition of a phoneme may require knowledge of immediately previous phonemes. Crude attempts to achieve this have been done by feeding back a previous classification as an input to the network. The fundamental and complex problem of determining the stability behaviour of such non-linear recursive adaptive systems has, however, not been addressed and consequently such recursive networks are not robust.

The analogy of such systems with adaptive IIR filters demonstrates the likely complexity of such stability analyses even before non-linearities are taken into account.

11.1.2 Non-clustering Patterns and Preprocessing

At the moment there are two approaches to the determination of the preprocessor stage in pattern recognition. The first is to assume problem-specific models or problem properties which may be exploited to produce a network input data form that will lead to a sufficiently small number of single class sub-clusters. The second is an ad hoc trial and error approach.

Both of these approaches are problem specific and the first requires knowledge of a suitable

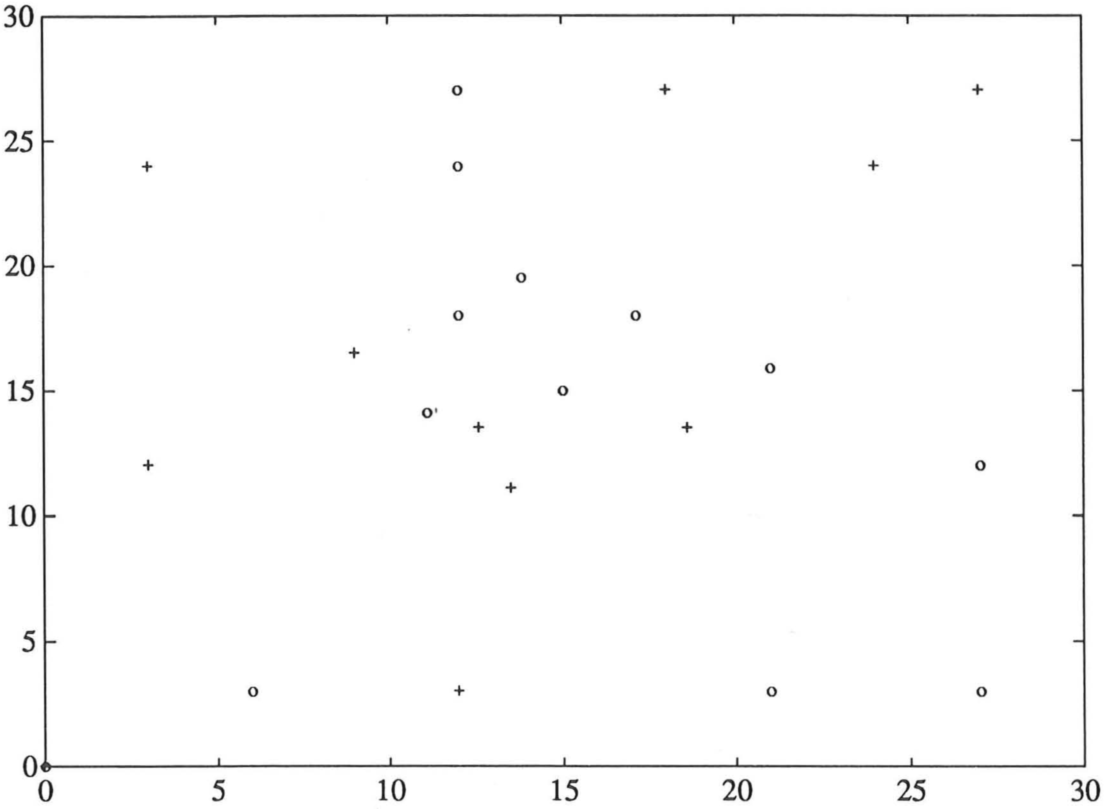


Figure 11.2: Badly Preprocessed Problem.

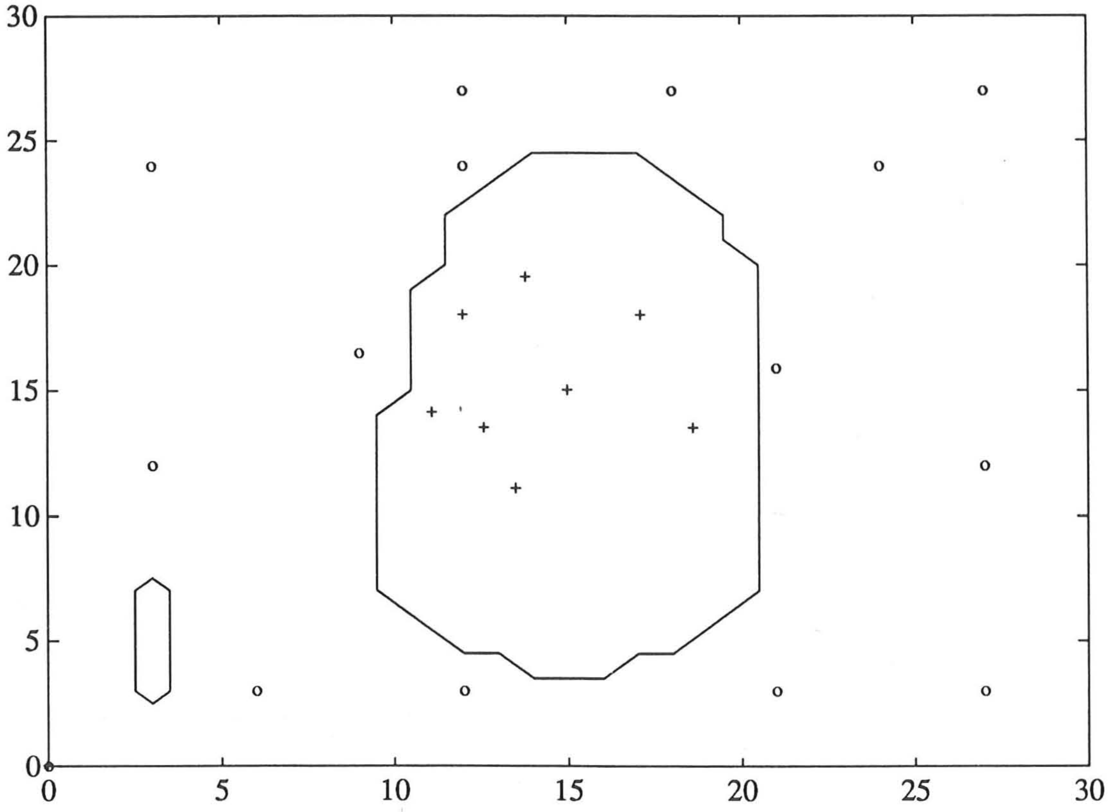


Figure 11.3: Well Preprocessed Problem.

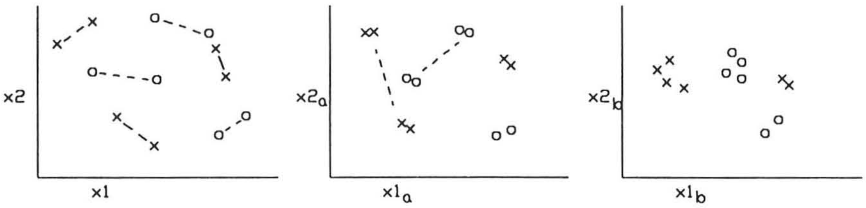


Figure 11.4: Successive clustering.

system model.

A more general approach would be to generate a transformation from the training data that will transform an unpreprocessed problem into a well processed one by combining subclusters belonging to the same classification.

The output mapping work may provide a method to give such a general preprocessor by utilising a space created from the larger eigenvalue eigenvectors of the output mapping analysis.

11.1.3 Image Coding

The image coding problem can be viewed as an attempt to find a low dimensionality signal set that represents a higher dimensionality signal set, the original image. By transmitting the lower dimensionality set, a compression in the amount of data needing to be transmitted over a communication channel may be achieved. The Volterra connectionist model when combined with an optimal output mapping strategy has some interesting properties for use in such a problem, and would allow such a system to incorporate subjective perceptual judgements into the compression process.

11.1.4 Other Data sets

It is hoped to apply the network to other problems such as speech, fingerprint , underwater transient, and more general image recognition tasks. It is also hoped to apply the network to more examples of time series data, especially time series with non-linear statistics.

11.2 Conclusion

The new approaches have been shown to have some advantages over current methods and allow greater understanding of network operation than some of the more empirical methods.

Firstly, the network convergences to its optimal solution orders of magnitude faster than current networks, and this convergence time does not rise as quickly as a function of the size of the pattern vectors.

Secondly, it is unimodal and consequently does not suffer from the problems of local minima, such as needing to be reset, getting stuck or uncertainty about having converged. These problems can seriously affect the performance of current networks and prevent them being used in an adaptive mode while actually performing their tasks.

Thirdly, the complete mathematical derivation of the new network allows a much fuller understanding of its operation and allows direct application of adaptive filter knowledge. This allows easy parameter tuning and application of other adaptive filter update methods. It also allows the effects of approximation and implementation to be considered and the advantages and understanding of aspects such as output index selection to be obtained.

The adaptive network output index and network structure methods allow the application of the networks to problems which previously would have needed impractically large networks.

The work done in this thesis has only begun to probe the possibilities which may be explored by a firm signal processing approach to connectionist modelling and it is hoped that others will continue on this path.

Bibliography

[1] S.Haykin, *Adaptive Filter Theory* Englewood Cliffs, NJ:Prentice-Hall, 1986.

[2] E.Polak, *Computational Methods in Optimisation* London, UK:Academic Press, 1971.

[3] F.Yassa, *Optimality in the Choice of the Convergence Factor for Gradient Based Adaptive Algorithms* IEEE Trans. ASSP, vol. ASSP-35, No.1, Jan. 1987, pp.48-59.

[4] R. Fletcher, *Practical Methods of Optimisation* New York, NY: John Wiley, 1987.

[5] E.Eleftheriou & D.Falconer, *Tracking Properties and Steady-state Performance of RLS Adaptive Algorithms* IEEE Trans. ASSP, vol. 34 ASSP-34, No.5, Oct. 1986, pp.1097-1109.

[6] D.Panda & A.Kak, *Recursive Least Squares Smoothing of Noise in Images* IEEE Trans. ASSP, vol. 25 ASSP-25, No.6, Dec. 1977, pp.520-524.

[7] D. Tufts & R. Kumaresan, *Data Adaptive Signal Estimation by Singular Value Decomposition of Data Matrix* IEEE Proc. Vol. 70 , No.6, Jun. 1982, pp.684-685.

[8] B.Widrow, "Adaptive Signal Processing ", Prentice Hall, New York 1985.

[9] F.M.Reed and P.Feintuch, "Time Delay Estimation Using LMS Adaptive Filters.", IEEE trans, ASSP, Vol.29 No.3 June 1981.

[10] Y.Bressler and S.Merhav, " Recursive Image Registration with Application to Motion Detection", IEEE Trans. ASSP vol.35 No.1 Jan 1987.

[11] S.Alliney and C.Morandi, "Digital Image Registration Using Projections", IEEE Trans. PAMI vol.8 No.2 March 1986.

[12] H.Nagel and W.Enklemann,"An Investigation of the Smoothness Constraint for the Estimation of Displacement Vector Fields from Image Sequences", IEEE Trans. PAMI vol.8 No.5 Sept.86.

[13] Schetzen, *The Volterra and Wiener Theories of Non-linear Systems* New York, NY:John Wiley, 1980.

[14] G.Tesauro & R.Janssens, *Scaling Relationships in Backpropagation Learning* Technical Report:Center for Complex Systems Research Univ. of Illionis. 19-2-1988.

[15] P. Alper, *A Consideration of the Discrete Volterra Series* IEEE Trans. Automatic Control, vol. , Jul. 1965, pp.322-327.

[16] M.J.D.Powell, *Radial basis function approximations to Polynomials* Internal Report . Department of Applied Mathematics and Theoretical Physics. Cabridge University.

[17] D.Spect, *Generation of Polynomial Discriminant Functions for Pattern Recognition* IEEE Trans. EC, vol. EC-16, No.3, Jun. 1967, pp.308-319.

[18] S.T.Alexander, *Transient Weight Misadjustment Properties for the Finite Precision LMS Algorithm* IEEE Trans. ASSP,vol. ASSP-35, No. 9, Sept. 1987, pp.1250-1258.

[19] M.Niranjan & F.Fallside *Neural Networks and Radial basis functions in classifying static speech patterns* Internal Report Cambridge University Engineering Department CUED F-INFENG TR 22(1988).

[20] D. Rumelhart & G. Hinton *Learning Representations by back-propagating errors* Nature Vol.323 9-10-1986.

[21] S.Fakhouri, *Identification of the Volterra Kernels of Non-linear Systems* IEE Proc. vol.127 No.6, Nov. 1980, pp.296-304.

[22] S.Narayan, *Frequency Domain Least-Mean-Squares Algorithm* IEEE Proc. vol.69 No.1, Jan. 1981, pp.125-126.

[23] C.Nikias, *Bispectrum Estimation: A Digital Signal Processing Framework* IEEE Proc. vol.75 No.7, Jul. 1987, pp.869-891.

[24] R.Lippman, *An Introduction to Computing with Neural Nets* IEEE ASSP Magazine. Apr. 1987, pp.4-22.

[25] D Luenberger *Linear and Non-linear Programming* Addison-Wesley, Massachesetts.

[26] C Therrien *Decision Estimation and Classification* John Wiley, New York.

[27] Hand *Discrimination and Classification* John Wiley, New York.

[28] J Glover, *Adaptive Noise Cancelling Applied to Sinusoidal Interference.* IEEE Trans. ASSP vol.ASSP-25 1977, pp.484-491.

[29] P Feintuch, *An adaptive Recursive LMS Filter.* IEEE Proc. vol.64 1976, pp.1622.

[30] B Friedlander, *Analysis and Performance evaluation of an adaptive notch filter.* IEEE Trans. Info. Theory vol.IT-30 1976, pp.283-295.

[31] S Elliot & P Darlington, *Adaptive Cancellation of periodic synchronously sampled interference.* IEEE Trans. ASSP vol.ASSP-33 1985, pp.715-717.

[32] C Johnson, *Adaptive IIR Filtering:Current Results and Open Issues.* IEEE Trans. Info. Theory vol.IT-30 1984, pp.237-250.

[33] W Press et. al *Numerical Recipes:The Art of scientific Computing.* Cambridge University Press 1986.

[34] Rabiner and Gold. *Theory and Application of Digital Signal Processing.* Prentice Hall. 1975.

[35] W Harrison and J Lim *A New Application of Adaptive Noise Cancellation.* IEEE Trans. ASSP vol.ASSP-34 1986, pp.21-27.

[36] S. Farlow, *Self-Organizing Methods in Modeling* Dekker Inc, New York 1984.

[37] T Koh and E Powers, *Second Order Volterra Filtering and Its application to Non-linear System Identification* IEEE Trans. ASSP vol.ASSP-33 1985.

[38] A Invakhnenko, *The Group Method of Data Handling.* Soviet Automatic Control, vol.13 No 12 . 1968.

[39] D Psaltis and C Park, *Non-linear Discriminant Functions and Associative Memories.* American Institute of Physics 1986.

[40] R Gorman and T Sejnowski, *Learned Classification of Sonar Targets Using a Massively Parallel Network.* IEEE Trans. ASSP vol.ASSP-36 1988.

[41] V Klema and A Laub, *The Singular Value Decomposition: Its Computation and some applications.* IEEE Trans. Automatic Control. vol.AC-25 No.2 April 1980.

[42] D S Broomhead and D. Lowe, *Radial Basis Functions, Multivariable Functional Interpolation and Adaptive Networks.* Royal Signals and RADAR establishment Memorandum 4148. IEEE Trans. ASSP vol.ASSP-34 1986, pp.21-27.

[43] J Bunch and C Nielsen, *Updating the Singular Value Decomposition.* Numerische Mathematik 31 1978.

[44] W.S.McCulloch, *A logical Calculus of the ideas imminent in nervous activity.* Bulletin of mathematical biophysics 5, 115-137, 1943.

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

[46] B. Widrow, *Adaptive Switching Circuits.* 1960 IRE WESCON, Conv. record Part 4 96-104 Aug 1960.

[47] J Hopfield, *Computing with Neural Circuits.* Science vol. 233 , 625-673 Aug. 1986.

[48] Moody and Darka, *Learning with localised rceptive fields.* Research report YALEU/DCS/RR-649 Yale Univ.

Published Work

1. M.R.Lynch and P.J.Rayner "A New Approach to Image Registration Utilising Multidimensional Adaptive Filters", Proceedings of the International Conference on Acoustics Speech and Signal Processing (IEEE), New York, USA, April 1988.

2. P.J.Rayner and M.R.Lynch "A New Connectionist Model based on a Non-linear Adaptive Filter", Proceedings of the International Conference on Acoustics Speech and Signal Processing (IEEE), Glasgow, UK, May 1989.

3. M.R.Lynch and P.J.Rayner "Optical Character Recognition using a New Connectionist Model" Proceedings of the International Conference on Image Processing, (IEE) Univ. of Warwick , UK, July 1989.

4. M.R.Lynch and P.J.Rayner "The Properties and Implementation of the Non-Linear Vector Space Connectionist Model ", Proceedings of the First International Conference on Artificial Neural Networks, (IEE) Savoy Place, London UK. Oct 1989.

5. P.J.Rayner and M.R.Lynch "Network Complexity Reduction in Volterra Connectionist Modelling by Consideration of output Mapping and Principal Non-linearities", Proceedings of the International Conference on Acoustics Speech and Signal Processing, Alberquerque, New Mexico, USA, April 1990 .

Internal Report:

6. M.R.Lynch "A Unimodal Neural Network Utilising Non-linear Vector Space Expansion Methods, Comparison with Current Networks and Research Discussion". Cambridge University Engineering Department, Trumpington St., Cambridge.

Yet to be Submitted for publication:

7. M.R.Lynch "An Adaptive Filter Algorithm using Conjugate Gradient Methods."

8. M.R.Lynch "An Adaptive Pitch Estimator for LMS Notch Filters".

PhD. Thesis:

9. "Adaptive techniques in Signal Processing and Connectionist Models."

A New Approach to Image Registration Utilising Multidimensional LMS Adaptive Filters.

By
Michael R. Lynch and Peter J.W.Rayner
Engineering Department, University of Cambridge
Cambridge,England.

Abstract

In this paper a new method is presented for the adaptive registration of two images between which the image object is shifted. The method is based on multidimensional LMS adaptive filters, whose derivation is first given and the performance surface is shown to be unimodal.The multidimensional filter's similarities and differences with the one-dimensional LMS adaptive filter are briefly discussed. It is shown that the filter will perform the necessary operations of correlation, interpolation,filtering and shift to produce a registered output image. An algorithm is also developed to work in conjunction with the filter to provide explict shift measurements. The advantages of this inherently parrallel adaptive approach over current methods are discussed and are most apparent where the images are corrupted by noise or are related by spatially varying shifts. Finally the possibility of application of multidimensional LMS adaptive filters to pattern recognition is briefly mentioned.

Introduction

Examples of the need for application of image registration algorithms arise in many fields such as airborne ground sensing and aircraft navigation, medical imaging, in which separate frames of an image must be registered in the presence of involuntary patient movement and where ever there is relative movement between the image sensor and the image scene. In practical cases the images may be degraded by noise, may contain weakly spatially varying shifts or one of the two images to be registered may contain objects not found in the other or moving independently of the rest of the scene. All of these effects can significantly reduce the performance of current image registration methods such as phase correlation.

In the case of images this method employs,a two dimensional Least Mean Squares adaptive filter which makes almost no a priori assumptions about the images or their degredations. This allows the method to be used with general images and consequently it is not limited to images with sharp edges or similar artifacts as may be required by syntactic methods.

Derivation

The derivation of the mutildimensional LMS adaptive filter is closely related to that of the one dimensional case. The derivation is given for a N dimensional adaptive filter with each dimension of order m.

Consider an input tensor of rank N order m **X** and a N rank weight tensor **W**.Both tensors elements have N indices $i \ldots z$.

The N-dimensional adaptive linear combiner is given by:

$$y_k = \mathbf{W_k X_k}$$

Where the operation **WX** is defined as the inner product, that is:

$$\sum_{i=1,m} \sum_{j=1,m} \cdots \sum_{z=1,m} x_{i \ldots z} w_{i \ldots z}$$

The error may be defined as the difference between a desired output d_k and the actual output y_k.

$$\epsilon_k = d_k - y_k$$

$$\epsilon_k = d_k - \mathbf{W_k X_k}$$

Hence:

$$\epsilon_k^2 = d_k^2 - (\mathbf{W X_k})(\mathbf{W X_k}) - 2 d_k \mathbf{W X_k}$$

We now assume that ϵ_k, d_k and $\mathbf{X_k}$ are statistically stationary and consider the expectation of the error squared over k.This assumption can have some important implications for work with practical images as in many cases the degree of stationarity can be less than is normally found with many practical one dimensional signals.

$$E(\epsilon_K^2) = E(d_K^2) + \mathbf{W_K} E(\mathbf{X_k X_K})\mathbf{W} - 2E(d_k \mathbf{X_K})\mathbf{W}$$

We can define $E(\mathbf{X_k X_k})$ as the N dimensional correlation 'Tensor' of rank 2N, **R**. Likewise the cross-correlation tensor $E(d_k \mathbf{X_K})$ denoted, **P**.

$$E(\epsilon_k^2) = E(d_k^2) + \mathbf{W_k R W_k} + 2\mathbf{P}\omega$$

This shows that the mean square error is precisely a quadratic function of the weight values. Thus the error surface is unimodal with no local minima. It is this quality that will allow some form of gradient search to converge to the optimum solution.

Having shown the nature of the error or *Performance* surface the problem is now to find a search algorithm to find the minimum. Again the cue is taken from the 1D case of the derivation of the Widroff-Hoff LMS algorithm [1] and a simple gradient search is employed

An adaptive algorithm may be formed by the following approach:

$$\nabla_k = \frac{\partial \epsilon_k^2}{\partial \mathbf{W}} = 2\epsilon_k \frac{\partial \epsilon_k}{\partial \mathbf{W}} = -2\epsilon_k \mathbf{X}$$

We can specify a steepest descent algorithm:

$$\begin{aligned} \mathbf{W_{k+1}} &= \mathbf{W_k} - \mu \nabla_k \\ &= \mathbf{W_k} + 2\mu\epsilon_k \mathbf{X_k} \end{aligned}$$

Where μ is the adaption coefficient. Explicitly in the case of 2D ADFs the update equation is:

$$W_{xy(k+1)} = W_{xyk} + 2\mu\epsilon_k X_{xyk}$$

The derivation of the LMS adaptive filter is independent of the dimensionality of the data, thus the same derivation can be used to lead to 2D,3D or ND ADFs. This derivation also shows us that the error surface is unimodal even for higher dimensionality data. The fact that 2D and 1D ADFs are based on the same theory also allows us to predict with reasonable confidence the behaviour of 2D ADFs by analogy with the 1D case, however care must be exercised in this practice as some of the problems encountered in 2D have no analogy in 1D, also the nature of practical image data may differ from practical 1d signals.

Image Resistration System

The basic shift registration system is shown in figure one. The reference image may in practice often be one of the previous frames of a series of image frames. The adaptive filter is run and its coefficients are updated. The filter will perform a gradient search to minimise the mean square error image. In the case of a shift this will occur when the output image has been correctly shifted by the filter to be registered with the reference image. The maximum shift that the system can cope with is given by the filter size and consequently its maximum spatial delay. The system is arranged so that there is in effect a spatial shift of half of the filter size between the two images so that the filter can cope with a positive or negative delay of magnitude less than m/2.

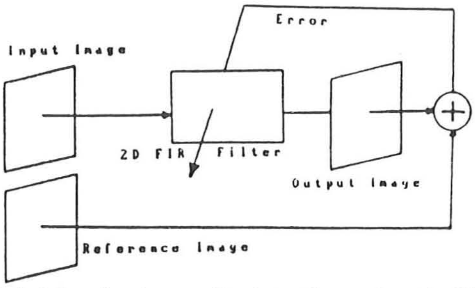


Fig1.Adaptive Image Registration using 2D Filters.

The particular path over the image that the filter runs can be important in the case of spatially varying shifts.If the variation is occurring more quickly in one of the two axis directions it would be advisable to track the filter along the other axis direction to allow the filter more iterations in which to update to the new shift. Consequently if images have a spatial variation which is mainly a function of the x coordinate it would be advisable to run the filter down the lines of constant y coordinate only advancing in x coordinate at the end of each y line.

It is possible to improve the performance of a given order of filter by ensuring that there is no large relative dc offset between the images. This may occur if the two images were captured under very different illumination levels. Although it may be expected that the filter could compensate for such an offset the process of doing this degrades the performance of the system as it is utilising zeroes of the filter that may be more usefully applied elsewhere. Correspondinly it is therefore often useful to high pass filter the images before registration if there is a possibility of large dc offsets.

The Adaption Coefficient

The value of the adaption coefficient is important in determining the mode of operation of the system. As a generalisation the nearer the adaption coefficient approches the divergence value so the output image becomes sharper and the system can track spatially varying parameters more rapidly. However smaller values of the adaption coefficient give higher noise immunity. One of the main problems in the practical implementation of adaptive image registration is the choice of a suitable adaption coefficient. In fact because of the nonstationary statistics of practical images it is necessary, in order to improve performance, to implement a self regulating adaption coefficient which varies with the image. A simple, yet reasonably effective approach is to update the coefficient of adaption as a function of the input power to the filter. This approach which is based on the work of Yassa [6] has been simplified to reduce computational load. A power estimate is obtained from the row of unseen data about to enter the filter, this is then filtered by a single pole filter whose feedback factor can be adjusted to control the smoothing of the power estimate.

Incident power:

$$P_k = \sum_{i=1,m} x_{i,1}^2$$

Smoothed power:$F_k = F_{k-1}g + P_k$

Where g is the feedback factor. The adaption coefficient is updated by:

$$\mu = \frac{\mu_{int}}{F_k}$$

Where μ_{int} is the initial adaption coefficient.

If it is desired to use the filter output image directly or to track spatially varying shifts a larger adaption coefficient may be needed. Even with the above adaption coefficient updator divergence may still occur on certain parts of some images so it is necessary to have a divergence detector. This may be done by checking that the output pixels do not execeed a certain value. If this should occur the μ_{int} value is immediately reduced by a factor (usually about 0.7) and the filter coefficient reset to either all zero, or all zero with a value one at the coefficient corresponding to a recent shift estimate.

Filter Solutions

As may be expected from the one dimensional analogy the performance of the two dimensional LMS algorithm depends on the eigenvalue spread of the input data and the time taken to converge to the correct shift estimate is reduced as the whiteness of the data increases. In order to extract the shift estimate explicitly from the coefficents of the adapted filter we must consider the filter solution. In the case of white input signals and integer shifts the filter converges to a Kronecker delta function at the shift delay as may be expected, and for non-integer shifts to an interpolator. Almost all practical images do converge to solutions with peaks at the delay value. The dominance of this peak is however determined by the whiteness of the input image, whiter images giving a more dominant peak.

A simple peak detector my be used to extract the shift [2] and interpolation of the peak may be used to obtain non-integer shift estimates. By using such a shift extractor at regular intervals a displacement vector map be built up showing the displacement vector over the image.

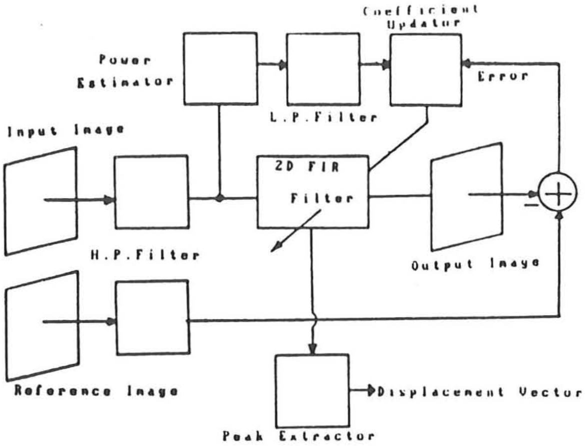


Fig 2:The Complete Registration System

Output Image

As a biproduct of the processing, a registered and shifted corrected image is produced at the filter output. However, apparently due to the sensitivity of the LMS algorithm to eigenvalue spread and the differening convergence rates for different eigenvalues, the solutions reached are near optimal but lead to subjectively slightly blurred images. These problems may be alleviated by using adaption coefficient values near to divergence, but this can lead to a risk of of divergence occurring. Another approach is to use the displacement vector field to 'manually' shift back the original image. This is currently done with many registration methods.

Performance

The filters were run on clean images with a spatially stationary shift. In practice a lock onto the correct shift is achieved in thirty or forty pixels. The adaption coefficient value may be small such that the output image is very blurred but yet give good results for the displacement vector. All the results shown were generated using images 80 by 80 pixels which were quantised to 6 bits.

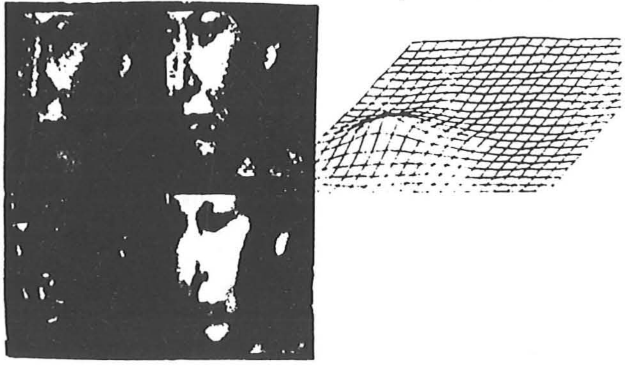


Fig 3. Results of Constant shift Case. Filter 20 by 20.Shift (5,4), and μ_{int} = .008 Top left Reference,Top right Input, Bottom Left Error squared, Bottom right Output. Also Plot of final coefficient values.

The same experiment was repeated with the addition of equal levels of white uncorrelated noise to each of the images and the displacement vector field recorded. As may be expected the filters took longer to lock onto the shift, and in the case of high noise levels, the lock could be lost and regained throughout the picture. However even at these high noise levels further increases in performance could be obtained by the utilisation of averaging of the displacement vector by use of the property of continuity of the displacement vector [7].

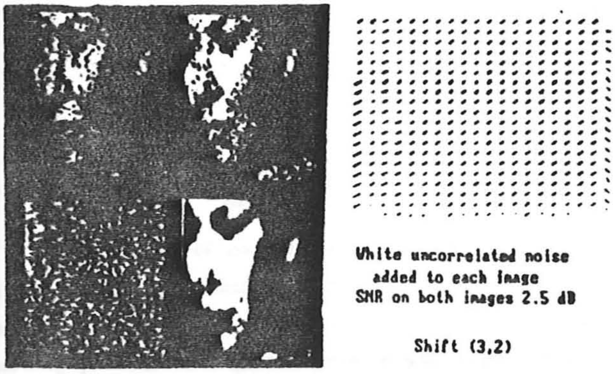


Fig 4. Results with addition of white noise to both Images.Filter 20 by 20,μ_{int} = .004 Format as Fig3 Also Displacement vector map.

Finally the filters were run on an image pair in which the filter input image had been obtained by a spatial varying shift which was a function of the x coordinate.

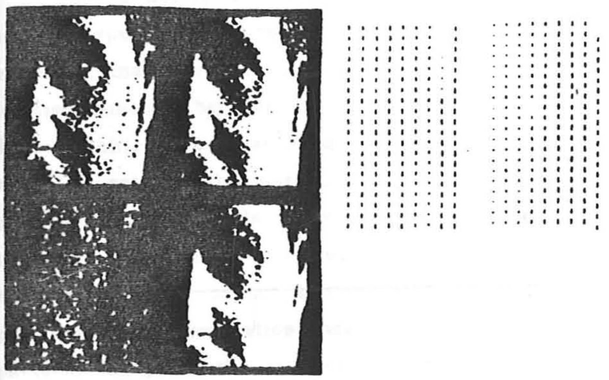


Fig 5. Results of Spatially varying shift case.Shift $\delta y = 5 * \sin(0.04 * x)$. Filter 12 by 12 μ_{int} = .004

Computational Load

For spatially stationary shifts the filter need only be run over small portions of the image. The filter size is set by the largest shift to be dealt with, but this may be reduced by calculating displacement vector fields from a decimated image. The algorithm is inherently parrallel and suited to the use of current adaptive digital filter integrated circuits.

Related Possibilities

The above registration system may be adapted for pattern recognition by using a reference image of an object as the desired input and a scene as the filter input. The filter correlates the object in the scene and reference and shifts the scene. Thus the error image can be used as a measure of the match over the relevant region, thus providing a shift invariant pattern recognition system. The use of higher order,for example 3D, adaptive filters may be utilised to register over a sequence of frames by the same principle as is outlined above. The similarity of such higher order adaptive filters to neural networks has been noted.It is hoped that the insight given by the description of mutlidimensional adaptive filters may lead to a fuller understanding of the behaviour of some classes of neural networks.

Conclusion

The LMS multidimensional adaptive filter can provide a new approach to image registration which is well suited to parralel implementations. The system also has some useful properties giving the noise immuninty of correlation based methods but not requiring a search over possible, shifts.It also the ability to deal with weakly spatially varying shifts unlike transform based methods. The adaptive filter can also handle weak image degredations as its frequency response is not constrained and may adapt to correct differences in the two images caused by sensors or uneven illumination and so on.

References

[1] B.Widrow, *et al.* "Adaptive Signal Processing ", Prentice Hall, New York 1985.

[2] F.M.Reed and P.Feintuch, "Time Delay Estimation Using LMS Adaptive Filters.", IEEE trans, ASSP, Vol.29 No.3 June 1981.

[3] Y.Bressler and S.Merhav, " Recursive Image Registration with Application to Motion Detection", IEEE Trans. ASSP vol.35 No.1 Jan 1987.

[4] S.Alliney and C.Morandi, "Digital Image Registration Using Projections", IEEE Trans. PAMI vol.8 No.2 March 1986.

[5] S.Haykin, "Adaptive Filter Theory", Prentice Hall, New York, 1986.

[6] F.Yassa, "Optimality in the choice of Convergence factor for Gradient Based Adaptive Algorithms",IEEE Trans.ASSP vol.35 No.1 Jan87.

[7] H.Nagel and W.Enklemann,"An Investigation of the Smoothness Constraint for the Estimation of Displacement Vector Fields from Image Sequences",IEEE Trans. PAMI vol.8 No.5 Sept.86.

A New Connectionist Model Based on a Non-Linear Adaptive Filter

Peter J. W. Rayner and Michael R. Lynch

Department of Engineering
Cambridge University, Cambridge, U.K.

ABSTRACT

A new type of connectionist model is introduced based on the non-linear extension of adaptive filter theory. It is shown that the model converges in the learning process to a global optimum and experimental results are presented which indicate that the rate of convergence is considerably faster than has been reported for other models.

1. Introduction

Connectionist models are being studied in many areas of speech and pattern recognition [1,2]. The basic operation that the model is required to realise is that of discriminating between various classes of patterns. This is achieved by presenting the connectionist model with a set of training patterns and associated classification indices. In the supervised learning phase the model parameters are automatically adjusted so that the network output approximates, as closely as possible, the classification index associated with the pattern. Many of the connectionist models currently being investigated are based on approximations gained from physiological studies of neural behaviour. However these networks are inherently non-linear and little is understood about the optimality of the network and the rate of convergence in the learning process. Indeed it is known that the networks exhibit many local minima and that the rate of convergence can be excessive [3]

In the supervised training phase, let a training pattern be represented by the vector $\underline{x}$. The connectionist model operation may be written as a general non-linear function $h(\underline{x})$ of the pattern vector $\underline{x}$. Associated with the pattern vector $\underline{x}$ is a classification index c_j which indicates to which of M sets of patterns Ω_j the pattern vector belongs. If the classification indices are ordered so that:

$$c_1 < c_2 < c_3 < \cdots < c_{M-1} < c_M ,$$

then the operation of the supervised learning phase is to adjust the parameters of the model function $h(\underline{x})$ so that:

$$c_i \leq h(\underline{x}) < c_{i+1} \Rightarrow \underline{x} \in \Omega_i \qquad \ldots(1)$$

2. Discriminant Functions

The operation described by eqn. 1 is known as a Discriminant function [4] and it is the purpose of this section to examine some of the basic properties of these functions.

Consider first the case of a linear discriminant function where the general scalar function $h(\underline{x})$ of the N-dimensional pattern vector $\underline{x}$ is linear. This may be written as:

$$y = h(\underline{x}) = \underline{w}^T \underline{x}$$

$$y = \begin{bmatrix} w_0 & w_1 & w_2 & \cdots & w_N \end{bmatrix} \begin{bmatrix} 1 \\ x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix}$$

The constraint of linearity implies that the dicriminator boundaries are straight lines in 2-D space or, more generally, hyper-planes in multi-dimensional space.

In general it is not possible to discriminate between pattern classes with a linear discriminant function as can be seen by considering the n-bit parity problem which is often used as a test case for connectionist models.The 2-bit parity problem may be stated as finding a discriminant function which satisfies the following conditions.

$$y = \underline{w}^T \underline{x} \geq c \Rightarrow \underline{x} \in \Omega_1$$
$$y = \underline{w}^T \underline{x} < c \Rightarrow \underline{x} \in \Omega_2$$

where:

$$\underline{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

$$x_1 , x_2 \in \{ -1 , +1 \}$$
$$\Omega_1 = \{ \underline{x}^T \mid (1 , -1) , (+1 , +1)\}$$
$$\Omega_2 = \{ \underline{x}^T \mid (-1 , +1) , (+1 , -1)\}$$

Typical discriminator boundaries are shown in fig. 1.

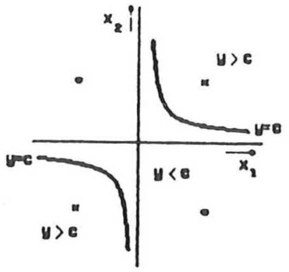


Fig. 1. Discriminator boundaries for 2-bit parity problem

In algebraic terms the following examples would serve as discriminators for the 2-bit parity problem.

$$y = x_1 \, x_2$$

gives: $y > 0 \, , \underline{x} \in \Omega_1 = \{\text{Even Parity}\}$

$y < 0 \, , \underline{x} \in \Omega_2 = \{\text{Odd Parity}\}$

$$y = (x_1 + x_2)^2 - 2$$

$$y = -2 + x_1^2 + 2x_1x_2 + x_2^2$$

gives: $y > 0 \, , \underline{x} \in \Omega_1 = \{\text{Even Parity}\}$

$y < 0 \, , \underline{x} \in \Omega_2 = \{\text{Odd Parity}\}$

The general discriminator may be considered as a non-linear operator on the elements of the pattern vector. However an alternative viewpoint is to consider the general discriminator as a Linear operator over a Non-linearly Extended Vector Space. This approach gives considerable insight and leads to a Connectionist Model with desirable properties.

For the 2-bit parity problem the discriminator:

$$y = x_1 \, x_2$$

may be considered as the linear discriminator:

$$y = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_1x_2 \end{bmatrix}$$

operating over the non-linear space shown in fig. 2.

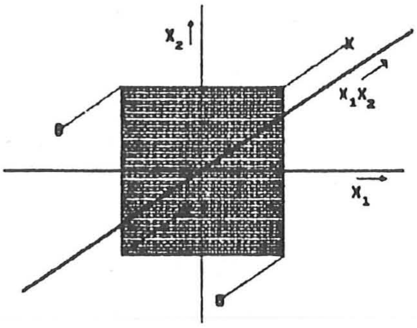


Fig. 2. Discriminator boundaries in extended vector space for 2-bit parityproblem

3. Form of the Non-linear Extension

There are a number of possible formulations for the non-linear extension vector but one which is closely related to the Volterra expansion for a non-linear dynamic system, is the discrete Kolmogorov-Gabor polynomial [5]

$$y = \underline{w}^T \, \underline{x}_o$$

$$= w_0 + \sum_{i_1=1}^{N} w_{i_1} x_{i_1} + \sum_{i_1=1}^{N} \sum_{i_2=1}^{N} w_{i_1 i_2} \, x_{i_1} \, x_{i_2} + \cdots$$

$$\cdots + \sum_{i_1=1}^{N} \sum_{i_2=1}^{N} \cdots \sum_{i_K=1}^{N} w_{i_1 i_2 \cdots i_K} \, x_{i_1} \, x_{i_2} \cdots x_{i_K} \qquad \ldots (2)$$

This will be termed an (N, K) discriminator system. ie. N variables and upto K^th. order product terms. An N variable linear discriminator is an (N, 1) system.

The form of the Extension Vector for a (3,2) system is:

$$\underline{x}_o^T =$$

$$\begin{bmatrix} 1 & x_1 & x_2 & x_3 & x_1^2 & x_1x_2 & x_1x_3 & x_2x_1 & x_2^2 & x_2x_3 & x_3x_1 & x_3x_2 & x_3^2 \end{bmatrix}$$

4. Optimum Discriminator Weight Vector

The optimum discriminator weight vector $\underline{w}$ may be determined with reference to fig.3.

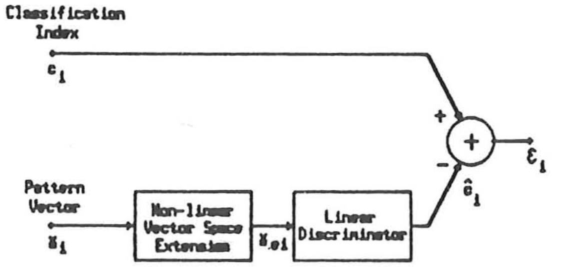


Fig.3. Extended vector space linear discriminator

The error in the discriminator output for an input pattern $\underline{x}_i$ and associated classification index $c_j(i)$ is

$$\epsilon_i = c_j(i) - \hat{c}_j(i)$$

$$= c_i(i) - \underline{w}^T \, \underline{x}_{oi} \qquad \ldots (3)$$

Expected squared error $E\{ \epsilon_i^2 \}$

$$= E\{ (c_j(i) - \underline{w}^T \, \underline{x}_{oi})^2 \} \qquad \ldots (4)$$

Minimising with respect to the weight vector $\underline{w}$:

$$\underline{w}^T = \underline{R}_{cx}^T \, \underline{R}_{xx}^{-1} \qquad \ldots (5)$$

where: $\underline{R}_{cx}^T = E\{c_j(i) \, \underline{x}_{oi}^T \}$

$$\underline{R}_{xx}^{-1} = E\{ \underline{x}_{oi} \, \underline{x}_{oi}^T \}$$

In practical applications the dimensions of the correlation matrix $\underline{R}_{xx}$ may mean that the computation of the inverse is not practicable. Moreover some applications of Connectionist Models require that the model be continually re-trained as the pattern data varies. In signal processing terms this is equivalent to dealing with non-stationary signals. It is necessary, therefore, to seek an alternative approach. The form of the linear discriminator is identical to that of a Finite Impulse Response filter and eqn.5 is, in effect, the classical Wiener filter but defined over a non-linearly extended vector space. There has been a considerable amount of work over the past few years dealing with adaptive approaches to determining optimal filters [6] and most of this work may be applied to the extended vector space system. The simplest of the algorithms wil be derived; this is known as the Least Mean Square (LMS) algorithm or the Stochastic Gradient algorithm.

The block diagram of an adaptive approach to the discriminator problem is shown in fig. 4.

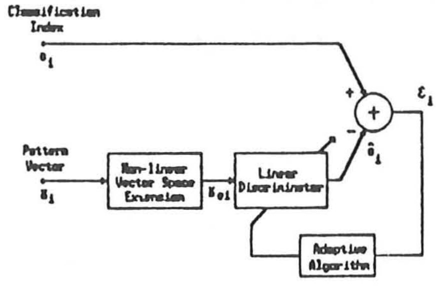


Fig.4. Adaptive extended vector space linear discriminator

The principle of the LMS algorithm is to update the discriminator weight vector in the direction of steepest descent down the error surface.

The update equation is:

$$\underline{w}_{i+1} = \underline{w}_i + \mu \frac{\partial E\{ \epsilon_i^2 \}}{\partial \underline{w}_i}$$

where μ is a scalar parameter which determines the rate of convergence.

$$\underline{w}_{i+1} = \underline{w}_i + E\{2 \mu \epsilon_i \frac{\partial \epsilon_i}{\partial \underline{w}_i} \}$$

From equation 3,

$$\frac{\partial \epsilon_i}{\partial \underline{w}_i} = -\underline{x}_{ei}$$

$$\therefore \underline{w}_{i+1} = \underline{w}_i - 2 \mu E\{ \epsilon_i \underline{x}_{ei} \} \qquad \ldots (6)$$

In the LMS algorithm the expected value of the error gradient is approximated by the instantaneous value so that the update equation becomes:

$$\underline{w}_{i+1} = \underline{w}_i - 2 \mu \epsilon_i \underline{x}_{ei}$$

Thus the operation of the system in the supervised learning phase is to present a sequence of pattern vectors $\underline{x}_i$ and associated classification indices $c_j(i)$. The discriminator weight vector is updated in accordance with eqn.6 as each pattern and index is presented.

A crucial point is brought out in eqn.4 which shows that the error surface is hyper-paraboloid and so has a single global minimum. This in sharp contrast to the more standard Connectionist Models in which the shape of the error surface is not known and it is not possible, therefore, to determine whether the model has converged to a local or global minimum. A further point of note is that the learning or adaptive phase of the network has been described in terms of the LMS algorithm but there are a number of alternative algorithms, such as the Recursive Least Squares or Deterministic Kalman Filter, which exhibit considerably faster convergence but at the expense of additional computational complexity.

5. Network Structure

The realisation of the proposed Connectionist Model has not been considered in any detail but it is clear that there is considerable structure in the non-linear formulation given by eqn.2. One particular decomposition will be demonstrated. Eqn.2 may be rewritten as:

$$y = \sum_{i_1=0}^{N} \sum_{i_2=0}^{N-i_1} \cdots\cdots\cdots \sum_{i_K=0}^{N-i_1-i_2-\cdots-i_{K-1}} w_{i_1 i_2 \cdots i_K} x_1^{i_1} x_2^{i_2} \cdots x_K^{i_K}$$

which may be further expressed as:

$$y = \sum_{i_1=0}^{N} x_1^{i_1} \sum_{i_2=0}^{N-i_1} x_2^{i_2} \cdots\cdots\cdots \sum_{i_K=0}^{N-i_1-i_2-\cdots-i_{K-1}} w_{i_1 i_2 \cdots i_K} x_K^{i_K}$$

The above equation may be computed using a nested poloynomial approach as shown below for a (2,3) system.

$$\begin{aligned} y = (&w_{00} + w_{01}x_2 + w_{02}x_2^2 + w_{03}x_2^3) \\ &+ x_1 (w_{10} + w_{11}x_2 + w_{12}x_2^2) \\ &+ x_1^2 (w_{10} + w_{11}x_2) \\ &+ x_1^3 (w_{10}) \end{aligned}$$

and in nested form:

$$y = p_0 + x_1(p_1 + x_1(p_2 + x_1(p_3)))$$

where:

$$p_0 = w_{00} + x_2(w_{01} + x_2(w_{02} + x_2(w_{03})))$$

$$p_1 = w_{10} + x_2(w_{11} + x_2(w_{12}))$$

$$p_2 = w_{20} + x_2(w_{21})$$

$$p_3 = w_{30} + x_2(w_{31})$$

6. Experimental Results

In order to compare quantitatively the performance of the extended vector space system with current connectionist models, the network was trained to perform the n-bit parity recognition problem . This problem is one of the few for which there are sets of quantitative convergence time data for current connectionist models and demonstrates the ability of the network to perform nonlinear decision making processes.The network is presented with an n bit binary word and is required to output the binary parity value of the word.The nonlinear extended space network was implemented using the LMS adaptive algorithm on the extension space generated by the Kolmogorov-Gabor polynomial given by equation 2. This network is compared with a standard hidden layer back propogation neural network adapted with the Back Propagation algorithm [2]. Such networks are among the most powerful and widely used current networks.The figures for the convergence times of the back propogation network are taken from a recent study of network convergence times on the n-bit parity problem [3]. The back progation network used in this study was composed of n input units, 2n hidden units and one output unit with full connectivity. The problem of choosing the network parameters was addressed by running the networks many times for each n with differing network parameters and taking the best run. The problem of back propogation nets converging on local minima was dealt with by discarding from the results any run that did not appear to beconverging. In the case of the extended space network the problem of parameter choice did not arise as the only parameter, the optimal adaption coefficient μ, can be chosen using methods standard to adaptive filter theory. A simple and usually adequate method is that of Yassa [7]. Thus only one run was needed for each value of n. The problem of local minima for the extended space network does not arise as the system is unimodal so that the system always converges to the optimal solution. Table 1 compares the convergence times T for the proposed connectionist model with the convergence times T_{BP} for the Back Projection algorithm as a function of the number of bits n.

No.bits	Convergence Time	
n	T	T_{BP}
2	15	95
3	17	265
4	62	1200
5	106	4100
6	292	20000
7	558	100000
8	1287	500000

Table 1. Comparison of convergence times

7. Conclusions

The extended space approach leads to networks which can, with sufficient extension, synthesise any nonlinear discriminant function whilst maintaining unimodality. This leads to general networks which are factors of between one hundred and one thousand times faster in convergence in the experimental investigations performed. The property of unimodality means that there are no local minima and the network always converges to the global optimum. The approach has a firm analytical basis and large body of supporting knowledge in adaptive filter theory. The adaptive filter knowledge also allows analytical solutions to the vital question of parameter tuning thus allowing good first run performance on practical data.The supporting knowledge from adaptive filter theory may also be used to predict the effects of eigenvalue spread and rank defficiency in the correlation matrix in equation 5, or to utilise other more powerful adaption methods such as Recursive Least Squares, Singular Value Decomposition, and their derivatives.

References

[1] D. Rummelhart, J. L. McClelland & the PDP Research Group, "Parallel Distributed Processing: Explorations in the Microstructure of Cognition," Vol. 1, MIT Press, Cambridge, Massachusetts. 1986

[2] D. E. Rummelhart, G. Hinton & R. Williams, "Learning representations by back propagating errors," Nature, Vol. 323, pp. 533-536.

[3] G. Tesauro & R. Janssens, "Scaling relationship in back-propagation learning: dependence on predicate order," Technical Report, Center for Complex Systems Research, University of Illinois at Urbana-Champaign, 1988

[4] D. J. Hand, "Discrimination and Classification," John Wiley & Sons, Chichester, 1981

[5] A. G. Ivakhnenko, "Heuristic self-organisation in problems of engineering cybernetics," Avtomatika ,Vol. 6, pp. 207-219, 1970

[6] S. S. Haykin, "Adaptive Filter Theory," Prentice-Hall, New Jersey, 1986.

[7] F. Yassa, "Optimality in the choice of convergence factor for gradient-based adaptive algorithms," IEEE Transactions Acoustics, Speech and Signal Processing, ASSP-35, No.1, pp. 48-59, 1987.

Optical Character Recognition using a New Connectionist Model

M.R.Lynch and P.J.Rayner
Cambridge University Engineering Department, U.K.

1 Abstract

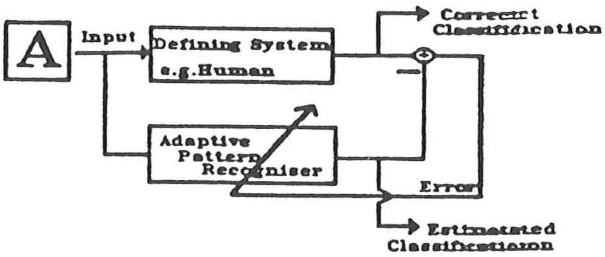
This paper describes the developement of a system for recognising hand drawn capital letters, which is designed to be position, scale and rotation invariant. The object of the paper is to investigate the performance of a new unimodal form of connectionist model when coupled to a preprocessing stage. The new connectionist model is briefly derived and its error surface is shown to be unimodal.

The new connectionist model may be considered as being composed of a non-linear vector space expander followed by a linear adaptive filter operating in the extended space. The vector space expansion may be performed by means of the Volterra series. This approach has the advantage of allowing the direct application of adaptive filter theory.

1.1 Non-linear System Modeling Approach

The pattern recognition problem may be thought of as a system modeling problem, in which the adaptive pattern recogniser is attempting to model the classification system; for example, a human defining the relationship between a set of character font images and a set of collating sequence numbers. The block diagram showing the recognition problem cast as a system modeling problem is shown in Fig.1.

Fig.1



It is clear that the relationship between the input and output of such a system is in general a non-linear one, thus the recognition system must be capable of generating a sufficiently close approximation to the non-linear relationship defining the system.

1.2 The Volterra Functional Annalysis

In signal processing such non-linear systems may be modeled by use of the Volterra series. The form of the Volterra series was first studied by Vito Volterra, but Norbert Wiener was the first to apply it to non-linear systems theory. He used it to model the input output relationship of non-linear systems. The continuous Volterra series may be written:

$$y(t) = \mathbf{H}_0 + \mathbf{H}_1[x(t)] + \mathbf{H}_2[x(t)] + \ldots + \mathbf{H_n}[x(t)] + \ldots$$

Where $y(t)$ is the output of a system and $x(t)$ is the input and in which:

$$\mathbf{H_n}[x(t)] =$$

$$\int_{-\infty}^{+\infty} \cdots \int_{-\infty}^{+\infty} h_n(\tau_1, \ldots, \tau_n) x(t-\tau_1) \ldots x(t-\tau_n) d\tau_1 \ldots d\tau_n$$

Where $h_n(\tau_1, \ldots, \tau_n)$ is the Volterra kernel. A discrete form of the series was developed.

In general this will need to be a heterogeneous Volterra expansion in order to model systems composed of multiple orders of non-linearity. The Heterogeneous discrete Volterra series may written as:

$$\sum_{i=1,N} H_i$$

Where H_n are the Volterra Kernels, and w are coefficients which define the model as follows.

$$H_0 = w_0 \text{ Zeroth Order `D.C.' term}$$

$$H_1 = \sum_i w_i x_i \text{ First Order Linear term}$$

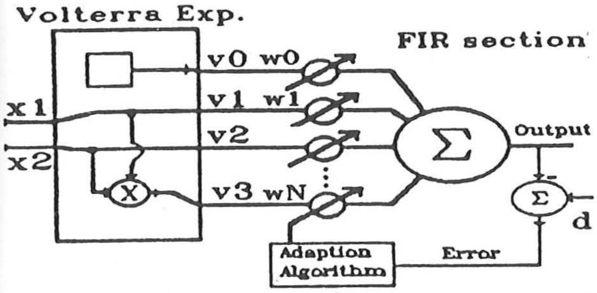
$$H_2 = \sum_i \sum_j w_{ij} x_i x_j \text{ Second order Quadratic term}$$

$$H_3 = \sum_i \sum_j \sum_k w_{ij,k} x_i x_j x_k \text{ ....and so on.}$$

Normally the coefficients are fixed by use of analytical methods however they may be fixed adaptively. Consider the following system in which the inputs are combined to give the individual Volterra terms. These terms are then multiplied by their respective coefficients and the terms summed. It can be seen that the system can be decomposed into a vector expansion stage followed by a linear Finite Impulse Response (FIR) filter. The adaption of the coefficients may then be achieved by any of the methods used in FIR adaptive filter theory. The analytical basis of this approach allows us to explicitly calculate the form of the error surface. The Volterra series is found to converge for most practical non-linearities, although large input levels or vey high order non-linearities should be avoided. A very closely related expansion may be obtained by using the Gabor-Kolmogorov polynomial:

$$y = w_0 + w_1 x_1 + w_2 x_2 + w_{11} x_1 x_1 + w_{12} x_1 x_2 +$$
$$w_{22} x_2 x_2 + w_{111} x_1 x_1 x_1 + w_{112} x_1 x_1 x_2 + \dots$$

Fig.2 The Connectionist model in block form.



In a system such as that shown in Fig.2 the effect of joining an FIR filter to the Volterra expansion may be considered. The system has a set of input values $x_0 \dots x_n$ which are input to the non-linear Volterra state expander to produce a set of output values $v_0 \dots v_q$ with n being less than q. These values then form the input to an FIR adaptive filter. Consider the output values for the vector state expander. This set of values may be represented by the vector **V**. The filter coefficients may also be represented by the vector **W**.

$$\mathbf{W} = \begin{pmatrix} w_0 \\ w_1 \\ w_2 \\ \vdots \\ w_n \end{pmatrix} \quad \mathbf{V} = \begin{pmatrix} v_0 \\ v_1 \\ v_2 \\ \vdots \\ v_n \end{pmatrix}$$

Using the standard FIR equation we may write the error ϵ as:

$$\epsilon = d - \sum_i v_i w_i$$

In which d is the desired output from the filter, that is in our case the correct classification given the set of inputs to the whole system x_i. In vector form this may be written:

$$\epsilon = d - \mathbf{W}^{\mathbf{T}}\mathbf{V}$$

If we wish to consider the error function for the Least Mean Squares, or LMS, critereon with the mean being considered over the pattern presentations, the equation becomes:

$$E(\epsilon^2) = E[(d - \mathbf{W}^{\mathbf{T}}\mathbf{V})^2] = \sigma^2 - 2\mathbf{WP} + \mathbf{WRW}$$

Where R is the autocorrelation matrix of the data in the non-linearly extended vector space and similarly P is the crosscorrelation vector between the desired d signal and the non-linearly expanded data.σ is the variance of the desired response.

The objective is to find a set of weights W which will minimise the mean square error. The system then will output classifications which differ from the desired classifications with the minimum mean square error possible for the system. The solution for the weights of the filter is obviously the same as that of a Weiner Filter operating in the extended space and so the methods used to find the Weiner solution explicitly may be applied. A more suitable approach for the recognition problem is to find the weight solution adaptively. This has the advantage of not requiring a priori knowledge of R, and does not require the inversion of a matrix.

It is clear from the above that the LMS error is solely a quadratic function of the weight vector. This shows that the performance surface is a hyperparaboloid and consequently always unimodal. This is a highly desirable property as it shows that the system will not exhibit local minima on the error surface. Such local minima cause the adaption performance of networks to be greatly reduced as the adaptive algorithm may get stuck in local minima and consequently not reach the optimal coefficient values. The property of unimodality also confers the ability to adapt much more quickly as there are no valleys in which to dither and the gradient is proportional to the distance from the solution. This last property prevents the adaption being greatly slowed or halted by regions of the error surface with small gradients. A stochastic gradient algorithm may be easily applied. Following the derivation of the LMS algorithm. The usual gradient algorithm may be written by differentiating the mean square error with respect to the weights to give an iterative weight update equation.

$$\mathbf{W} = \mathbf{W} - \mu\nabla E(\epsilon^2)$$

Where μ is an adaption coefficient which sets the speed of learning . A simple expression for $\nabla E(\epsilon^2)$ may be derived.

$$\frac{\partial E(\epsilon^2)}{\partial \mathbf{W}} = 2\epsilon\frac{\partial \epsilon}{\partial \mathbf{W}} = -2\epsilon\mathbf{X}$$

The analytic basis of the method combined with the property of linearity in the coefficients allows the application of the standard least squares methods. In general for pattern recognition the problem is degenerate or rank-deficient. A single solution may still be obtained by the application of Singular Value Decomposition (SVD) which yields the minimum norm solution. Such analytical method may also be used to study the network performance and obtain the eigenvalues for real recognition problems.

1.3 Parameter Tuning

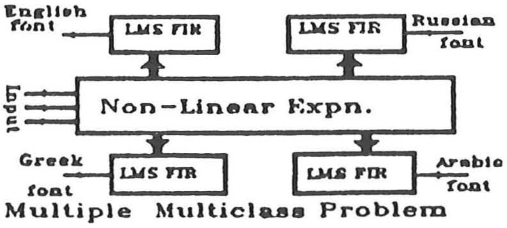
There is only one parameter to be chosen in the new network, this is the adaption coefficient. However, this adaption coefficient is that of a standard adaptive filter. Consequently the large body of knowledge pertaining to adaptive filter theory may be immediately applied. Yassa[7] developed an expression for choosing the optimal adaption coefficient. No such method may be applied in the case of current networks as the nature of the error surface is complex and contains local minima. Consequently the parameters such as the adaption coefficient must in general be set by trial and error in current networks.

1.4 Implementation

The Volterra expansion may be factorised to give efficient methods for its calculation requiring fewer multiplications than the direct form. It is necessary to take care in the case of large input signals and high order expansions to prevent overflow problems. Such problems are easily avoided by scaling the data such that it is approximately normalised. The extended space network is found to work well for multiclass problems, in which the network is required to classify the input to one of a series of output classes. This is most efficiently done by using one output which is required to output a series of levels corresponding to the different classes. This is in contrast with many networks which use a series of binary outputs to output multiple class classifications.

If a direct implementation of the network is used it is also possible to use the same expansion stage with multiple adaptive filters. This leads to an efficient system which may be used to perform a series of unrelated multiclass problems. Each muliclass problem being independent of the others and having its own set of coefficients.

Fig.3 Multiple Multiclass Recognition.



2 Direct Recognition

The network was first applied to the application of recognising binary font images of size 8 by 8 pixels. The sixty four pixels were input to the network and an error signal was generated by subtracting the network output from the value of the font member in its collating sequence. The adaption coefficient was approximately set by using an estimate of the average power of the input image. A second order network was used which was adapted using the LMS algorithm.

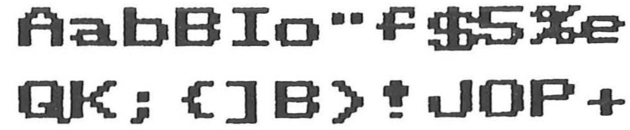


Fig.4 Examples of the font.

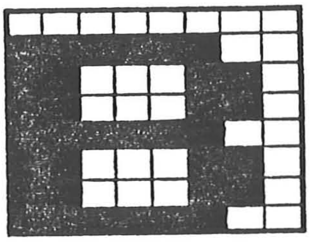


Fig.5 Individual Character

3 Invariant Recognition

A preprocessing section was used to produce patterns which were invariant to shift, scale or rotation. The recognition process was repeated using real camera images of simple handwritten capital letters.

3.1 Preprocessor

One of a series of capital letters was drawn with a marker pen onto a card. The image of the card was digitised by the use of a camera and framestore. The first stage performed by the preprocessor was an adaptive level binarisation; this was followed by a calculation of the centroid of the resulting image. The next stage was to use a simple edge finding algorithm to reduce the image to an outline image of the character. A line following algorithm was used to find sixty-four points in order around the outline. This algorithm was started from the leftmost point on the letter, and this process was repeated three times to take account of inner loops in the letters, for example, the letter B.

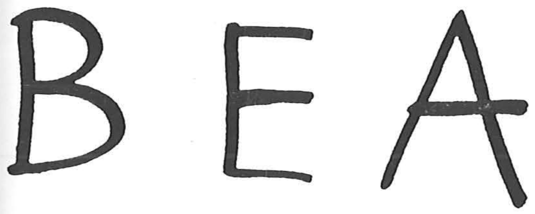


Fig.6 Example characters.

The euclidean distance from each point to the centroid was calculated. The referencing of the distance data to the centroid of the letter confers the property of positional invariance on the data. The power of the data is now normalised to be one. This confers the property of scale invariance on the data. A peak finding algorithm was used on the data and the data was shifted in a circular manner to put the peak value as the first data point. This operation gave the data the property of rotational invariance. Consequently the centroidal data patterns output by the post processor should be shift, scale, and rotation independent.

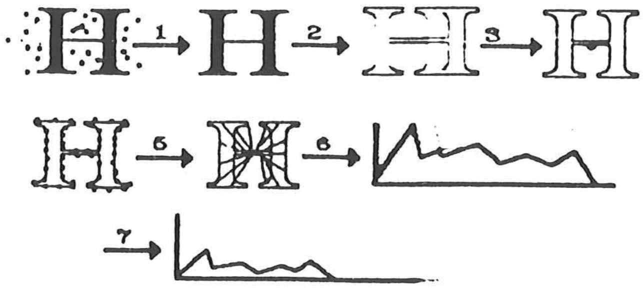


Fig.7 Preprocessor Stages.

Although in the majority of cases s.the preprocessor produces similar output patterns for aran individual letter there are some letters which underer certain circumstances will produce significantly differerent patterns. An example is the letter B. The first pass o:of the outline follower follows the outside of the letter, ;, however the second pass could follow the top loop or r the bottom loop and correspondingly generate significamantly different patterns depending soley on which loop haas the further left furthest left point. Consequently a commnectionist model must be used which is powerful enoough to cope with two significantly different clusters mapapping to the same output index.

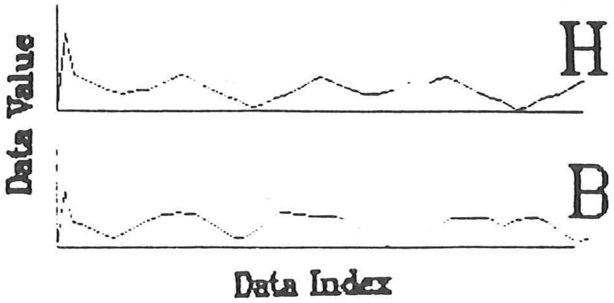


Fig.8 Centroidal Data form.

3.2 Results

Direct recognition The netwrok w was found to correctly recognise all the characters in aran eighty character font in around 7000 pattern presentatations.

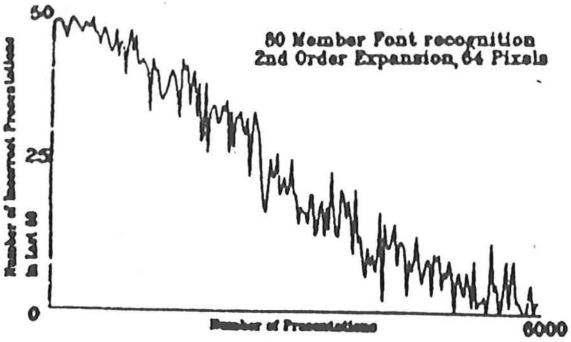


Fig.9 Plot of number of errors in the last fifty presentations against number of presentations.

Preprocessed Recognition The system was trained on twelve letters A.....L, initially using one card for each letter. Once the network was found to correctly recognise each card then gradually other cards with the same letters were introduced into the training runs. A new card was only introduced when the network had converged to a state giving no errors on the current cards.

	A	B	C	D	E	F	G	H	I	J	K	L
No. of Cards	4	4	4	4	4	4	4	4	4	4	4	4
%Error	3	3	2	3	6	6	4	2	1	3	2	2

Table showing number of letter cards and percentage error over random position ,scale, rotation.

Fig.10

4 Conclusion

In the direct recognition case the network performed well learning eighty separate class members, this is a higher number than usually reported for current networks. The convergence times were fast by comparison with current hidden-layer network which have been reported to require many tens of thousands of presentations to learn fewer characters.

In the preprocessed case the network demonstrated its ability to cope with variation in the patterns, and consequently be applicable in practical recognition systems which require not only position invariance, but tolerance to character variation.

This paper has introduced an application of the extended space network in optical character recognition. The network has been demonstrated to have a series of desirable properties and by the use of more advanced preprocessors may be used as the basis of higher performance recognition systems.

References

[1] S.Haykin, *Adaptive Filter Theory* Englewood Cliffs, NJ:Prentice-Hall, 1986.

[2] Schetsen, *The Volterra and Weiner Theories of Non-linear Systems* New York, NY:John Wiley, 1980.

[3] P. Alper, *A Consideration of the Discrete Volterra Series* IEEE Trans. Automatic Control, vol. , Jul. 1965, pp.322-327.

[4] D.£Spect, *Generation of Polynomial Discriminant Functions for Pattern Recognition* IEEE Trans. EC, vol. EC-16, No.3, Jun. 1967, pp.308-319.

[5] F.Yassa, *Optimality in the Choice of the Convergence Factor for Gradient Based Adaptive Algorithms* IEEE Trans. ASSP, vol. ASSP-35, No.1, Jan. 1987, pp.48-59.

[6] S.T.Alexander, *Transient Weight Misadjustment Properties for the Finite Precision LMS Algorithm* IEEE Trans. ASSP,vol. ASSP-35, No. 9, Sept. 1987, pp.1250-1258.

[7] D. Rumelhart & G. Hinton *Learning Representations by backpropogating errors* Nature Vol.323 9-10-1986.

[8] E.Polak, *Computational Methods in Optimisation* London, UK:Academic Press, 1971.

The Properties and Implementation of the Non-Linear Vector Space Connectionist Model.

M.R.Lynch and P.J.Rayner
Cambridge University Engineering Department, U.K.

.1 Introduction

In considering a connectionist model and its derivation it is necessary to first define the desirable properties of such a system and then hopefully construct such a system on a firm mathematical basis.

A suitable pattern recognition system will need to be adaptive for most applications. The necessary rules to define most real-world recognition tasks are too complex to be analytically derived. This becomes more so when the effects of noise or class variation are considered.

Once a system has been trained using the members of a training set it is hoped that the system will correctly classify a new pattern which was not in the training set but a member of a class defined by it. This property is often referred to as interpolation or generalisation as it is the ability of the system to interpolate the classifier output between training patterns.

It is obviously an advantage if the recognition system can adapt itself to a state of 'good' performance with as few presentations as possible. It is impractical to have systems which may get stuck in their learning processes and not reach their optimal performance; we thus require a system that will not suffer from local minima problems. These problems would require the system to be supervised and reset incurring multiple runs and consequently large computational expenditure.

Although most recognition tasks concerning multiple classes may be reduced to a series of one class recognition problems, this is in general highly inefficient in terms of computational load. An ideal system should be able to perform multiple class recognition.

For a system to be practically implementable a method of tuning any of the system network parameters must be found to prevent the need to repeat training runs in order to optimise these parameters.

The system should be highly parallel to allow high speed operation. A parallel implementation may also allow the processing resources to be distributed and so it may be composed of many simple processing units.

By considering work in the field of signal processing and connectionist models it possible to produce such a network, with a defined mathematical derivation.

Firstly consider a basic problem. A standard 'bench-mark' for neural networks is the exclusive OR or parity problem. In this problem a series of binary inputs are input to the network and it is required to return the parity function value of the inputs.

We can consider this problem from a signal space approach. Consider a vector space created by plotting each of the input variables along a set of orthogonal axes. For a 2 input variable problem this space is a plane. For the binary exclusive-OR problem the input values (or 'input vectors') are (1,1) (0,0) (1,0) (0,1). The first two being in class one and the second in class two. It is clear [11] that a linear discriminator cannot be used to differentiate between the classes in the EX-OR problem. We can, however, transform the problem into an extended vector space in which the problem may be solved using a linear discriminator. We increase the dimensionality of the decision space by adding a dimension x_1x_2 . A linear plane may now be used to separate the classes.

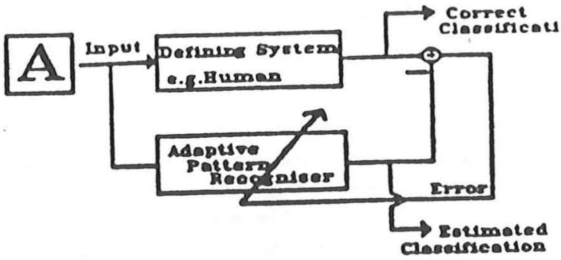
So by using a linear discriminator in an extended vector space it is possible to perform classification tasks requiring non-linear discriminators.

.2 Non-linear System Modeling

The pattern recognition problem may be thought of as a system modelling problem, in which the adaptive pattern recogniser is attempting to model the classification system; for example, a human defining the relationship between a set of character font images and a set of collating sequence numbers. The block diagram showing the recognition problem cast as a system modelling problem is shown in Fig.1.

It is clear that the relationship between the inputs and outputs of such a system is in general a non-linear one, thus the recognition system must be capable of generating a sufficiently close approximation to the non-linear relationship defining the system. Recently the field of Signal Processing has seen a large growth in interest in non-linear systems analysis by use of such methods as higher order spectra, and polycepstra [12] [4] [10]. These methods owe much of their basis to the Volterra Functional Analysis of non-linear systems.

Fig.1



.3 The Volterra Functional Analysis

In signal processing such non-linear systems may be modeled by use of the Volterra series. The form of the Volterra series was first studied by Vito Volterra, but Norbert Wiener was the first to apply it to non-linear systems theory. He used it to model the input output relationship of non-linear systems. The continous Volterra series may be written:

$$y(t) = \mathbf{H_0} + \mathbf{H_1}[x(t)] + \mathbf{H_2}[x(t)] + \dots + \mathbf{H_n}[x(t)] + \dots$$

Where $y(t)$ is the output of a system and $x(t)$ is the input and in which:

$$\mathbf{H_n}[x(t)] =$$

$$\int_{-\infty}^{+\infty} \dots \int_{-\infty}^{+\infty} h_n(\tau_1, \dots, \tau_n)x(t-\tau_1)\dots x(t-\tau_n)d\tau_1 \dots d\tau_n$$

Where $h_n(\tau_1, \dots, \tau_n)$ is the Volterra kernel. A discrete form of the series was developed.

In general this will need to be a heterogenous Volterra expansion in order to model systems composed of multiple orders of non-linearity. The Heterogeneous discrete Volterra series may written as:

$$\sum_{i=1,N} \mathbf{H_n}$$

Where H_n are the Volterra Kernels, and w are coefficients which define the model as follows.

$$\mathbf{H_0} = \mathbf{w_0} \text{ Zeroth Order term 'D.C.' term}$$

$$\mathbf{H_1} = \sum_{\mathbf{i}} \mathbf{w_i x_i} \text{ First Order Linear term}$$

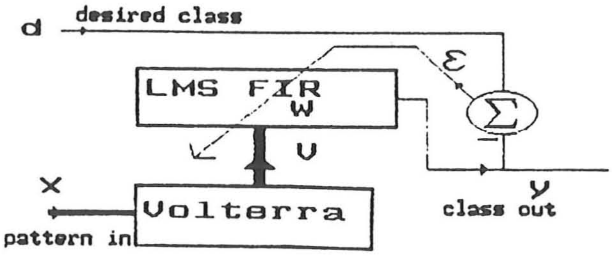
$$\mathbf{H_2} = \sum_{\mathbf{i}} \sum_{\mathbf{j}} \mathbf{w_{i,j} x_i x_j} \text{ Second order Quadratic Term}$$

$$\mathbf{H_3} = \sum_{\mathbf{i}} \sum_{\mathbf{j}} \sum_{\mathbf{k}} \mathbf{w_{i,j,k} x_i x_j x_k} \text{ ....and so on.}$$

Normally the coefficients are fixed by use of analytical methods, however they may be fixed adaptively. Consider the following system in which the inputs are combined to give the individual Volterra terms. These terms are then multiplied by their respective coefficients and the terms summed. It can be seen that the system can be decomposed into a vector expansion stage followed by a linear Finite Impulse Response (FIR) filter. The adaption of the coefficients may then be achieved by any of the methods used in FIR adaptive filter theory. The Volterra Series is found to converge for most practical non-linearites. Other very closely related expansions may also be used such as the Gabor-Kolmogorov polynomial [13]:

$$y = w_0 + w_1x_1 + w_2x_2 + w_{11}x_1x_1 + w_{12}x_1x_2 +$$
$$w_{22}x_2x_2 + w_{111}x_1x_1x_1 + w_{112}x_1x_1x_2 + \dots$$

Fig.2



In a system such as that shown in Fig.2 the effect of joining an FIR filter to the Volterra expansion may be considered. The system has a set of input values $x_0 \dots x_n$ which are input to the non-linear Volterra state expander to produce a set of output values $v_0 \dots v_q$ with n being less than q. These values then form the input to an FIR adaptive filter. Consider the output values for the vector state expander. This set of values may be represented by the vector V. The filter coefficients may also be represented by the vector W.

$$\mathbf{W} = \begin{pmatrix} w_o \\ w_1 \\ w_2 \\ \vdots \\ w_q \end{pmatrix} \quad \mathbf{V} = \begin{pmatrix} v_o \\ v_1 \\ v_2 \\ \vdots \\ v_q \end{pmatrix}$$

Using the standard FIR equation we may write the error ϵ as:

$$\epsilon = d - \sum_i v_i w_i$$

In which d is the desired output from the filter, that is in our case the correct classification given the set of inputs to the whole system x_i. In vector form this may be written:

$$\epsilon = d - \mathbf{W^T V}$$

If we wish to consider the error function for the Least Mean Squares, or LMS, critereon with the mean being considered over the pattern presentations, the equation becomes:

$$E(\epsilon^2) = E[(d - \mathbf{W^T V})^2] = \sigma^2 - 2\mathbf{WP} + \mathbf{WRW}$$

Where $\mathbf{R}$ is the autocorrelation matrix of the data in the non-linearly extended vector space and similarly $\boldsymbol{P}$ is the crosscorrelation vector between the desired d signal and the non-linearly expanded data.σ is the variance of the desired response.

The objective is to find a set of weights $\mathbf{W}$ which will minimise the mean square error (the Wiener solution in the extended space). The most suitable approach for the recognition problem is to find the weight solution adaptively. This has the advantage of not requiring a priori knowledge of $\mathbf{R}$, and does not require the inversion of a matrix.

It is clear from the above that the LMS error is solely a quadratic function of the weight vector. This shows that the performance surface is a hyperparaboloid and consequently always unimodal. This is a highly desirable property as it shows that the system will not exhibit local minima on the error surface. Such local minima cause the adaption performance of networks to be greatly reduced as the adaptive algorithm may get stuck in local minima and consequently not reach the optimal coefficient values. The property of unimodality also confers the ability to adapt much more quickly as there are no valleys in which to dither and the gradient is proportional to the distance from the solution. A stochastic gradient algorithm may be easily applied, following the derivation of the LMS algorithm, to give an iterative weight update equation.

$$\mathbf{W} = \mathbf{W} - \mu \nabla E(\epsilon^2)$$

Where μ is an adaption coefficient which sets the speed of learning and noting:

$$\frac{\partial E(\epsilon^2)}{\partial \mathbf{W}} = 2\epsilon \frac{\partial \epsilon}{\partial \mathbf{W}} = -2\epsilon \mathbf{X}$$

The analytic basis of the method combined with the property of linearity in the coefficients allows the application of the standard and other adaptive least squares methods (RLS [1] and so on). In general for pattern recognition the problem is degenerate or rank-deficient. A single solution may still be obtained by the application of Singular Value Decomposition (SVD) which yields the minimum norm solution. Such analytical methods are also useful in the study of the network performance.

.4 Network Parameter Tuning

There is only one parameter to be chosen in the new network, this is the adaption coefficient. However, this adaption coefficient is that of a standard adaptive filter. Consequently the large body of knowledge pertaining to adaptive filter theory may be immediately applied. No such method may be applied in the case of current networks as the nature of the error surface is complex and contains local minima. Consequently the parameters such as the adaption coefficient must in general be set by trial and error.

Yassa [7] demonstrated the relationship between the average input power P to the adaptive filter and the optimal adaption coefficient μ_{opt}. (μ_{int} is the constant adaption coefficient).

$$\mu_{opt} \alpha \frac{\mu_{int}}{P}$$

In setting the adaption coefficient value it has, as may be expected, been found that the system works best with an adaption coefficient which differs for each order of non-linearity as the order of the non-linearity affects the input power seen by the adaptive filter section. A neat method of achieving this is to use the same adaption coefficient for all terms but to arrange that the input power is roughly normalised for each tap by a suitable constant scaling of the input patterns. This also has the advantage of reducing the dynamic range of values for the higher order non-linearities which could be helpful for non digital implementations.

.5 Network Comparisons

The new network's performance was compared with that of the current networks. The exclusive OR problem was used as a test. The adaption times in terms of numbers of iterations were compared for the different networks. The times for the hidden layer back-propogation networks are taken from an American study [3]. These timings are for the best run, which was achieved by optimising the network parameters over a series of runs. If the network became stuck in a local minima it was stopped and restarted from different initial conditions. In the case of the new network the results are for the first time run as the adaption coefficient could be selected beforehand, and the local minimum problem did not exsist. Thus when comparing the figures it is necessary to realise that in terms of total pattern presentation numbers the factor between the number of presentations to each network is in fact considerably greater.

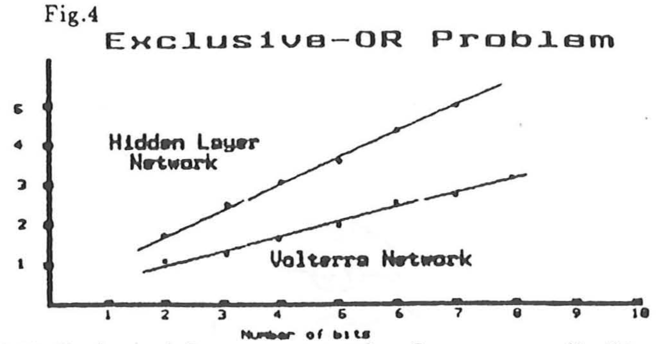
Fig.3

No.bits	Convergence Time	
n	T	T_{BP}
2	15	95
3	17	265
4	62	1200
5	106	4100
6	292	20000
7	558	100000
8	1287	500000

Table 1. Comparison of convergence times

If the figures in the above table are ploted for the two networks it is possible to see that the learning times for the current network rise quickly as a function of the number of input variables, whereas those for the new network rise at a lower rate. The results of the graph imply that the learning times for current networks are likely to be extremely large for complex problems with more than a few input variables.

Fig.4

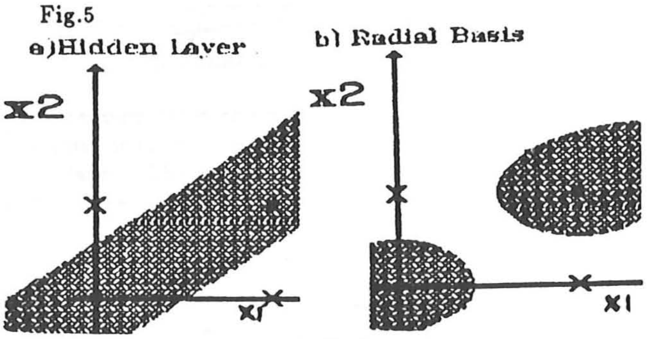


LOG Number of Presentations for Convergence V. Bits

.6 Interpolation Performance Comparison.

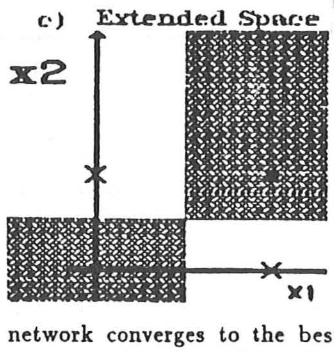
In this section the interpolative ability of two current network types (Hidden-Layer [8] and Radial Basis Function (RBFs)[5]) is compared with that of the new network. The network is trained on the exclusive OR class points (0,0), (1,0),(0,1),(1,1) until full convergence. The adaption is then stopped and the inputs are varied over the full signal space. The class decision is then ploted as a point for class one and no plot for class two [8].

Fig.5



The hidden layer network displays poor interpolative abilities which is to be expected as the decision region is bounded by linear boundries. The Radial Basis Function network performs much better, producing suitably localised classes. This increase in performance is to be expected due to the relationship between RBFs and multidimensional interpolation.

Fig.5a



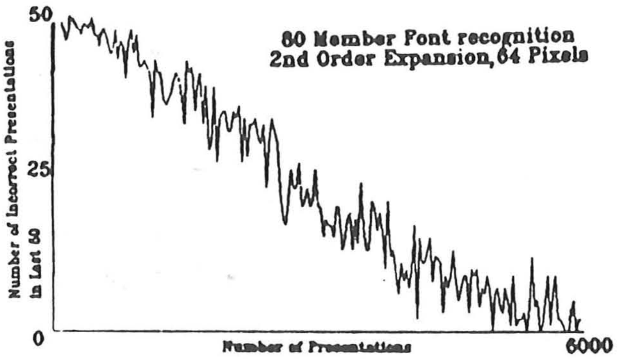
The new network converges to the best solution of the three, which is extremely near to the optimal solution, given no information other than at the class points. In fact, if the above is repeated for the network before it has converged, it displays results which are similar to those of RBF networks. However the new network is not constrained to a set of *a priori* 'RBFs' and so can converge further. In fact RBF networks may be considered as a subset of expanded space networks in which basis functions are used by using the same coefficient multiplying a series of expansion terms.

.7 Multiclass Performance

In practice it has not proved possible, in general, to use current networks for classification with large numbers of classes. In general, multiple class problems are addressed by using a series of networks. The flexibility of the decision surfaces of the new network, however, allows a single network to perform multiclass problems. One such problem is that of character recognition. The network was presented with an 8 by 8 pixel image of a character and required to output the characters position in the collating sequence. The network was capable of learning to recognise all 80 characters in the font. A second order Volterra extension was used with all terms present. The coefficients were set by the LMS algorithm. More details of this and other recognition experiments may be found in [9].

It is also possible to increase the number of classes that a system can recognise by using the same vector expansion 'hardware' with multiple adaptive filter sections connected to it. Each problem, although connected to the same expansion system, has completely separate coefficients and thus they do not interact.

Fig.6 Plot of number of incorrect classifications in the last fifty presentations against presentation number.



.8 Implementation

Factorisation: In order to facilitate the understanding of the new network it has so far been discussed as two sections: firstly the vector state expansion and secondly the adaptive filter section. The same network may be implemented in a more distributed form composed of a series of identical 'neurons' by factorising the expansion polynomial. Once this has been done it is possible to find factorisations composed of identical units. Consequently a network may be created using a network of simple terms [13]. Another approach which is currently under investigation is to define two types of 'neuron'. The first is composed of a simple multiplier and the second a standard linear perceptron. An interesting form of the second implementation is that in which the network elements are randomly connected. This generates a connectionist model containing random terms from the non-linear state expansion. However, provided that the network is large enough, the inherent degeneracy allows the network to find another possible solution without the missing terms.

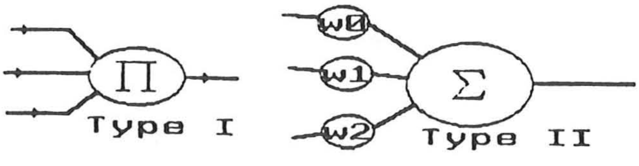


Fig.7 Basic Perceptrons

The level of degeneracy in a given network for a given problem may be indicated by the rank of the extended space correlation matrix **R**. This may be calculated by Singular Value Decomposition [1].

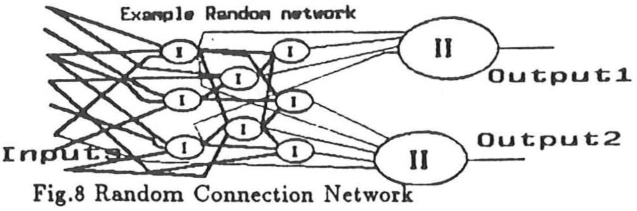


Fig.8 Random Connection Network

The random network was created by randomly connecting the inputs of type I neurons to type I neuron outputs in any earlier layer. The output type II neuron was connected to random type one outputs.

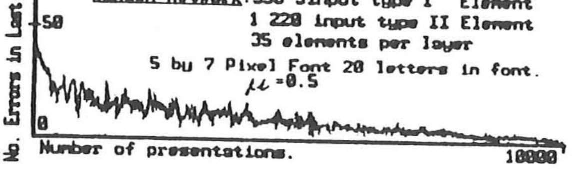


Fig.9 Results of Random Connection Network

.9 Conclusion

The non-linear vector space expansion connectionist model has been shown to have a number of advantages over currently applied networks for application to general connectionist problems.

Firstly, it converg es to its optimal solution far faster, by several orders of magnitude, than current networks, and this convergence time does not rise as quickly as a function of the size of the pattern vectors.

Secondly, it is unimodal and consequently does not suffer from the problems of local minima such as needing to be reset, getting stuck or uncertainty about having converged. It may thus be left in adaptive mode while being 'on the job' as well as in training.

Thirdly, the complete mathematical derivation of the new network allows a much fuller understanding of its operation and allows direct application of adaptive filter knowledge. This allows easy parameter tuning and application of other adaptive filter update methods. It also allows the effects of approximation and implementation to be considered.

The ability of the network to function well for random connection also allows the possibility of the implementation of the network by use of less exact technologies than electronic digital methods.

References

[1] S.Haykin, *Adaptive Filter Theory* Englewood Cliffs, NJ:Prentice-Hall, 1986.

[2] Schetzen, *The Volterra and Wiener Theories of Non-linear Systems* New York, NY:John Wiley, 1980.

[3] G.Tesauro & R.Janssens, *Scaling Relationships in Backpropogation Learning* Technical Report:Center for Complex Systems Research Univ. of Illionis.19-2-1988.

[4] P. Alper, *A Consideration of the Discrete Volterra Series* IEEE Trans. Automatic Control, vol. , Jul. 1965, pp.322-327.

[5] M.J.D.Powell, *Radial basis function approximations to Polynomials* Proc. Department of Applied Mathematics and Theoretical Physics.

[6] D.f.Spect, *Generation of Polynomial Discriminant Functions for Pattern Recognition* IEEE Trans. EC, vol. EC-16, No.3, Jun. 1967, pp.308-319.

[7] F.Yassa, *Optimality in the Choice of the Convergence Factor for Gradient Based Adaptive Algorithms* IEEE Trans. ASSP, vol. ASSP-35, No.1, Jan. 1987, pp.48-59.

[8] M.Niranjan & F.Fallside *Neural Networks and Radial basis functions in classifying static speech patterns* Internal Report Cambridge University Engineering Department CUED F-INFENG TR 22(1988).

[9] M.R.Lynch & P.J.Rayner *Optical Character Recognition Using a New Connectionist Model* Proc. IEE International Conference on Image Processing 89, Univ of Warwick.

[10] S.Fakhouri, *Identification of the Volterra Kernels of Non-linear Systems* IEE Proc. vol.127 No.6, Nov. 1980, pp.296-304.

[11] P.J.Rayner & M.R.Lynch *A New Connectionist Model Based on a Non-linear Adaptive Filter* Proc. ICASSP 89, pp.1191.

[12] C.Nikias, *Bispectrum Estimation: A Digital Signal Processing Framework* IEEE Proc. vol.75 No.7, Jul. 1987, pp.869-891.

[13] A.G.Ivakhnenko *Heuristic self-organisation in Problems of Engineering Cybernetics* Automatika vol.6 1970, pp.207-219.

Complexity Reduction in Volterra Connectionist Modelling by Consideration of Output Mapping

P.J.Rayner and M.R.Lynch
Cambridge University Engineering Department, U.K.

Introduction

The Volterra Connectionist model has already been demonstrated to have advantages over current networks [6][8][9]. It has been shown to have much faster learning times and being unimodal it does not suffer from local minima problems. The mathematical derivation of the network has also allowed a deeper analysis of network operation, and it is this understanding of the system which may be used to greatly reduce network complexity by consideration of the output mapping. Consider the pattern recognition problem as a system transfer function:

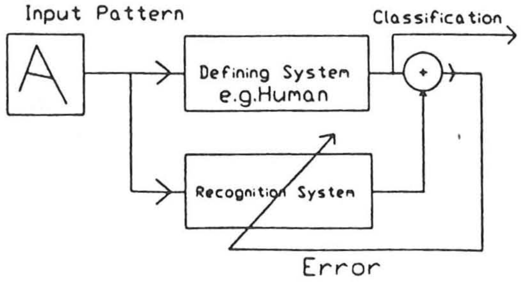


Fig.1 Pattern Recognition Problem

In general the relationship between the input pattern data and the output classification data is non-linear and consequently the pattern recognition system must in general be a non-linear one. The Volterra connectionist model may be shown to perform this task [2][9].

A heterogenous Volterra may be used to model systems composed of multiple orders of non-linearity. The Heterogenous discrete Volterra series may written as:

$$\sum_{n=1,N} H_n$$

Where H_n are the Volterra Kernels, and w are coefficients which define the model as follows.

$$H_0 = w_0 \text{ Zeroth Order 'D.C.' term}$$

$$H_1 = \sum_i w_i x_i \text{ First Order Linear term}$$

$$H_2 = \sum_i \sum_j w_{i\,j} x_i x_j \text{ Second order Quadratic Term}$$

$$H_3 = \sum_i \sum_j \sum_k w_{i\,j,k} x_i x_j x_k \text{ ....and so on.}$$

Consider the following system in which the inputs are combined to give the individual Volterra terms. These terms are then multiplied by their respective coefficients and the terms summed. It can be seen that the system can be decomposed into a vector expansion stage followed by a linear Finite Impulse Response (FIR) filter. The adaption of the coefficients may then be achieved by any of the methods used in FIR adaptive filter theory. The Volterra Series is found to converge for most practical non-linearites.

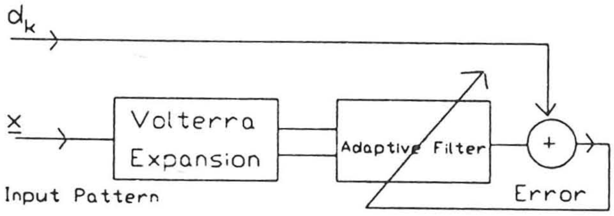


Fig.2 Recognition System

Output Mapping

The basis of complexity reduction may be easily demonstrated by consideration of a classification problem for one input variable.(Fig.3). This would require a higher order of non-linearity and consequently many terms from the Volterra expansion leading to the need for a connectionist model with many terms. If, however, the particular values of the classification indices had been correctly chosen the problem could be reduced to a linear one (Fig.4). The problem of choosing the indices suitably is considerably more difficult for practical problems with many input variables.

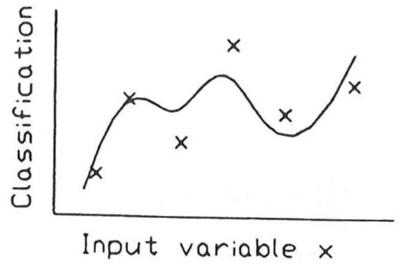


Fig.3 One variable recognition problem.

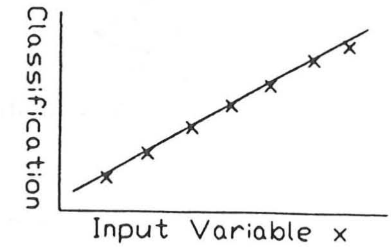


Fig.4 Linearised Problem

Derivation

The following approach to suitable selection of these indices has been shown to give good results.It uses a constraint to ensure that the output indices derived are separated from one another and consequently takes the form of a constrained optimisation.

Suppose that there are M classes and in the training phase there are N_i (i=1,M) presentations for each class . Let d_i be the classification index for each class. Let $\mathrm{x_n}(i)$ be the extended space pattern vector for the ith class and the nth example of it. Let w be the network weight vectors.

$$\epsilon = \sum_{i=1}^{M} \left[\sum_{n=1}^{N_i} \left[d_i - \mathbf{w}^t \mathbf{x}_n(i) \right]^2 \right]$$

The d_i must be constrained. A suitable constraint is found to be:

$$\sum_{i=1}^{M} d_i^2 = c$$

where c is an arbitrary constant.

The method of Lagrange multipliers may now be applied. Set up the objective function:

$$Q = \sum_{i=1}^{M} \left[\sum_{n=1}^{N_i} \left[d_i - \mathbf{w}^t \mathbf{x}_n(i) \right]^2 \right] + \lambda \sum_{i=1}^{M} d_i^2$$

$$\frac{\partial Q}{\partial \mathbf{w}} = \sum_{i=1}^{M} 2 \left[\sum_{n=1}^{N_i} \left[d_i - \mathbf{w}^t \mathbf{x}_n(i) \right] \mathrm{x_n}(i) \right] = 0 \ldots 1$$

$$\frac{\partial Q}{\partial d_j} = \sum_{n=1}^{N_k} 2 \left[d_k - \mathbf{w}^t \mathbf{x}_n(k) \right] + \lambda 2 d_k = 0 \ldots 2$$

where k=1,M.

From 1:

$$\sum_{i=1}^{M} \left[d_i \sum_{n=1}^{N_i} \mathbf{x}_n(i) - \left(\sum_{n=1}^{N_i} \mathbf{x}_n(i) \mathbf{x}_n^t(i) \right) \mathbf{w} \right] = 0$$

$$\sum_{i=1}^{M} d_i \sum_{n=1}^{N_i} {}' \mathbf{x}_n(i) - \left[\sum_{i=1}^{M} \sum_{n=1}^{N_i} \mathbf{x}_n(i) \mathbf{x}_n^t(i) \right] \mathbf{w} = 0$$

Define:

$$\sum_{n=1}^{N_i} \mathbf{x}_n(i) = \mathbf{p}(i)$$

and:

$$\sum_{i=1}^{M} \sum_{n=1}^{N_i} \mathbf{x}_n(i) \mathbf{x}_n^t(i) = \mathbf{R_{XX}}$$

Then:

$$\sum_{i=1}^{M} d_i \mathbf{p}(i) - \mathbf{R_{XX}} \mathbf{w} = 0 \ldots 5$$

From 2:

$$(\lambda + N_k) d_k - \sum_{n=1}^{N_k} \mathbf{w}^t \mathbf{x}_n(k) = 0$$

$$(\lambda + N_k) d_k - \sum_{n=1}^{N_k} \mathbf{x}(k) = 0 \ldots 4$$

4 becomes:

$$(\lambda + N_k) d_k - \mathbf{w}^{\mathbf{t}} \mathbf{p}(k) = 0 \ldots 6$$

$k = 1, M$. Multiply 6 by d_k and sum over k (change to i):

$$\sum_{i=1}^{M} (\lambda + N_i) d_i^2 - \sum_{i=1}^{M} \mathbf{w}^{\mathbf{t}} \mathbf{p}(i) d_i = 0$$

For convenience, assume $N_i = N$ for $i = 1, M$:

$$(\lambda + N) \sum_{i=1}^{M} d_i^2 - \sum_{i=1}^{M} \mathbf{w}^{\mathbf{t}} \mathbf{p}(i) d_i = 0$$

But :

$$\sum_i d_i^2 = c$$

so:

$$\lambda + N = \frac{1}{c} \mathbf{w}^{\mathbf{t}} \sum_{i=1}^{M} d_i \mathbf{p}(i) \ldots 7$$

Substitution (5,6):

$$\sum_{i=1}^{M} \frac{1}{\lambda + N} \mathbf{w}^{\mathbf{t}} \mathbf{p}(i) \mathbf{p}(i) - \mathbf{R_{XX}} \mathbf{w} = 0$$

$$\frac{1}{\lambda + N} \mathbf{P_{XX}} \mathbf{w} = \mathbf{R_{XX}} \mathbf{w}$$

where:

$$\mathbf{P_{XX}} = \sum_{i=1}^{M} \mathbf{p}(i)\mathbf{p}^{t}(i)$$

Assuming $\mathbf{R_{XX}}$ non-singular then:

$$\mathbf{R_{XX}^{-1}P_{XX}w} = (\lambda + N)\mathbf{w}$$

That is, w is an eigenvector of $\mathbf{R_{XX}^{-1}P_{XX}}$.

$$\mathbf{w} = \alpha\mathbf{e}$$

where e is an eigenvector. Multiply 5 by $\mathbf{w}^{t}$:

$$\mathbf{w}^{t}\sum_{i=1}^{M} d_i\mathbf{p}(i) - \mathbf{w}^{t}\mathbf{R_{XX}w} = 0 \ldots 10$$

Substitute 7 into 10:

$$c(\lambda + N) = \mathbf{w}^{t}\mathbf{R_{XX}w} \ldots 11$$

Substituting 11 into 6 gives:

$$d_k = \frac{c\mathbf{w}^{t}\mathbf{p}(k)}{\mathbf{w}^{t}\mathbf{R_{XX}w}}$$

From 11:

$$c\mu = \alpha^2\mathbf{e}^{t}\mathbf{R_{XX}e} \ldots 13$$

where μ is the eigenvalue corresponding to e, i.e. $\lambda + N = \mu$. Now, to decide on which eigenvalue to choose, consider the error:

We have:

$$\epsilon = \sum_{i=1}^{M}\left[\sum_{n=1}^{N_i}\left[d_i - \mathbf{w}^{t}x(i)\right]^2\right] \ldots 14$$

and:

$$d_k = \frac{c\mathbf{w}^{t}\mathbf{p}(k)}{\mathbf{w}^{t}\mathbf{R_{XX}w}}$$

and from 13:

$$\alpha = \left[\frac{\mu c}{\mathbf{e}^{t}\mathbf{R_{XX}e}}\right]^{\frac{1}{2}}$$

Expanding 14 gives:

$$\epsilon = \sum_{i=1}^{M}\sum_{n=1}^{M}\left[d_i^2 - 2d_i\mathbf{w}^{t}\mathbf{xn}(i) + (\mathbf{w}^{t}\mathbf{xn}(i))^2\right]$$

$$= Nc - 2\sum_{i=1}^{M} d_i\mathbf{w}^{t}\mathbf{p}(i) + \mathbf{w}^{t}\mathbf{R_{XX}w}$$

Substitute 10:

$$\epsilon = Nc - \sum_{i=1}^{M} d_i\mathbf{w}^{t}\mathbf{p}(i) \ldots 17$$

From 15:

$$\mathbf{w}^{t}\mathbf{p}(k) = \frac{d_k}{c}\mathbf{w}^{t}\mathbf{R_{XX}w}$$

Substitute in 17:

$$\epsilon = Nc - \sum_{i=1}^{M}\frac{d_i^2}{c}\mathbf{w}^{t}\mathbf{R_{XX}w}$$

$$= Nc - \mathbf{w}^{t}\mathbf{R_{XX}w}$$

$$\epsilon = c(N - \mu)$$

where μ is the eigenvalue corresponding to e. Hence the error is aminimum when the largest eigenvalue is selected. Substitution shows:

$$d_k = \frac{\mathbf{w}^{t}\mathbf{p_k}}{\mu}$$

By use of this expression a set of 'optimal' output indices may be found. These indices are in general not integers and in fact a convenient approach to practical implementation has been found. The original integer indices are simply reordered for the classes in the same order as the optimal ones. This is found to work well as may be expected if the connectionist model is viewed as a multidimensional interpolator.

Results

The output mapping scheme was first applied to a two input variable problem with class clusters as shown below:

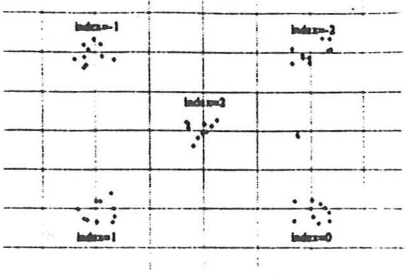


Fig.5 Input pattern clusters and indices

These clusters and assignments gave the following outputs for each class. The output mapping system was now run to give the following optimal indices and the original indices reordered. The plots show the desired and actual outputs against pattern number, solid line is the desired output and the dotted line the actual output.

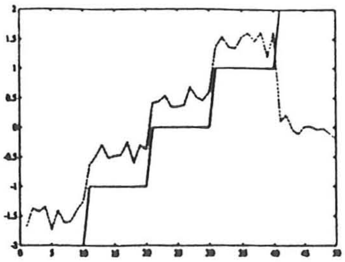


Fig.6 Plot of output with original indices

It can be seen that the output error has been greatly reduced by optimal ordering of the output indices.

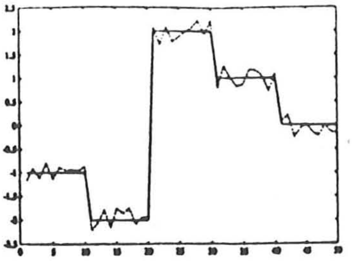


Fig.7 Plot of outputs with reordered indices

The output mapping approach was next applied to the problem of optical character recognition, using a 80 font member set with 8 by 8 pixels for each character. The desired output was the position of the character in the collating sequence 0 to 79.

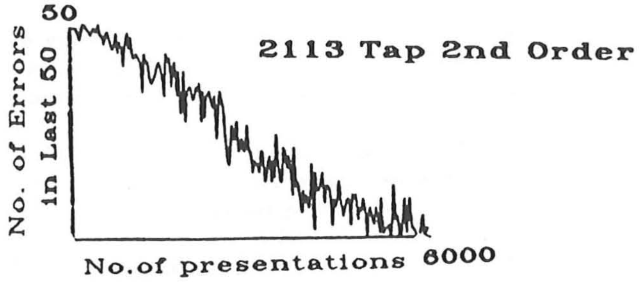


Fig 8.Results for 80 member font OCR with first and second order terms

With the arbitrary indices the linear system is not capable of solving the problem. However, if the indices are optimised a linear system can now solve the problem.

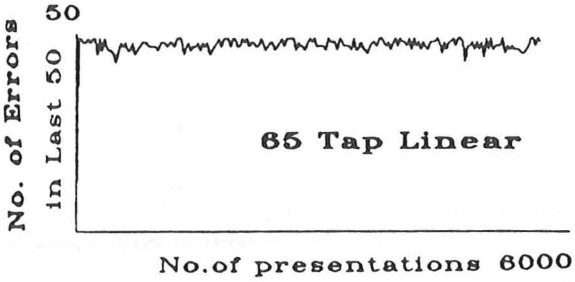


Fig. 9 Ocr results using linear first order terms.

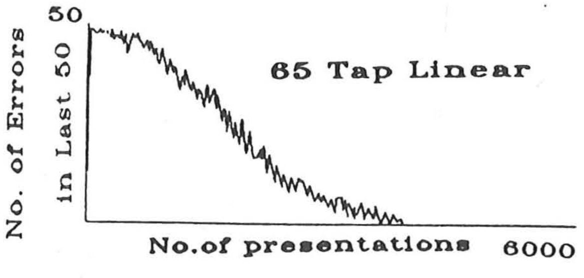


Fig.10 Results for 80 member font OCR with first order terms but optimised indices

Conclusion

The Volterra connectionist model has already been shown to have significant advantages over other systems in terms of analytical basis, speed of learning, scaling properties, and generalisation. The output mapping method demonstrates an approach which makes the Volterra connectionist model highly efficient computationally, by comparison with current neural networks. Although in the above results the system was applied to yield linear recognisers exactly the same approach may be used to reduced a higher order non-linear recogniser to a lower order one. The network is no longer being constrained to fit arbitrary indices and some may utilise all of its degrees of freedom to improve recognition performance.

References

[1] S.Haykin, *Adaptive Filter Theory* Englewood Cliffs, NJ:Prentice-Hall, 1986.

[2] Schetzen, *The Volterra and Wiener Theories of Nonlinear Systems* New York, NY:John Wiley, 1980.

[3] G.Tesauro & R.Janssens, *Scaling Relationships in Backpropogation Learning* Technical Report:Center for Complex Systems Research Univ. of Illionis.19-2-1988.

[4] P. Alper, *A Consideration of the Discrete Volterra Series* IEEE Trans. Automatic Control, vol. , Jul. 1965, pp.322-327.

[5] D.F.Specht, *Generation of Polynomial Discriminant Functions for Pattern Recognition* IEEE Trans. EC, vol. EC-16, No.3, Jun. 1967, pp.308-319.

[6] M.R.Lynch & P.J.Rayner *Optical Character Recognition Using a New Connectionist Model* Proc. IEE International Conference on Image Processing 89, Univ of Warwick.

[7] S.Fakhouri, *Identification of the Volterra Kernels of Non-linear Systems* IEE Proc. vol.127 No.6, Nov. 1980, pp.296-304.

[8] P.J.Rayner & M.R.Lynch *A New Connectionist Model Based on a Non-linear Adaptive Filter* Proc. ICASSP 89, pp.1191.

[9] M.R.Lynch & P.J.Rayner *Properties and Implementation of the Non-linear vector space Connectionist model* Proc. IEE 1st International Conference on Artificial Neural Networks 89, London.

A Unimodal Neural Network Utilising Non-linear Vector Space Expansion Methods, Comparison with Current Networks and Research Discussion.

M.R.Lynch
Cambridge University Engineering Department,
Trumpington Street, Cambridge,
CB2 1PZ, U.K.

1 Introduction

This document is designed to be a basic introduction to the vector space expansion connectionist model.

The modern computer has proved itself to be well suited to solving many problems and provides a powerful tool for aiding in the solution of many other problems. There has, however, emerged a type of problem for which standard 'syntactic' or programming methods do not perform well. These problems generally have many input variables and many complex rules relating their outputs to their inputs. The rules that arise in many real world problems, such as visual object recognition under practical conditions, are in general too complex for a suitable set of rules to be explicitly worked out. Consequently, in general, syntactic methods of sufficient complexity cannot practically be constructed to solve these problems.

The adaptive neural network [15] can address this problem in that it does not require explicit statement of the rules of the problem, as it will implicitly learn them. Such networks are composed of many simple processing units linked together to create a system capable of performing complex tasks.

A standard application of such networks is in pattern recognition. This document attempts to define the abilities that a pattern recognition system requires, and, to investigate the performance of the extended vector space connectionist model with respect to these abilities.

1.1 Adaptivity and Learning

A suitable pattern recognition system will need to be adaptive for most applications. The necessary rules to define most real-world recognition tasks are too complex to be analytically derived. This becomes more so when the effects of noise or class variation are considered. These problems may be circumvented by utilisation of a self learning system, which by experience of the recognition problem adapts itself to solve the problem.

1.2 Interpolation

Once a system has been trained using the members of a training set it is hoped that the system would correctly classify a new pattern which was not in the training set, but a member of a class defined by it. This property is often referred to as interpolation or generalisation, as it is the ability of the system to interpolate the classifier output between training patterns. There is an implicit assumption in the concept of interpolation that patterns belonging to the same class will cluster to a similar region of the decision space. This may not always apply, for example in the case of a capital and a small letter 'A'. However, from an analytical viewpoint this may be considered as two classes mapping to the same output value. It would be hoped that capital A letters will cluster and so on.

1.3 Learning Speed

It is obviously an advantage if the recognition system can adapt itself to a state of 'good' performance with as few presentations as possible. The ability to adapt quickly is also of advantage as it also enables the recognition system to respond to changes in the noise or class statistics.

1.4 Certainty of learning

It is impractical to have systems which may get stuck in their learning processes and not reach their optimal performance; we thus require a system that will not suffer from local minima problems. These problems would require the system to be supervised and reset incurring multiple runs and consequently large computational expenditure.

1.5 Multiple class Problems

Although most recognition tasks concerning multiple classes may be reduced to a series of one class recogntion problems, this is in general highly inefficient in terms of computational load. An ideal system should be able to perform multiple class recognition.

1.6 Parameter Tuning

For a system to be practically implementable a method of tuning any of the system network parameters must be found to prevent the need to repeat training runs inorder to optimise these parameters.

1.7 Parallelism

The system should be highly parallel to allow high speed operation. A parallel implemetation may also allow the processing resources to be distributed and so it may be composed of many simple processing units.

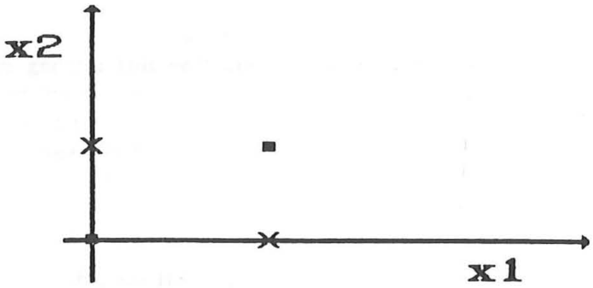
2 The Signal Processing Approach

In the early days of neural networks there was a tendency to over-extrapolate known physiological information to provide networks . There was also a tendency to produce networks by empirical methods. This led to systems which could not easily be analysed or optimised. Recently the field of signal processing has advanced sufficiently to allow the mathematical derivation of a network form by using a well defined signal processing approach.

Firstly it is necessary to consider a basic problem. A standard 'bench-mark' for neural networks is the exclusive OR or parity problem. In this problem a series of binary inputs are input to the network and it is required to return the parity function value of the inputs.

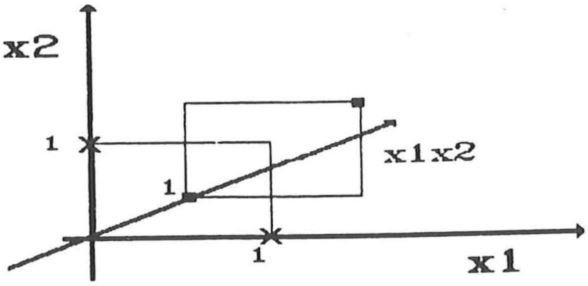
We can consider this problem from a signal space approach. Consider a vector space created by plotting each of the input variables along a set of orthogonal axes. For a 2 input variable problem this space is a plane. For the binary exclusive-OR problem the input values (or 'input vectors') are (1,1) (0,0) (1,0) (0,1). The first two being in class one and the second in class two.

Fig.1



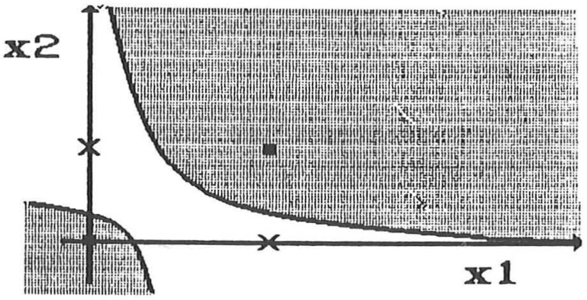
It can be seen from Fig.1 that a linear discriminator cannot be used to differentiate between the classes in the EX-OR problem. We can however transform the problem into an extended vector space in which the problem may be solved using a linear discriminator. We increase the dimensionality of the desicion space by adding a dimension x_1x_2 . A linear plane may now be used to separate the classes.

Fig.2



If we collapse the desicion plane of Fig.2 onto the 2D plane as in Fig.1 it can be seen that the decision contour is non-linear.

Fig.3

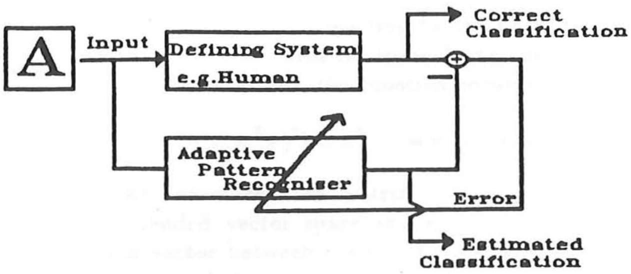


So by using a linear discriminator in an extended vector space it is possible to perform classification tasks requiring non-linear discriminators.

2.1 Non-linear System Modeling Approach

The pattern recognition problem may be thought of as a system modeling problem, in which the adaptive pattern recognition is attempting to model the classification system; for example, a human defining the relationship between a set of character font images and a set of collating sequence numbers. The block diagram showing the recognition problem cast as a system modelling problem is shown in Fig.4.

Fig.4



It is clear that the relationship between the inputs and outputs of such a system is in general a non-linear one, thus the recognition system must be capable of generating a sufficiently close approximation to the non-linear relationship defining the system.

2.2 The Volterra Functional Analysis

In signal processing such non-linear systems may be modeled by use of the Volterra series [2]. The form of the Volterra series was first studied by Vito Volterra, but Norbert Wiener was the first to apply it to non-linear systems theory. He used it to model the input output relationship of non-linear systems. The continous Volterra series may be written:

$$y(t) = \mathbf{H_0} + \mathbf{H_1}[x(t)] + \mathbf{H_2}[x(t)] + \ldots + \mathbf{H_n}[x(t)] + \ldots$$

Where $y(t)$ is the output of a system and $x(t)$ is the input and in which:

$$\mathbf{H_n}[x(t)] =$$

$$\int_{-\infty}^{+\infty} \cdots \int_{-\infty}^{+\infty} h_n(\tau_1, \ldots, \tau_n)x(t-\tau_1)\ldots x(t-\tau_n)d\tau_1 \ldots d\tau_n$$

Where $h_n(\tau_1, \ldots, \tau_n)$ is the Volterra kernel. A discrete form of the series was developed [4].

In general this will need to be a heterogenous Volterra expansion in order to model systems composed of multiple orders of non-linearity. The Heterogenous discrete Volterra series may written as:

$$\sum_{i=1,N} \mathbf{H_n}$$

Where H_n are the Volterra Kernels, and w are coefficients which define the model as follows.

$$\mathbf{H_0} = \mathbf{w_0} \text{ Zeroth Order term `D.C.' term}$$

$$\mathbf{H_1} = \sum_{i} \mathrm{w_i x_i} \text{ First Order Linear term}$$

$$\mathbf{H_2} = \sum_{i}\sum_{j} \mathrm{w_{i,j} x_i x_j} \text{ Second order Quadratic Term}$$

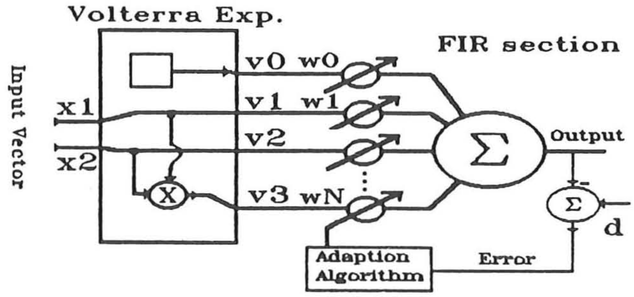
$$\mathbf{H_3} = \sum_{i}\sum_{j}\sum_{k} \mathrm{w_{i,j,k} x_i x_j x_k} \text{ ....and so on.}$$

Normally the coefficients are fixed by use of analytical methods [12] however they may be fixed adaptively. Consider the following system in which the inputs are combined to give the individual Volterra terms. These terms are then multiplied by their respective coefficients and the terms summed. It can be seen that the system can be decomposed into a vector expansion stage followed by a linear Finite Impulse Response (FIR) filter. The adaption of the coefficients may then be achieved by any of the methods used in FIR adaptive filter theory. The analytical basis of this approach allows us to explicitly calculate the form of the error surface. The Volterra Series is found to converge for most practical non-linearites, although large input levels or vey high order non-linearities should be avoided. A very closely related expansion may be obtained by using the Gabor-Kolmogorov polynomial: [16]

$$y = w_0 + w_1x_1 + w_2x_2 + w_{11}x_1x_1 + w_{12}x_1x_2 +$$

$$w_{22}x_2x_2 + w_{111}x_1x_1x_1 + w_{112}x_1x_1x_2 + \ldots$$

Fig.5



In a system such as that shown in Fig.5 the effect of joining an FIR filter to the Volterra expansion may be considered. The system has a set of input values $x_0 \ldots x_n$ which are input to the non-linear Volterra state expander to produce a set of output values $v_0 \ldots v_q$ with n being less than q. These values then form the input to an FIR adaptive filter. Consider the output values for the vector state expander. This set of values may be represented by the vector V. The filter coefficients may also be represented by the vector W.

$$\mathrm{w} = \begin{pmatrix} w_o \\ w_1 \\ w_2 \\ \vdots \\ w_n \end{pmatrix} \quad \mathrm{v} = \begin{pmatrix} v_o \\ v_1 \\ v_2 \\ \vdots \\ v_n \end{pmatrix}$$

Using the standard FIR equation we may write the error ϵ as:

$$\epsilon = d - \sum_{i} v_i w_i$$

In which d is the desired output [1] from the filter, that is in our case the correct classification given the set of inputs to the whole system x_i. In vector form this may be written:

$$\epsilon = d - \mathbf{w}^{\mathbf{t}}\mathbf{v}$$

If we wish to consider the error function for the Least Mean Squares, or LMS, critereon with the mean being considered over the pattern presentations, the equation becomes:

$$E(\epsilon^2) = E[(d - \mathbf{w}^{\mathbf{t}}\mathbf{v})^2] = \sigma^2 - 2\mathbf{w}\mathbf{P} + \mathbf{w}^{\mathbf{t}}\mathbf{R}\mathbf{w}$$

Where R is the autocorrelation matrix of the data in the non-linearly extended vector space and similarly P is the crosscorrelation vector between the desired d signal and the non-linearly expanded data.σ is the variance of the desired response.

The objective is to find a set of weights $\mathbf{w}$ which will minimise the mean square error. The system then will output classifications which differ from the desired classifications with the mimimum mean squared error possible for the system. The optimal solution for the weights of the filter is the same as that of a Wiener Filter [1] operating in the extended space and so the methods used to find the Weiner solution explicitly may be applied. A more suitable approach for the recognition problem is to find the weight solution adaptively. This has the advantage of not requiring a priori knowledge of $\mathbf{R}$, and does not require the inversion of a matrix.

It is clear from the above that the LMS error is solely a quadratic function of the weight vector. This shows that the performance surface is a hyperparaboloid and consequently always unimodal. This is a highly desirable property as it shows that the system will not exhibit local minima on the error surface. Such local minima cause the adaption performance of networks to be greatly reduced as the adaptive algorithm may get stuck in local minima and consequently not reach the optimal coefficient values. The property of unimodality also confers the ability to adapt much more quickly as there are no valleys in which to dither and the gradient is proportional to the distance from the solution. This last property prevents the adaption being greatly slowed or halted by regions of the error surface with small gradients. A stochastic gradient algorithm [11] may be easily applied. Following the derivation of the LMS algorithm. The usual gradient algorithm may be written by differentiating the mean square error with respect to the weights to give an iterative weight update equation.

$$\mathbf{w} = \mathbf{w} - \mu\nabla E(\epsilon^2)$$

Where μ is an adaption coefficient which sets the speed of learning . A simple expression for $\nabla E(\epsilon^2)$ may be derived.

$$\frac{\partial E(\epsilon^2)}{\partial \mathbf{w}} = 2\epsilon\frac{\partial\epsilon}{\partial \mathbf{w}} = -2\epsilon\mathbf{x}$$

The analytic basis of the method combined with the property of linearity in the coefficients allows the application of the standard least squares methods. In general for the pattern recognition the problem is degenerate or rank-deficient. A single solution may still be obtained by the application of Singular Value Decomposition (SVD) which yields the minimum norm solution. Such analytical method may also be used to study the network performance and obtain the eigenvalues for real recognition problems.

2.3 Parameter Tuning

There is only one parameter to be chosen in the new network, this is the adaption coefficient. However, this adaption coefficient is that of a standard adaptive filter. Consequently the large body of knowledge pertaining to adaptive filter theory may be immediately applied. Yassa[7] developed an expression for choosing the optimal adaption coefficient. No such method may be applied in the case of current networks as the nature of the error surface is complex and contains local minima. Consequently the parameters such as the adaption coefficient must in general be set by trial and error in current networks.

3 Network Comparisons

The new network's performance was compared with that of the current networks. The exclusive OR problem was used as a test. The adaption times in terms of numbers of iterations were compared for the different network times. The times for the hidden layer back-propogation networks [10] are taken from an American study [3]. These timings are for the best run, which was achieved by optimising the network parameters over a series of runs. If the network became stuck in a local minima it was stopped and restarted from different initial conditions. In the case of the new network the results are for the first time run as the adaption coefficient could be selected beforehand, and the local minimum problem did not exsist. Thus when comparing the figures it is necessary to realise that in terms of total pattern presentation numbers the factor between the number of presentations to each network is in fact considerably greater.

Fig.6

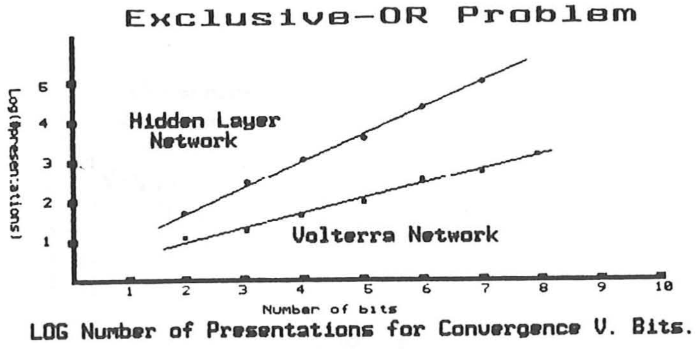
Parity Problem

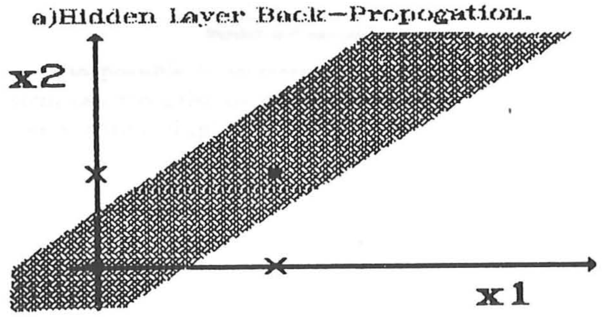
$\mu = .01$

No.of Bits. n	Volterra Extension Tconv	Hidden Layer Tconv
2	15	95
3	17	265
4	62	1200
5	106	4100
6	292	20000
7	556	100000
8	1287	500000

If the figures in the above table are ploted for the two networks it is possible to see that the learning times for the current network rise quickly as a function of the number of input variables, whereas those for the new network rise at a lower rate. The results of the graph imply that the learning times for current networks are likely to be extremely large for complex problems with more than a few input variables.

Fig.7

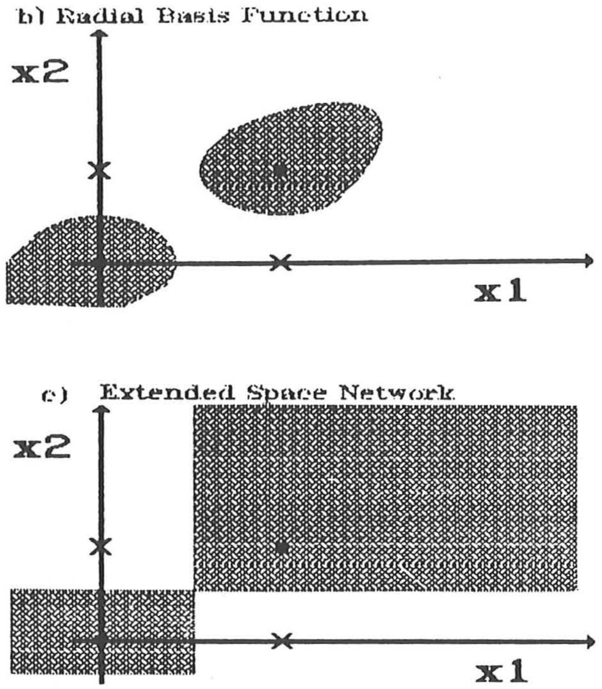




3.1 Interpolation Performance Comparison.

In this section the interpolative ability of two current network types Hidden-Layer and Radial Basis Function (RBFs)[5][9] is compared with that of the new network. The network is trained on the exclusive OR class points (0,0), (1,0),(0,1),(1,1) until full convergence. The adaption is then stopped and the inputs are varied over the full signal space. The class decision is then ploted as a point for class one and no plot for class two. Graphs a) and b) were provided by Niranjan[9].

Fig.8



The hidden layer network displays poor interpolative abilities which is to be expected as the decision region is bounded by linear boundries. The Radial Basis Function network performs much better, producing suitably localised classes. This increase in performance is to be expected due to the relationship between RBFs and multidimensional interpolation. The new network converges to the best solution of the three, which is extremely near to the optimal solution, given no information other than at the class points. In fact, if the above is repeated for the network before it has converged, it displays results which are similar to those of RBF networks. However the new network is not constrained to a set of *a priori* 'RBFs' and so can converge further.

3.2 Multiclass Performance

In practice it has not proved possible, in general, to use current networks for classification with more than a 26 classes. In general, multiple class problems are addressed by using a series of networks. The flexibility of the decision surfaces of the new network however allows a single network to perform multiclass problems. One such problem is that of character recognition. The network was presented with an 8 by 8 pixel image of a character and required to output the characters position in the collating sequence. The network was capable of learning to recognise all 80 characters in the font. A second order Volterra extension was used with all terms present. The coefficients were set by the LMS algorithm.

Fig.9 Some examples of the training font.

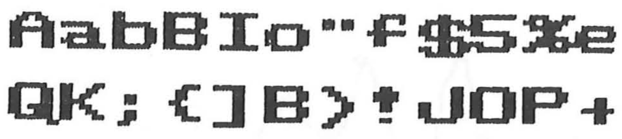
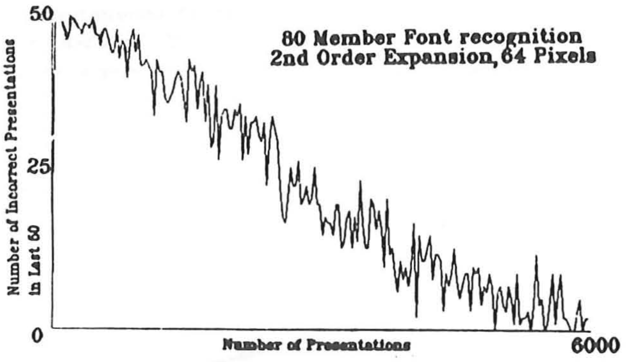
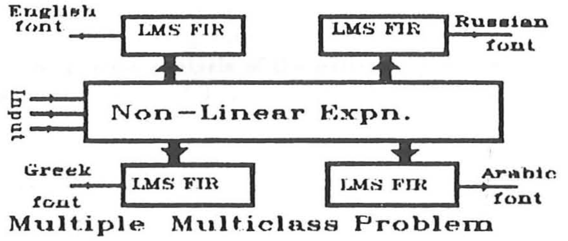


Fig.10 Plot of number of incorrect classifications in the last fifty presentations against presentation number.



It is also possible to increase the number of classes that a system can recognise by using the same vector expansion 'hardware' with multiple adaptive filter sections connected to it.

Fig.11



3.3 Implementation

In order to facilitate the understanding of the new network it has so far been discussed as two sections: firstly the vector state expansion and secondly the adaptive filter section. The same network may be implemented in a more distributed form composed of a series of identical 'neurons' by factorising the expansion polynomial.Once this has been done it is possible to find factorisations composed of identical units. Consequently a network may be created using a network of simple terms. Another approach which is currently under investigation is to define two types of 'neuron'. The first is composed of a simple multiplier and the second a standard linear perceptron. An interesting form of the second implementation is that in which the network elements are randomly connected. This generates a connectionist model containing random terms from the non-linear state expansion. However, provided that the network is large enough, the inherent degeneracy allows the network to find another possible solution without the missing terms.

$$y = w_0 + w_1x_1 + w_2x_2 + w_3x_3 + w_{11}x_1x_1 + w_{12}x_1x_2 + w_{13}x_1x_3$$
$$+ w_{21}x_2x_1 + w_{22}x_2x_2 + w_{23}x_2x_3 + w_{31}x_3x_1 + w_{32}x_3x_2 + w_{33}x_3x_3$$

may be factorised as:

$$y = w_0 + x_1(w_1 + w_{11}x_1 + w_{12}x_2 + w_{13}x_3) + x_2(w_2 + w_{21}x_1 + w_{22}x_2 + w_{23}x_3) +$$
$$+ x_3(w_3 + w_{31}x_1 + w_{32}x_2 + w_{33}x_3)$$

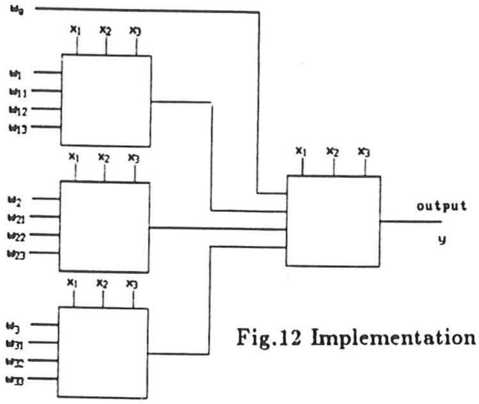


Fig.12 Implementation of Network.

4 Current Research

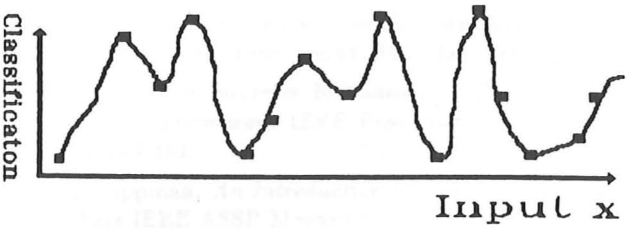
The quadratic error surface lends itself ideally to the use of a more efficient adaption algorithm, such as that of conjugate gradients [11]. A variation of this method is currently being developed, which is giving considerably faster convergence times than those discussed earlier without requiring the storage of very large matrices.

As previously mentioned the effect of various implementations is being considered, including random connection under a series of network generating rules, and the effect of limited accuracy and limited dynamic range multiplications [8].

A form of the network based on the Weiner G-functionals [2] is being considered, as these form an orthogonal set. This may lead to advantages in adaptive performance [13] for some applications and form another basis from which to analyse the network behaviour.

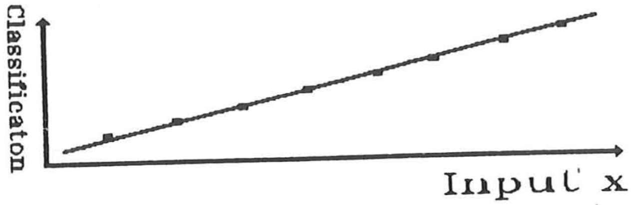
An examination of the network from a vector space perspective has made it apparent that the actual values of the outputs for various classes if suitably chosen can lead to the use of very small networks. Although an analytical solution has not yet been found, a series of good approximate solutions have been developed and yeiled some surprisingly small networks. The importance of the output values can be easily demonstrated. Consider a single input system with a series of output classes as shown below:

Fig.13



This would require a high order non-linearity and consequently complex boundries to the desicion regions and so a larger network. If the output indices had been suitably chosen the whole problem could have been reduced to a linear one:

Fig.12



Although the above is a simple case the problem becomes difficult to solve in multiple dimensions with a degenerate system. This question of output values has not been fully appreciated with current networks and a method for determining suitable indices would lead to far smaller networks. The approach being currently investigated involves a principle component analysis of the pattern difference vectors and has provided the ability to use smaller networks. It is hoped to develope an adaptive approach to network contraction.

The use of a least squares critereon for multiclass problems is not a particularly good one, other error functions are being investigated.

Ideally a pattern recognition system should construct an estimate of the underlying probability density functions which are generating the input vectors given an element from a particular class. Although, due to the expectation operator in the mean square error equation, the probability of a pattern vector occurring is taken into account by the network, it may be possible for the network to make far more complete estimations of underlying probability processes by using Bayesian methods.

5 Conclusion

The non-linear vector space expansion connectionist model has been shown to have a number of advantages over currently applied networks for application to general connectionist problems.

Firstly it convergences to its optimal solution far faster by orders of magnitude than current networks, and this convergence time does not rise as quickly as a function of the size of the pattern vectors.

Secondly it is unimodal and consequently does not suffer from the problems of local minima such as needing to be reset, getting stuck or uncertainty about it having converged. These problems can seriously affect the performance of current networks and prevent them being used in an adaptive mode while actually performing their tasks.

Thirdly, the complete mathematical derivation of the new network allows a much fuller understanding of its operation and allows direct application of adaptive filter knowledge. This allows easy parameter tuning and application of other adaptive filter update methods. It also allows the effects of approximation and implementation to be considered.

References

[1] S.Haykin, *Adaptive Filter Theory* Englewood Cliffs, NJ:Prentice-Hall, 1986.

[2] Schetzen, *The Volterra and Weiner Theories of Non-linear Systems* New York, NY:John Wiley, 1980.

[3] G.Tesauro & R.Janssens, *Scaling Relationships in Backpropogation Learning* Technical Report:Center for Complex Systems Research Univ. of Illionis.19-2-1988.

[4] P. Alper, *A Consideration of the Discrete Volterra Series* IEEE Trans. Automatic Control, vol. , Jul. 1965, pp.322-327.

[5] M.J.D.Powell, *Radial basis function approximations to Polynomials* Proc. Department of Applied Mathematics and Theoretical Physics.

[6] D.f.Spect, *Generation of Polynomial Discriminant Functions for Pattern Recognition* IEEE Trans. EC, vol. EC-16, No.3, Jun. 1967, pp.308-319.

[7] F.Yassa, *Optimality in the Choice of the Convergence Factor for Gradient Based Adaptive Algorithms* IEEE Trans. ASSP, vol. ASSP-35, No.1, Jan. 1987, pp.48-59.

[8] S.T.Alexander, *Transient Weight Misadjustment Properties for the Finite Precision LMS Algorithm* IEEE Trans. ASSP,vol. ASSP-35, No. 9, Sept. 1987, pp.1250-1258.

[9] M.Niranjan & F.Fallside *Neural Networks and Radial basis functions in classifying static speech patterns* Internal Report Cambridge University Engineering Department CUED F-INFENG TR 22(1988).

[10] D. Rumelhart & G. Hinton *Learning Representations by backpropogating errors* Nature Vol.323 9-10-1986.

[11] E.Polak, *Computational Methods in Optimisation* London, UK:Academic Press, 1971.

[12] S.Fakhouri, *Identification of the Volterra Kernels of Non-linear Systems* IEE Proc. vol.127 No.6, Nov. 1980, pp.296-304.

[13] S.Narayan, *Frequency Domain Least-Mean-Squares Algorithm* IEEE Proc. vol.69 No.1, Jan. 1981, pp.125-126.

[14] C.Nikias, *Bispectrum Estimation: A Digital Signal Processing Framework* IEEE Proc. vol.75 No.7, Jul. 1987, pp.869-891.

[15] R.Lippman, *An Introduction to Computing with Neural Nets* IEEE ASSP Magazine. Apr. 1987, pp.4-22.

[16] A.Ivakhnenko, *Heuristic Self-Organisation Problems of Engineering Cyberneticss* Automatica. vol.6 1970, pp.207-219.

[17] V.Klema & A.Laub, *The Singular Value Decomposition: Its Computation and Some Applications.* IEEE Trans. AC,vol. AC-25,No.2 . Apr. 1980, pp.164-176.

An Adaptive Filter Algorithm using Conjugate Gradient Methods

M.R.Lynch and P.J.Rayner
Cambridge University Engineering Department,
Trumpington Street, Cambridge,
CB2 1PZ, U.K.

1 Introduction

Over recent years a series of algorithms for the updating of adaptive filter coefficients have been developed. Some of these methods, notably those related to the method of Recursive Least Squares (RLS) [1] have provided significant increases in performance over the Least Mean Squares (LMS) [1] algorithm. For many applications the problems posed by the lack of performance of the LMS algorithm can be overcome by implemeting one of these other algorithms.

However, almost all of this new generation of algorithms require the storage of a matrix and a significant increase in the computational load. This is not a problem for most adaptive filters which are one-dimensional and utilise a limited number of coefficients. In the case of multidimensional adaptive filters, such as those used in image processing [7], or connectionist models, or for any adaptive filter with many coefficients however, it may be totally impractical to consider storage of a matrix containing $O(n^2)$ coefficients where n is the number of coefficients. Likewise the increase in computational load associated with current approaches may be too great.

In this paper a new algorithm is presented which gives a substantial increase in performance over the LMS algorithm but does not require the storage of a matrix and has a computational burden of approximately five times that of LMS.

1.1 The LMS Algorithm

In order to construct a new algorithm with increased performance over the LMS algorithm it is necessary to first consider the failings of the LMS algorithm. The update equation for the LMS or stocastic gradient method is [1]:

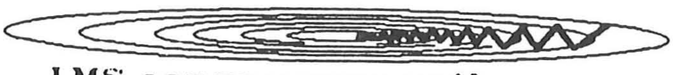
$$\mathbf{w_{k+1}} = \mathbf{w_k} + 2\mu\epsilon\mathbf{x}$$

or more generally:

$$\mathbf{w_{k+1}} = \mathbf{w_k} - \alpha\nabla(\epsilon^2)$$

Where $\mathbf{w}$ is the filter weight vector and k is the time index, α and μ are adaption coefficients, ϵ the filter error and $\mathbf{x}$ the filter input data vector. Consider an error surface which is a long narrow valley. The steepest descent type algorithms, such as LMS, will in general tend to zig-zag down the valley as shown in Fig.1. This behaviour even occurs for perfectly paraboidal surfaces and so leads to large inefficiencies in terms of convergence time.

Fig.1



LMS convergence path.

2 Derivation of the Algorithm

Consider the minimisation of a function f over an N dimensional coordinate system. The function can be approximated by the first three terms of a Taylor series of the function about a point P.

$$f(\mathbf{w}) \approx f(\mathbf{P}) + \sum_i \frac{\partial f}{w_i} w_i + \frac{1}{2}\sum_{ij} \frac{\partial^2 f}{\partial w_i \partial w_j} w_i w_j$$

$$= c - \mathbf{bw} + \frac{1}{2}\mathbf{wAw}$$

Where:

$$c = f(\mathbf{P})$$

$$\mathbf{b} = -\nabla f$$

$$[A]_{ij} = \frac{\partial^2 f}{\partial w_i \partial w_j}$$

It should be noted that the above is exact for a quadratic function. Matrix A is the second partial derivative matrix of the function f ,that is the Hessian, of f at $\mathbf{P}$. If a minimisation in the direction of vector $\mathbf{u}$ is performed it is necessary to find a direction in which to minimise so as not to destroy the previous minimisation. Consequently the next direction δx must be such that the gradient is perpendicular to $\mathbf{u}$, that is:

$$\mathbf{u}.\delta(\nabla\mathbf{f}) = \mathbf{u}.\mathbf{A}.\mathbf{v} = \mathbf{0}$$

By calculating the gradient of the above approximation with respect to the weights we obtain:

$$\nabla\mathbf{f} = \mathbf{A}.\mathbf{w} - \mathbf{b}$$

And hence, the effect on the gradient of a small change in w is:

$$\delta\nabla\mathbf{f} = \mathbf{A}.\delta\mathbf{w}$$

u and v are referred to as mutually conjugate. It is also necessary that our search directions must be mutually orthogonal so that the whole space may be spanned in the search.

Following the derivation of the Fletcher-Reeves [4] algorithm it is possible to invoke a theorem which will allow the construction of mutually orthogonal and conjugate vector sequences.

2.1 Theorem 1

If A is a symmetric, positive definite $NbyN$ matrix and $\mathbf{g_0}$ is an arbitary vector. For values of $i = 0, 1, 2......$ two series of vectors may be defined:

$$\mathbf{g_{i+1}} = \mathbf{g_i} - \lambda_\mathbf{i}\mathbf{A}\mathbf{h_i}$$

$$\mathbf{h_{i+1}} = \mathbf{g_{i+1}} + \gamma_\mathbf{i}\mathbf{h_i}$$

Where:

$$\lambda_i = \frac{\mathbf{g_i^t}\cdot\mathbf{g_i}}{\mathbf{g_i}.\mathbf{A}.\mathbf{h_i}}$$

$$\gamma_i = \frac{\mathbf{g_{i+1}^t}.\mathbf{A}.\mathbf{h_i}}{\mathbf{h_i^t}.\mathbf{A}.\mathbf{h_i}}$$

Then for all i such that $i \neq j$:

$$\mathbf{g_i^t}\cdot\mathbf{g_j} = 0$$

$$\mathbf{h_i^t}.\mathbf{A}.\mathbf{h_j} = 0$$

That is, the $\mathbf{g_i}$ are mutually orthogonal and the $\mathbf{h_i}$ are mutually conjugate. The proof of this theorem is beyond the scope of this paper but may be found in Polak's book [2], it is a form of Gram-Schmitt orthogonalisation. However, this theorem still requires storage of a matrix A. Another theorem enables this porblem to be overcome.

2.2 Theorem 2

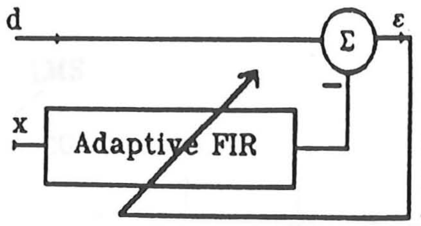
If $\mathbf{g_i} = -\nabla\mathbf{f}(\mathbf{P_i})$ and a minimisation of f in the direction $\mathbf{h_i}$ of f is performed to find a new point $\mathbf{P_{i+1}}$ and $\mathbf{g_{i+1}} = -\nabla\mathbf{f}(\mathbf{P_{i+1}})$ The same sequence is generated as in theorem 1. This may be proved by induction [2].

Consequently a series of mutually orthogonal vectors and a series of mutually conjugate vectors may be generated without knowledge or storage of A.

2.3 Application to Adaptive Filters

In order to use the above theorems it is still necessary to perform line minimisations and evaluate the gradient. The gradient evaluation may be easily performed for the adaptive filter case. For the standard adaptive filter [1] as shown below:

Fig.2 Standard Adaptive Filter



$$\epsilon = d_k - \mathbf{w}^\mathbf{t}\mathbf{x}$$

Takinǵ the mean square error:

$$E(\epsilon^2) = E(d_k^2) - 2\mathbf{wv} + \mathbf{wRw}$$

Where $\mathbf{x}$ is the vector representing the filter inputs, d is the desired response,W the weight vector,R is the autocorrelation matrix of the input and V the cross-correlation vector of the input and the desired signal. The gradient may thus be calculated as:

$$\nabla(\epsilon^2) = \frac{\partial\epsilon^2}{\partial\mathbf{w}} = 2\epsilon\frac{\partial\epsilon}{\partial\mathbf{w}} = -2\epsilon\mathbf{x}$$

The other problem is the line minimisation. The mean square error is a parabola as a function of distance along the line. Consider the line generated by:

$$\mathbf{w} = \mathbf{w} + \beta\mathbf{h}$$

Where β is the line generating scalar, the filter error can be written:

$$\epsilon = d - \sum_i x_i(w_i + \beta h_i)$$

The second derivative of the mean square error with respect to distance along the line h may be calculted as:

$$\frac{\partial^2\epsilon^2}{\partial\beta^2} = \frac{\partial}{\partial\beta}\frac{\partial\epsilon^2}{\partial\beta} = \frac{\partial}{\partial\beta}\left(2\epsilon\frac{\partial\epsilon}{\partial\beta}\right) = \left(\sum_i x_i.h_i\right)^2$$

Given expressions for the derivative, second derivative and the knowledge that the relationship is paraboidal it is possible to find:

$$\beta = \frac{\epsilon}{\mathbf{x^t}.\mathbf{h}}$$

Is the value of β at the line minimum along h. This gives only an estimate of the line minimum as the instantaneous error values could be affected by noise and incorporate no time averaging aspect. The approximation is found in practice to be sufficient for many applications. This approximation can lead to a set of mutually conjugate and mutually orthogonal vector series which use up all viable directions before arriving at the minimum . This problem can be cured by periodic reseting of the algorithm, but a much more elegant solution is to use the Polak-Ribiere[2] extension of the Fletcher-Reeves algorithm, which by using a slightly different expression for γ leads to an algorithm which in effect automatically resets the algorithm if necessary.

2.4 The Algorithm

$$\mathbf{w}_{i+1} = \mathbf{w}_i + \frac{\mathbf{h}\epsilon}{\mathbf{x}^t.\mathbf{h}}$$

$$\epsilon = d - \mathbf{x}^t.\mathbf{w}$$

$$\mathbf{g}_{i+1} = 2\epsilon\mathbf{x}$$

$$\gamma = \frac{(\mathbf{g}_{i+1} - \mathbf{g}_i)^t.\mathbf{g}_{i+1}}{\mathbf{g}_i^t.\mathbf{g}_i}$$

$$\mathbf{h}_{i+1} = \mathbf{g}_{i+1} + \gamma\mathbf{h}_i$$

With initial values:

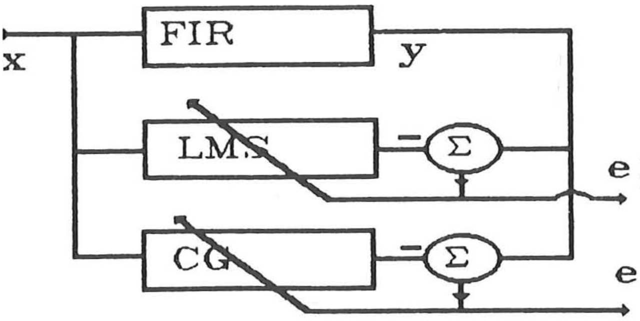
$$\mathbf{g}_0 = 2\epsilon\mathbf{x}$$

$$\mathbf{h}_0 = 2\epsilon\mathbf{x}$$

3 Results

The algorithm was tested against the LMS algorithm. Two filters, one LMS and one conjugate gradient filter (CG) were run in parallel modelling a FIR system.

Fig.3



The experiment was repeated with random coefficients n the 50_th order FIR system being modelled. The input ignal was also varied: it was composed of a series of siiusoids of random amplitude and frequency. The conver;ence curves for these runs where averaged and plotted:

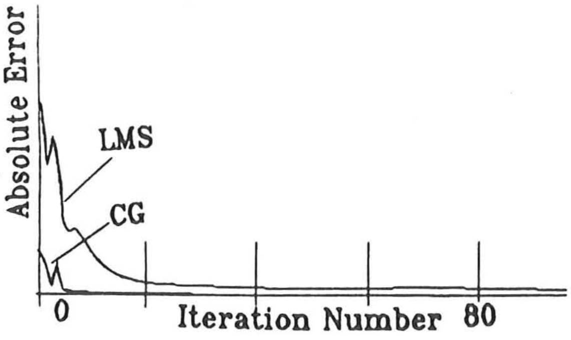


Fig.4 Plot of average absolute error values over fifty runs.

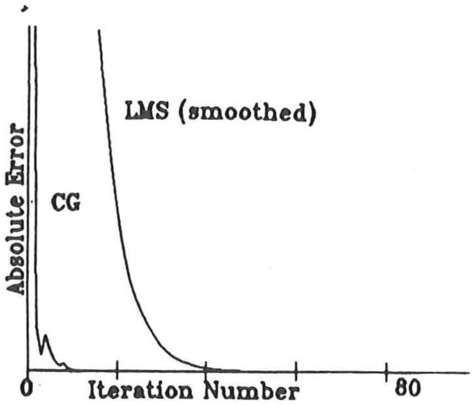


Fig.5 Typical Run with no averaging. Note smooth convergence of LMS algorithm compared with the jumps of the conjugate gradient algorithm.

4 Conclusion

The algorithm performs well, giving significant increases in convergence rates over the LMS algorithm and requires only $O(4n)$ storage rather than $O(n^2)$ of other fast methods. It is necessary to note that the basic algorithm is operating more in a Least Squares rather than a Mean Least Squares mode. This can lead to unexpected results, especially for short filters with low frequency signals and some form of error averaging may be needed to give a Mean Least Squares solution. Provided that this complication is considered, the algorithm is an improvement to LMS and may be used when methods such as Recursive Least Squares [1][6] or Singular Value Decomposition [7] cannot be applied due to storage or computational load constraints.

References

[1] S.Haykin, *Adaptive Filter Theory* Englewood Cliffs, NJ:Prentice-Hall, 1986.

[2] E.Polak, *Computational Methods in Optimisation* London, UK:Academic Press, 1971.

[3] F.Yassa, *Optimality in the Choice of the Convergence Factor for Gradient Based Adaptive Algorithms* IEEE Trans. ASSP, vol. ASSP-35, No.1, Jan. 1987, pp.48-59.

[4] R. Fletcher, *Practical Methods of Optimisation* New York, NY: John Wiley, 1987.

[5] E.Eleftheriou & D.Falconer, *Tracking Properties and Steady-state Performance of RLS Adaptive Algorithms* IEEE Trans. ASSP, vol. 34 ASSP-34, No.5, Oct. 1986, pp.1097-1109.

[6] D.Panda & A.Kak, *Recursive Least Squares Smoothing of Noise in Images* IEEE Trans. ASSP, vol. 25 ASSP-25, No.6, Dec. 1977, pp.520-524.

[7] D. Tufts & R. Kumaresan, *Data Adaptive Signal Estimation by Singular Value Decomposition of Data Matrix* IEEE Proc. Vol. 70 , No.6, Jun. 1982, pp.684-685.

An Adaptive Pitch Estimator for LMS Notch Filters.

M.R.Lynch and P.J.Rayner
Cambridge University Engineering Department,
Trumpington Street, Cambridge,
CB2 1PZ, U.K.

1 Abstract

An adaptive method of pitch estimation is presented which will estimate the pitch of a harmonic signal in noise to a high degree of accuracy, and track any slow drifts in the pitch of the harmonic signal. The approach is based on an adapitive notch filter and hence avoids the problems encountered in the use of integer length adaptive delay lines. The approach is particularly efficient when using an LMS notch filter and will allow increased performance of the notch filter by accurately fixing the reference signal frequency.

Adaptive pitch estimators for harmonic signals in noise can be difficult to design due to the nature of the error surface encountered if one attempts to adapt a fundamentally time stationary system. This approach solves this problem by utilising a time varying system.

An attempt was made to remove a harmonic interference from an archive gramophone recording using a standard FIR LMS adaptive notch filter [1]. This approach worked well in periods in which the harmonic interference was present with background noise of similar amplitude, but failed in periods with music or speech present. The amplitude of the music was many times greater than that of the interference and caused the adaptive notch filter to ring and destroyed its interference cancelling properties. As in the case encountered by Lim [3] it was decided to freeze the adaptive notch filter in these regions. However as these regions persisted for tens of thousands of samples the reference sinusoids to the adaptive notch filter had to have their pitches accurately determined so that in the frozen periods the interference and anti-interference output from the FIR filter should stay synchronised.

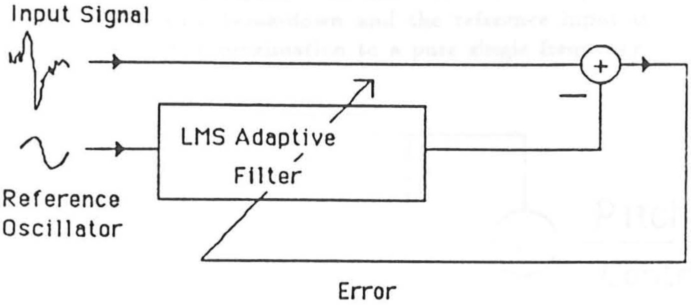


Fig1: An adaptive notch filter.

Thus an accurate and computationally efficient pitch estimator was required which could also track a slowly drifting pitch. Consider an LMS adaptive notch filter with one notch. Glover [2] analysed the LMS adaptive notch filter and showed its behaviour to include time varying aspects under certain conditions.

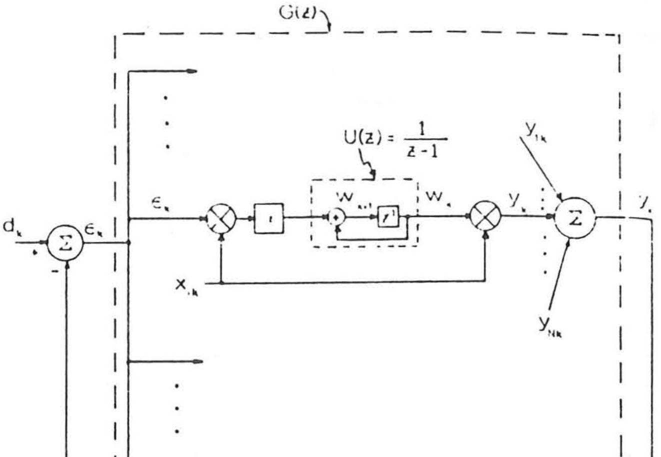


Fig 2:Notch Filter Analysis Diagram.

Let $C_i(z)$ be the Z-transform of the i_{th} coefficient of the adapted notch filter, with adaption coefficient α and reference sinusoid of frequency ω_r and amplitude C, $U(z)$ is the Z transform of the LMS weight update equation (an integrator) as shown in Fig2.

$$C_i(z) = \frac{\alpha C}{2} U(z)[E(ze^{-j\omega_r T})e^{j\theta_i} + E(ze^{j\omega_r T})e^{-j\theta_i}]$$

Glover shows that the pole zero plot of E(z) would indicate poles at $z = e^{\pm j\omega_d T}$ and $E(ze^{-j\omega_r T})$ represents a counter-clockwise rotation of $E(z)$ through angle $\omega_r T$.

Therefore the pole-zero plot of :

$$[E(ze^{-j\omega_r T})e^{j\theta_i} + E(ze^{j\omega_r T})e^{-j\theta_i}]$$

would show poles at $\pm(\omega_r + \omega_d)T$ and at $\pm(\omega_r - \omega_d)T$. This rotated spectrum is filtered through $U(z)$ to give $C_i(z)$. Each weight therefore consists of the sum and difference frequency of ω_r and ω_d. $U(z)$ is strongly low pass, hence the difference frequency dominates . For the FIR delay line $\theta_i = \theta - \omega_r T[i-1]$ which shows that the coefficients c_i are a sinusoid of frequency ω_r with each weight varying as a sinusoid at frequency $\omega_r - \omega_d$.

Glover refers to this as the sinusoid in the coefficients 'moving' at a rate equal to the difference frequency between the desired and reference frequencies. This may be viewed as a heterodyning process and is a time varying aspect to the solution for an adaptive notch filter; it may also be considered as a simple case of a quasistationary filter using a slowly varying phase reference at the desired frequency. It is this time varying aspect that the adaptive pitch estimator (APE) will use.

A single notch adaptive filter is constructed and the motion of the sinusoid is detected by the following algorithm:

$$P_{n+1} = P_n + \mu_p \frac{1}{\phi_k} \frac{\partial c_k}{\partial t}$$

Where ϕ_k is the gradient of the coefficients with respect to i at a particular coefficient k, and n is the time index. The coefficent of pitch adaption μ_p set the sensitivity and convergence rate of the pitch estimator.

The time derivative may be replaced by:

$$\frac{\partial c_n}{\partial t} \approx c_{kn} - c_{k(n-1)}$$

And the gradient by:

$$\phi_k \approx c_{n(k-1)} - c_{n(k+1)}$$

A simpler form of the algorithm which does not require the division and hence avoids any possible divide by zero problems is:

$$P_{n+1} = P_n + \mu_p sgn[\phi_k \frac{\partial c_k}{\partial t}]$$

In practice Glover's analysis does not always apply. If we consider the frequency response of the FIR stage of the adaptive notch filter we can see that there are an infinite number of possible solutions for the case of a single sinusoid. The above algorithm, however, requires the particular solution with a sinusoid in the coefficients to be adopted. If this solution is not adopted or is disrupted the system will fail. Occasionally in practice it may be necessary to force the filter to assume the sinusoidal solution. In this case we must force its stopband to include all frequencies except the reference frequency. This is easily done by injecting some white noise with the reference sinusoid. This gives a good stable sinusoid in the coefficients as the filter should block as much of this white noise as possible to minimise the mean square error.

The other two disruptions of the sinusoidal solution are a transient effect as the system is first set running and a destabilising effect caused by varying the pitch of the reference sinusoid too rapidly. In this case the assumptions of quasistationarity break-down and the reference input is no longer a good approximation to a pure single frequency.

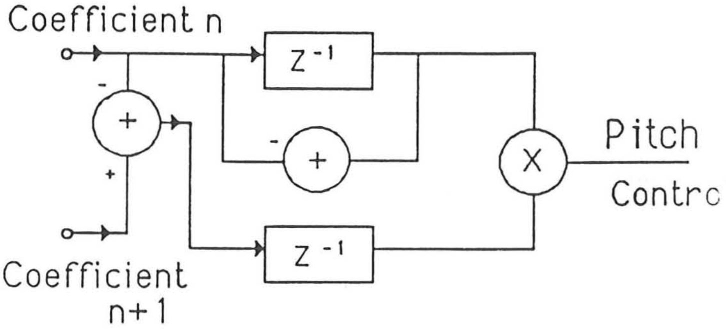


Fig.3 Coefficient Motion Detector.

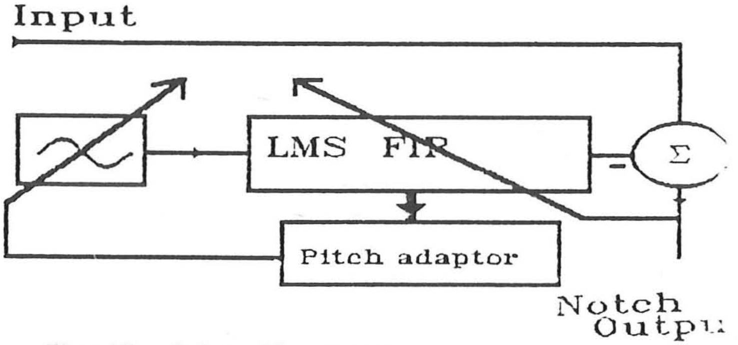


Fig4: The pitch tracking system.

1.1 Harmonic Signal Case:

In the case of harmonic signals the accuracy of the system may be increased by using a reference sinusoid of frequency near one of the higher harmonics present rather than the fundemental, as this will increase the difference frequency and increase the coefficient sinusoid movement by a factor equal to the harmonic number. Thus leading to a larger effect for the same pitch error in the fundemental.

1.2 Limitations:

The limitaions on the accuracy of the algorithm are set by the adaption noise found on the coefficient values. This adaption noise can be estimated from the expression for the minimum mean square error. It is clear that it may be reduced by reducing the adaption coefficient of the notch filter. The bandwidth of the notch filter is set by this coefficient which in turn determines the range of frequency error over which the above analysis applies. Thus there is a trade off of final pitch accuracy versus initial lock range. The initial lock range being defined as the maximum difference in frequency between the desired and reference signals for which the system will converge.

The notch bandwidth is given by [2]:

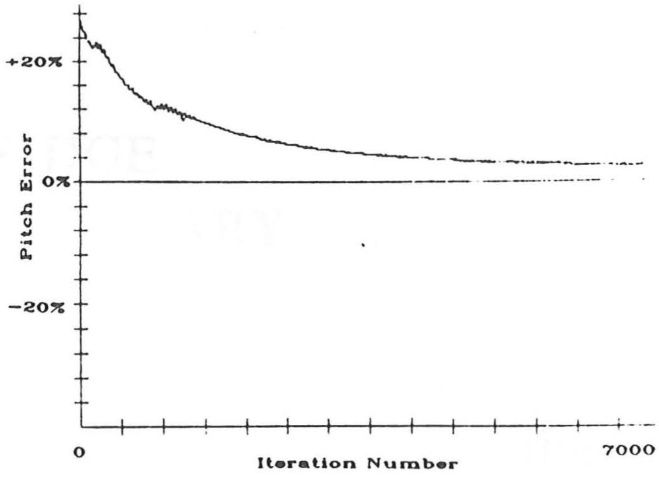
$$BW \approx \frac{N\alpha C^2}{2T} rads/s.$$

Where N is the filter length.

Limitations on the rate of convergence are set by the need prevent instability due to contravening of the quasistationary assumptions. The changes in the pitch of the reference oscillator must therefore be suitably slow, this may be acheived by using a suitably small pitch adaption coefficient μ_p.

1.3 Results:

The algorithm was found to be capable of high accuracy pitch estimates in quite high levels of noise and was capable of pitch convergence in SNR of -15dB. In the case of SNR of 10dB the system could attain pitch accuracy estimates in the order of 1 part in four thousand. A series of convergence curves are shown in Fig:5. The occasional initial divergences are caused by transient effects as the data enters fills the filter.



1.4 Conclusions:

The system is best suited to applications requiring high accuracy pitch estimates on stationary signals for which a low accuracy pitch estimate is already known. The necessary computations for the system are based around a FIR filter, thus making it easily implementable in hardware. The filter may be quite short in practice and in the case of an adaptive notch filter to remove a single sinusoid the pitch estimator may be added at very little extra computational cost by using the cancelling filter as the pitch estimating filter, this leads to deeper notches and increased performance. It is clear that there are many possibilities for variations on the above theme that may be well suited to other applications.

References

[1] WIDROW B., and GLOVER J.'Adaptive Noise Cancelling: Principles and Applications'. *Proc IEEE,1975*,**63**,pp.1692-1719.

[2] GLOVER J.'Adaptive Noise Cancelling Applied to Sinusoidal Interferences'. *IEEE Trans. ASSP.*,**ASSP-25**,December 1977,pp.484-491

[3] HARRISON W., and LIM.'A New Application of Adaptive Noise Cancellation'. *IEEE Trans. ASSP*,**ASSP-34**,February 1986,pp. 21-27.