(273601

UNIVERSITY OF STRATHCLYDE, GLASGOW

ACCELERATED GRADIENT TECHNIQUES AND ADAPTIVE SIGNAL PROCESSING

MOHAMMAD YAMINYSHARIF

DEPARTMENT OF ELECTRONIC AND ELECTRICAL ENGINEERING

DOCTOR OF PHILOSOPHY 1987

True riches are spiritual, not material.
Maturity a matter of intelligence, not of years.

(SAADI)

ABSTRACT

The main objective of this thesis is to demonstrate the application of the accelerated gradient techniques to various fields of adaptive signal processing.

A variety of adaptive algorithms based on the accelerated gradient techniques are developed and analysed in terms of the convergence speed, computational complexity and numerical stability.

Extensive simulation results are presented to demonstrate the performance of the proposed algorithms when applied to the fields of adaptive noise cancelling, broad band adaptive array processing and narrow band adaptive spectral estimation. These results are very encouraging in terms of convergence speed and numerical stability of the developed algorithms.

The proposed algorithms appear to be attractive alternatives to the conventional recursive least squares algorithms.

In addition, the thesis includes a review chapter in which the conventional approaches (ranging from the least mean squares algorithm to the computationally demanding recursive least squares algorithm) to three types of minimization problems (namely unconstrained, linearly constrained and quadratically constrained) are discussed.

ACKNOWLEDGEMENTS

The author wishes to thank the UK Science and Engineering Research Council for supporting this research and Professor T.S. Durrani for any advice given. Finally,special thanks are given to my wife Carolyn for typing this thesis while awaiting the birth of our first child.

LIST OF SYMBOLS

AG	Accelerated gradient
AGA	Accelerated gradient algorithms
ANC	Adaptive noise cancelling
AR	Autoregressive
BD	Backward
BICGM	Block data implementation conjugate gradients
BIELS	BLock data implementation exact linear search
CD	Computationally demanding
CE	Computationally efficient
CGM	Conjugate gradients method
DFT	Discrete Fourier transform
DS	Directional spectra
DSD	Differential steepest descent
EED	Eigenvalue eigenvector decomposition
ELS	Exact linear search
EVR	Eigenvalue ratio
FD	Forward
FIR	Finite impulse response
fl	Floating point
IIR	Infinite inpulse response
LMS	Least mean square
LMSE	Least mean squared error
LPC	Linear prediction coding
LRS	Linear random search
LSL	Least squares Lattice

MMSE	Minimum mean squared error
PGA	Projection gradient algorithm
RCs	Reflection coefficients
RLS	Recursive least squares
SGD	Stochastic gradient descent
SNR	Signal to noise ratio
TB	Transform based
TDL	Tap delay line
VLSI	Very large scale integration
E(.)	Expectation
G(W)	Gain matrix
$\nabla(.)$	Gradient (derivative)
$\nabla^2(.)$	Second derivative
tr(R)	Trace of R, R = a matrix
$\lvert . \rvert$	Vector norm
$\lvert . \rvert^2$	Second norm (two norm, Euclidean norm)
$.^t$	Transposition
$.^+$	Complex conjugate
$.^*$	Optimum
L	Denotes the dimension
$\lvert W \rvert$	$(W^tW)^{\frac{1}{2}}$, W = a vector
$\lvert W \rvert^2$	W^tW , W = a vector
J(.)	Function of (.)
I	Identity matrix
$\triangleq$	By definition

∇	Upper triangular matrix
$\triangle$	Lower triangular matrix with the exception of the diagonal terms
$\frac{\delta}{\delta a_K}$	The gradient with respect to a_K
$\nabla J(.)_W$	Gradient of the function J(.) with respect to W

In addition, various English letters and mathematical symbols are used throughout this thesis, the meaning of which is clearly explained in the corresponding paragraph, formula or section.

CONTENTS

CHAPTER 1

INTRODUCTION

1.1 Definition

<u>Definition</u>: As an initial definition of adaptive systems, one refers to the systems that can automatically adapt in the face of changing environments. In other words, adaptive systems are systems which can adjust their structure in order to meet some desired criterion. Adaptive systems can be trained to perform specific filtering and decision-making. Adaptive systems have been variously referred to as self-optimising systems, learning systems, updating systems, programmable systems etc. and by their very nature are time varying systems. These systems involve an iterative procedure to achieve the desired performance objective, and it is during this iterative period that the system is said to be "adapting" or "learning" or "self optimizing".

Current applications for adaptive systems include such fields as communications, radar, sonar, seismology, mechanical design, navigation systems and biomedical electronics. The above list is by no means exhaustive and despite a wide and ever increasing range of applications of adaptive systems, this thesis is concerned only with types of systems designed primarily for the purposes of adaptive signal processing.

1.2 Historical review

The origins of adaptive systems can be traced back to Newton; the root finding algorithm for a polynomial is the first ever implementation of the steepest descent technique. Similarly the minimization via the Newton-Raphson technique is another manifestation of an adaptive computing technique. It was during the 1960's that the subject of adaptive processing became an important research topic and since then adaptive processors have subsequently been applied in many practical systems mainly as adaptive filters or adaptive antennas.

Theoretical work on adaptive filters (processors) was reported in 1961 in the United States by Glaser and in the same year in the United Kingdom by Gabor et al. Other notable early developments occurred at the Technische Hochschule Karlsruhe in Germany and at Standford University, where adaptive pattern recognition systems were initiated in 1959. Collaboration in 1964 between these institutions produced a comparative evaluation of their respective techniques [Steinbuck and Widrow] which subsequently led to the development of the most widely used algorithm; the least mean square algorithm for processor weight adjustment. Further relevant work was being conducted simultaneously at the Institute of Automatics and Telemechanics in Moscow. In the early and middle 1960's, work on adaptive systems intensified. Hundreds of papers on adaptation, adaptive controls, adaptive filtering, and adaptive signal processing (the main objective of this thesis) appeared in the literature. An excellent summary of the status, in the middle 1960's, of adaptive processors (adaptive filters) and early relevant references for their use in adaptive equalization is provided by Rudin [1]. More recently, simple review articles have been prepared on echo cancellation in telephony [2] and adaptive equalization [3],[7].

The author does not propose to outline the detailed evolution of adaptive systems over a period of three decades, but to give a brief historical review of this important subject. Today adaptive systems are of paramount importance and they are being applied to a wide variety of scientific, engineering, economic and biomedical problems. Extensive studies have been conducted in the field of adaptive systems. Hundreds of published papers as a result of three decades of research on this subject, form comprehensive and useful research literature. In recent years a few text books [4-9] have been published which provide a coherent and comprehensive introduction to the subject of adaptive systems covering the basic theory, practical realization and current applications.

1.3 Conventional approaches to the problems in the field of adaptive systems and a review of the existing limitations

This section provides an essay type introduction to various conventional adaptive techniques and the existing trade-off between them.

The main objective of all adaptive algorithms is to determine the minimum of a performance surface by means of an iterative searching procedure. Most practical procedures capable of searching a performance surface to find its optimum point are based on the method of steepest descent or Newton's method and employ the gradient estimates to indicate the direction in which the minimum of the surface lies. They are thus referred to as "descent methods". In addition to these descent techniques, there exist another class of algorithms referred to as random search algorithms which can be employed to determine the minimum of a performance surface. The random search algorithms search a performance surface in a random pattern for its minimum.

The method of steepest descent can readily be implemented and has proven its value in a wide variety of practical applications. Generally speaking, the technique involves measurements of the gradient of the objective function (performance surface). Since in most applications an exact measurement of the gradient is not available, an estimate of the gradient based on a limited statistical sample is used. Two general methods of estimating the gradient are the "derivative measurement" and the "instantaneous measurement" [5, 8, 10]. The former technique known as differential steepest descent (DSD) algorithm, is straightforward and easy to implement and involves the direct measurement of the derivatives which are estimated numerically by taking symmetric differences. The latter method is the basis for the least mean square (LMS) algorithm and requires specific knowledge about the nature of the performance surface (unlike the DSD method which requires only very general knowledge of the performance surface). The DSD method is less efficient (in terms of speed of adaptation and misadjustment - a dimensionless measure of the difference between the actual and approximated optimum point) but more economical than the LMS algorithm and it is used where technical or economical considerations preclude use of the LMS algorithm or where high speed of adaptation is not required [5, 8, 10].

However, it is the LMS algorithm that is generally regarded as the best choice for many different applications of adaptive signal processing. The LMS algorithm is important because of its simplicity and ease of computation, thus it is an attractive solution for many practical problems. The main disadvantages of the LMS algorithm are related to its convergence properties. Although the LMS algorithm offers higher speed of adaptation with respect to the DSD algorithm, it is much slower than other computationally demanding techniques (such as those techniques based on Newton's method) particularly during ill condition situations. This is a major disadvantage of the LMS algorithm.

Newton's method is another standard descent method for finding a minimum of an objective function and offers a higher speed of adaptation (unlike steepest descent method). It is a computationally demanding technique and involves matrix inversion at each iteration cycle (each step in the search procedure). Thus it is frequently difficult to implement in practice. Newton's method which is a gradient search technique requires knowledge of the first and second derivative of the objective function. In practical applications, however, an exact measurement of the first derivative (the gradient) and the second derivative (Hessian matrix) of the objective function is not available and estimates based on a limited statistical sample are used. The algorithms that approximate Newton's method are referred to as Recursive Least Squares (RLS) algorithms. In spite of their superior performance (with respect to LMS algorithm), their use in adaptive signal processing applications has been relatively limited, due to their higher computational requirements. In recent years there has been a renewed interest in RLS techniques, especially in their "fast" (computationally efficient) versions, which have been applied to many problems of adaptive signal processing.

As stated earlier, the third possibility of searching a performance surface for its minimum is based on Random search procedures. Unlike Newton's and the steepest descent method that are systematic surface search procedures, the Random search techniques, by contrast search for the minimum point by making random changes (in random directions or by selection of random points). Two practical implementations of Random search methods are the Linear random search (LRS) and the Genetic optimizer (GO) techniques [5, 8, 10].

Although use of random search techniques may be appropriate in cases where the performance surface for the adaptive process is not well behaved and has both local and global optima, they are less efficient than techniques based on the Newton's or the steepest descent methods, thus they are not an attractive choice for various practical applications [5, 8, 10].

So far, various techniques of finding the optimum point of an objective function have been outlined along with their fundamental advantages and disadvantages. Generally one can divide these techniques into two types considering the computational complexity and the speed of adaptation. Among various computationally efficient techniques (eg. LMS, DSD, Random search methods), the LMS algorithm is regarded as the best choice, due to its simplicity and efficiency and is a standard technique in adaptive signal processing whenever simplicity is a prime objective and not the speed of adaptation. On the other hand, for applications where speed of adaptation is of primary importance the RLS technique, which is computationally demanding, is more popular. These two techniques (LMS, RLS) are the basis of algorithm design for adaptive signal processing - their fundamental trade-off being the computational complexity against the speed of adaptation. In other words, the most popular algorithms used in signal processing are either computationally efficient - primarily based on the LMS algorithm - or are computationally demanding - primarily based on the RLS algorithm- but are often not both.

The existing trade-off has been a fundamental issue in the development of new algorithms ie. algorithms that exhibit both high speed of adaptation and low computational complexity, and this forms the main issue emphasised throughout this thesis.

1.4 The scope of the thesis

Having discussed various techniques used in adaptive signal processing, their advantages and disadvantages, it remains to outline the contribution that this thesis intends to make.

The second chapter of the thesis presents a brief development and analysis of all currently used adaptive algorithms ranging from the simple LMS algorithm to computationally demanding RLS algorithms. Both TDL and Lattice realizations of the above algorithms are outlined along with their advantages and disadvantages.

Chapter three presents the main contribution of this thesis. In this chapter new adaptive algorithms are presented which are based on the accelerated gradient algorithms (AGA). Various adaptive AGA are established and analysed which match the performance of RLS algorithms. The proposed algorithms have better numerical stability in comparison with RLS techniques.

In chapter four the numerical stability of the AGA is discussed with a view to the effects of finite word length implementation.

Chapter five presents the results. Results are obtained by simulations performed on the VAX/VMS computer using single precision FORTRAN. Various conventional and recently developed algorithms are simulated and their relative performance in terms of speed of adaptation and computational efficiency are compared as applied to the fields of adaptive noise cancelling, adaptive array processing and adaptive spectral estimation.

The concluding chapter of this thesis is chapter six.

CHAPTER 2

FUNDAMENTALS AND CONVENTIONAL TECHNIQUES

2.1 Introduction

This chapter presents the basic aspects of adaptive signal processing. Three types of frequently arising minimization problems - namely; unconstrained, linearly constrained and quadratically constrained problems - in adaptive signal processing are considered.

Various conventional approaches to these problems are discussed along with their advantages and disadvantages.

2.2 Adaptive processors

Adaptive filters (processors) can be realized as adaptive finite impulse response (FIR), adaptive infinite impulse response (IIR) and transform based (TB) adaptive filters.

The IIR adaptive filters, though more economical (than FIR counterparts) to use in some applications, have had limited application because of the stability problem. TB adaptive filters, once again more economical than FIR versions, are also shown to introduce problems due to the introduction of circular convolution and block processing in the discrete fourier transform (DFT). By far, the most important adaptive filters are FIR versions, which are fundamental to adaptive signal processing and appear in most adaptive systems in one form or another. Because of their simple structure, they are easy to understand and analyse and offer the best performance (with respect to IIR and TB adaptive filters) in most applications.

The structure of the non-recursive adaptive filter (ie. FIR) is shown in Figure 2.1 , which is also known as the adaptive linear combiner.

The processor presented in Figure 2.1 is used in a wide range of applications in adaptive signal processing. It is simply an adaptive version of a digital non-recursive filter in which tap weights $w_1, w_2, w_3 \ldots\ldots w_L$ can be adjusted by an adaptive algorithm. Its input-output relation at each iteration K can be written using vector notations,

$$y_K = X_K^t W_K = W_K^t X_K \qquad (2.1)$$

where, y_K = output signal,

$W_K^t = (w_1, w_2, w_3 \ldots\ldots w_L)$

$X_K^t = (x_K, x_{K-1}, x_{K-2} \ldots\ldots x_{K-L})$

X_K is the data vector, W_K is the weight vector and t denotes transposition.

2.3 Optimum linear estimation

The concepts of optimum linear estimation are fundamental to any treatment of adaptive filters. Adaptive filtering problems involve two estimation procedures; first, estimation of the required filter output and secondly, estimation of the filter weights required to achieve the former objective. The second of these two procedures is required because the input signal characteristics are not known apriori in the adaptive filtering situation. In a previous section, the importance of FIR adaptive filter structures, which are the most commonly used structures in adaptive systems, was emphasised. Thus, in this section the theoretical basis of optimum non-recursive estimation, known as Wiener-Hopf estimation, is established. A detailed explanation of optimum recursive estimation is presented in [7].

In a non-recursive estimation, the estimate y_K is defined in terms of a finite linear polynomial in x_K,

$$y_K = \sum_{i=1}^{L} x_{K-i} w_i \tag{2.2}$$

where w_i are individual weights in the non-recursive FIR filter illustrated in Figure 2.2. Equation (2.2) may be written in vector notation as,

$$y_K = X_K^t W = W^t X_K \tag{2.3}$$

Because of its mathematical tractability the minimum or least mean squared error (MMSE) or (LMSE) criterion is the most common cost function used in adaptive systems. Thus, regarding Figure 2.2, the MMSE function becomes

$$E(e_K^2) = E(d_K - W^t X_K)^2 \tag{2.4}$$

where, d_K = desired signal and E denotes the expectation.

The equation (2.4) presents a well behaved quadratic performance surface with a single unique minimum. Differentiating (2.4) with respect to W^t yields,

$$\nabla E[e_K^2] = 2E(X_K X_K^t W - d_K X_K) \qquad (2.5)$$

and setting (2.5) equal to zero results in,

$$E(d_K X_K) = E(X_K X_K^t W) \qquad (2.6)$$

Assuming that the weight vector W and the signal vector X_K are uncorrelated, then

$$E(d_K X_K) = E(X_K X_K^t) W^* \qquad (2.7)$$

or

$$P = RW^* \qquad (2.8)$$

where, $P = E(d_K X_K)$ is the cross-correlation between the input signal and the desired signal, $R = E(X_K X_K^t)$ represents the autocorrelation matrix of the input signal sequence which is a symmetric and positive definite matrix, and W^* = the optimum weight vector. The equation (2.8) is known as the Wiener-Hopf equation. The optimum MMSE or the Wiener solution for W is given by,

$$W^* = R^{-1} P \qquad (2.9)$$

To obtain the Wiener solution, (2.9), conventionally requires the inversion of a Toeplitz matrix in addition to the evaluation of many cross and autocorrelation functions. This requires an enormous amount of computing, even for one solution vector of tap weights, without the added complication of having to repeat the process every time an update is required due to the changing input statistics. This difficulty leads to the necessity of designing adaptive algorithms. Algorithms with low computational requirements that can obtain optimum or near optimum solution.

2.4 Stochastic approximation estimation

The overall aim of stochastic approximation is a convergence onto the true parameter values by some search method, sometimes referred to as hill climbing or valley-descending on some defined error surface. All stochastic approximation methods are of the form,

$$W_{K+1} = W_K - G\,(W)\nabla J(W) \qquad (2.10)$$

where, $\nabla J(W)$ is the partial derivative with respect to the parameter vector W_K at the K-th step of iteration. This derivative represents the gradient of a performance surface. Use of equation (2.10) therefore involves three steps:

1. The definition of a performance criterion sometimes termed the loss function or cost function and its form is quite arbitrary and may be chosen as the most appropriate for any specific problem.

2. The determination of the derivative of the performance surface. The method usually assumes that $J(W)$ has a single extremum at $W = W^*$, and that $J(W)$ is continuously twice differentiable with respect to W. In this case the optimal value $W = W^*$ will satisfy the vector equation, $\nabla J(W^*) = 0$.

3. The definition of a gain matrix $G(W)$ which determines the weight to be placed on the gradient of the performance criterion. This is a function of the parameter values and various forms may be chosen, the problem being to choose the best form to achieve most rapid convergence to optimal (minimal) value of $J(W)$.

The choice of stochastic approximation schemes therefore reduces to a problem of choosing between the three elements; the form of the performance criterion, method of calculating the derivative of the performance surface and the choice of the gain matrix. Extended discussion of the stochastic approximation is outside the scope of this section and for a detailed treatment of the subject see [11, 12]. However, the most important element in the method is the choice of the optimal gain matrix $G(W)$. There are a number of useful choices of the matrix gain that can be made, the major ones being;

1. Steepest descent; the gain is set to,

$$G(W) = aI \tag{2.11}$$

where, a is a scalar constant and I is an identity matrix.

2. Newton-Raphson method; the gain is set to the inverse of the second derivative with respect to the parameter, W, of the performance surface J(W), [11, 12].

2.5 Quadratic function

A function expressable in the form,

$$J(W) = W^t RW - 2P^t W + c \tag{2.12}$$

where, R is a L by L symmetric matrix and c is a scalar constant, is called a quadratic function.It is convenient to say that J(W) is a positive definite quadratic function when R is a positive definite matrix. Let W^* be a solution of the linear equation

$$RW = P \tag{2.13}$$

Then $RW^* = P$ and a simple calculation yields the formula

$$J(W) = J(W^*) + (W-W^*)^t \, R(W-W^*) \tag{2.14}$$

Suppose that R is a positive definite matrix. Then considering, (2.14), $J(W) > J(W^*)$ whenever $W \neq W^*$ so that W^* is the unique point of J(W). Moreover the level surfaces(J(W) = constant) of J(W) are (L-1) dimensional ellipsoids having W^* as their common centre, as shown schematically in Figure 2.3 for the cases L = 2 and L = 3. When L = 2 the level surfaces of J(W) are curves and are in fact ellipses [12, 13, 14]. Positive definite quadratic functions play a significant role in analysis of adaptive systems and a general knowledge of their properties is an essential background requirement. A more complete account of quadratic functions is given in [12, 13, 14].

2.6 Unconstrained minimization problem and conventional adaptive algorithms

This section presents the application of the conventional adaptive algorithms to the unconstrained minimization problem. The problem can be outlined as;

$$\text{minimize} \quad J(W) = W^t R W - 2P^t W + c \tag{2.15}$$

The function of J(W) has a unique minimum point (if R is a positive definite matrix) which is expressible as;

$$W^* = R^{-1}P \tag{2.16}$$

2.6.1 The steepest descent method

The steepest descent method is an iterative scheme based on the gradient of the cost function J(W) and can be characterized by the equation;

$$W_{K+1} = W_K - a\nabla J(W_K) \tag{2.17}$$

where, a = the convergence factor or the step length, and

$$\nabla J(W_K) = 2RW_K - 2P \tag{2.18}$$

$\nabla J(W_K)$ denotes the gradient of the $J(W_K)$ at iteration cycle K. For any arbitrary initial value W, the equation (2.17) converges to the optimal tap weight vector W^*. The algorithm thus includes a correction term at each iteration which is proportional to the gradient of $J(W_K)$, and represents a step taken in the direction of the gradient of $J(W_K)$ where the step size is controlled by a.

Defining $V_K = W_K - W^*$ and using equation (2.18), (2.17) can be expressed as,

$$V_{K+1} = (I - 2aR)V_K \qquad (2.19)$$

where, I represents the identity matrix. To see a more precise behaviour of the error vector, V_K, equation (2.19) may be decoupled into L independent equations by applying a similarity transformation [5, 8].

Since the covariance matrix (autocorrelation matrix), R , is symmetric and positive definite, it can be decomposed into,

$$R = UDU^t \qquad (2.20)$$

where, U is an orthonormal matrix of eigenvectors of R,and D is a diagonal matrix containing its eigenvalues such that,

$$D = \mathrm{diag}[m_1, m_2, m_3 \;\cdots\cdots\; m_L] \qquad (2.21)$$

References [13-15] give a good introduction to the underlying theory of matrices, eigenvalues, eigenvectors, similarity transformation, etc. Utilizing equation (2.20) in equation (2.19) leads to,

$$U^t V_{K+1} = (I - 2aD) U^t V_K \tag{2.22}$$

Considering the similarity transformed vector,

$$U^t V_K = V'_K \tag{2.23}$$

then, equation (2.22) can be written as,

$$V'_{K+1} = (I - 2aD) V'_K \tag{2.24}$$

Because the eigenvalue matrix, D, is a diagonal matrix, equation (2.24) represents a set of L equations and the l-th equation can be written as,

$$v'^{l}_{K+1} = (1 - 2am_l) v'^{l}_K \tag{2.25}$$

Obviously equation (2.25) will converge if

$$0 < | 1 - 2am_l | < 1 \tag{2.26}$$

for all eigenvalues, m_l, l=1, 2, ... L, of D. The condition is most certainly satisfied if,

$$0 < a < \frac{1}{M} \tag{2.27}$$

where, M = largest eigenvalue of R and represents the worst case requirement for equation (2.25). Equation (2.25) is an extremely important relationship as it yields the bounds within which the step length parameter, a, should lie for the iterative algorithm (2.17) to converge. Condition (2.27) is necessary and sufficient for convergence of the steepest descent algorithm with a quadratic performance surface. If condition (2.27) is satisfied it follows that

$$\lim_{K \to \infty} V'_K = 0 \qquad (2.28)$$

and

$$\lim_{K \to \infty} W_K = W^* \qquad (2.29)$$

Equation (2.25) also provides an indication of the rate at which the various modes of the error equation (2.19) decay to zero. Defining the time constant of the l-th mode by T_l, (assuming that $am_l \ll 1$), then,

$$1 - 2am_l = e^{\frac{-1}{T_l}} = 1 - \frac{1}{T_l} \qquad (2.30)$$

or

$$T_l \simeq \frac{1}{2am_l} \qquad (2.31)$$

Therefore, the longest time constant involved in the error system is given by

$$T_{max} = \frac{1}{2am} \tag{2.32}$$

where, m = the minimum eigenvalue of the covariance matrix R. From equation (2.27) and (2.32) one concludes that,

$$T_{max} > \frac{M}{2m} \tag{2.33}$$

In other words, the larger the eigenvalue spread of the covariance matrix R, the longer it will take for the steepest descent method to converge [4-10, 13, 14].

2.6.2 The stochastic gradient descent algorithm, LMS algorithm

The steepest descent method described above required the knowledge of the exact gradient of the cost function J(W). However, in practice an exact gradient will not be known apriori and an estimate of the gradient should be used. A number of algorithms have been developed (eg. stochastic gradient descent SGD algorithm [5, 8, 10], DSD algorithm and LMS algorithm) that employ an estimate of the gradient. The most popular of these is called the LMS algorithm first proposed by Widrow [10] in the context of adaptive signal processing. The LMS algorithm uses an instantaneous estimate of the gradient, ie.

$$\nabla \bar{J}(W_K) = 2X_K X_K^t W_K - 2d_K X_K \tag{2.34}$$

which can be written as;

$$\nabla\tilde{J}(W_K) = -2e_K X_K \tag{2.35}$$

Equation (2.35) represents an instantaneous estimate of the gradient. Substituting equation (2.35) in equation (2.17) results in,

$$W_{K+1} = W_K + 2ae_K X_K \tag{2.36}$$

which is the well known LMS algorithm [10].

The LMS algorithm has several attractive features. It is simple and easy to implement, it requires no apriori information or data storage and is well suited for real time processing. On the other hand, the algorithm can be very slow to converge (will be clarified afterwards) requiring a long data sequence to compute the coefficients W^*. Note that the gradient estimate in (2.35) can easily be shown to be unbiased when the weight vector is held constant ie.,

$$\begin{aligned} E[\nabla\tilde{J}(W)] &= -2E(e_K X_K) \\ &= -2E(d_K X_K - X_K X_K^t W) \\ &= 2(RW - P) \\ &= \nabla J(W) \end{aligned} \tag{2.37}$$

Since the mean value of the gradient estimate, (2.35) is equal to the true gradient $\nabla J(W)$, one could convert the LMS algorithm into a true steepest descent algorithm, at least in the limiting case, by estimating the time-averaged stochastic gradient but hold the weight vector constant for a block of data. This procedure results in an algorithm known as the stochastic gradient descent algorithm which can be expressed as,

$$W_{K+1} = W_K - a\nabla J(W_K)$$

$$\nabla J(W_K) = -\frac{1}{N}\sum_{i=(K-1)N+1}^{KN} e_i X_i \qquad (2.38)$$

where, K = number of iterations, N = the averaging time interval, and $\nabla J(W_K)$ is the stochastic gradient of the time-averaged mean square error.

$$\overline{(e_K^N)^2} = \frac{1}{N}\sum_{i=(K-1)N+1}^{KN} e_i^2 \qquad (2.39)$$

The stochastic gradient descent algorithm has the advantage of using a more realistic estimate of the gradient of the mean squared error than the LMS algorithm with a consequently increased computational cost. Note that for N = 1, the algorithm reduces to the LMS algorithm.

A key issue in the analysis of any stochastic algorithm is the question of convergence. Extensive studies have been conducted on the convergence properties of the LMS algorithm [4-10, 16-26] which provide a comprehensive research literature useful for understanding the basic performance characteristics and properties of the algorithm. At this stage it is appropriate to study the fundamental convergence properties of the LMS algorithm and outline some recently published results.

2.6.3 Convergence analysis for the LMS algorithm

Referring to the LMS algorithm, the updating vector is given by,

$$W_{K+1} = W_K + 2ae_K X_K \tag{2.40}$$

Defining an error vector at iteration K,

$$V_K = W_K - W^* \tag{2.41}$$

and utilizing ensemble averaging, equation (2.40) can be written as

$$E(V_{K+1}) = E((I - 2aX_K X_K^t)V_K) \tag{2.42}$$

Assuming that X_K and V_K are statistically independent, then,

$$E(V_{K+1}) = (I - 2aR)E(V_K) \tag{2.43}$$

which indicates that the mean of the error vector behaves exactly as if the true gradient vectors were known. $R = E(X_K X_K^t)$ is a symmetric positive definite matrix and it can be expressed as $R = U^t DU$. Utilizing the same approach as (2.18 - 2.24) yields the well known bounds for the step length a,

$$0 < a < \frac{1}{M} \tag{2.44}$$

where, M = the largest eigenvalue of the covariance matrix R.

Within bounds (2.44), the speed of adaptation and also the noise in the weight vector solution are determined by the size of a. Since M cannot be greater than the trace of R (ie. sum of diagonal elements of R), the bounds on a can be restated as,

$$0 < a < \frac{1}{tr(R)} \tag{2.45}$$

which is much easier to apply than (2.44) since tr(R) can easily be estimated.

For an adaptive filter, the variance of the filter output is often used as a measure of its performance, which can be expressed as,

$$\begin{aligned} E(J(W_K)) &= E(e_K^2) = J(W^*) + V_K'^{t} D V_K'^{t} \\ &= J(W^*) + \sum_{i=1}^{L} m_i (v_K'^{i})^2 \end{aligned} \tag{2.46}$$

(assuming stationary data and no correlation between X_K and V_K), where, m_i = i-th eigenvalue of the covariance matrix R and $v_K'^{i}$ = i-th element of the transformed vector $V_K' = U^t V_K$ [5-9]. Utilizing the same approach illustrated by equations (2.20-2.26), (2.30-2.31), it can be shown that the elements of V_K' approach zero at exponential rates given by,

$$T_i^{V'} = \frac{1}{2am_i} \tag{2.47}$$

which is the time constant associated with the i-th mode. Since $J(W_K)$ involves a sum of the squares elements of V_K', it will decay at an exponential rate that is twice the rate of decay of V_K. Thus, the time constant for the i-th mode is,

$$T_i^J = \frac{1}{4am_i} \tag{2.48}$$

Due to the noisiness of the coefficient error vector V_K, the actual output variance will be larger than the predicted one given by (2.46) which results in a factor known as misadjustment. Considering the error vector V_K, the cost function can be expressed as,

$$E(J(W_K)) = J(W^*) + E(V_K^t X_K X_K^t V_K) \tag{2.49}$$

Assuming independent vectors X_K and V_K, equation (2.49) can be written as,

$$E(J(W_K)) = J(W^*) + tr(RE(V_K V_K^t)) \tag{2.50}$$

It is shown in [5, 7, 8] that $E(V_K V_K^t) = aJ(W^*)I$, where I = identity matrix. Thus,

$$\begin{aligned} E(J(W_K)) &= J(W^*) + aJ(W^*)tr(R) \\ &= J(W^*) + aJ(W^*)\sum_{i=1}^{L} m_i \end{aligned} \tag{2.51}$$

The misadjustment is defined as the ratio of the excess mean squared error to the minimum mean squared error [5-9].

$$\text{misadjustment} = \frac{E(J(W_K)) - J(W^*)}{J(W^*)}$$

$$= a\,\text{tr}(R) \tag{2.52}$$

Obviously the misadjustment is directly proportional to the step size parameter, a, indicating a trade-off between the misadjustment and the rate of adaptation. For design purposes it is useful to express the misadjustment factor in terms of the filter order and the speed of adaptation. An alternative relation for the misadjustment factor can be obtained in terms of the average settling time of the cost function and the number of filter coefficients [5-9],

$$\text{misadjustment} = \frac{L}{4T^J_{avg}} \tag{2.53}$$

Note that the above analysis was carried out for stationary data and assuming statistically independent input vectors. For most applications, the independence assumption is absolutely unjustified; however, it does simplify the analysis considerably. Results obtained under this assumption seem to agree with experimental results using small step size [7-8, 16, 21-23,25, 26]. Convergence analysis of the LMS algorithm with correlated data (stationary or non-stationary data) is presented in [17-20, 24], and is much more complex than the results outlined above. References [19, 20] provide the results for stationary and correlated data, and results for non -stationary and correlated data are given in [17-18].

From the results outlined so far, it is clear that the convergence factor, a, plays a crucial role in the performance of the LMS algorithm. The choice of, a, involves a trade-off between the speed of adaptation and the steady state of misadjustment. Large a results in; high speed of adaptation and large steady state misadjustment. Small a results in; low steady state misadjustment and low speed of adaptation.

Gersho [22] has shown that the fastest convergence takes place (channel equalization, assuming stationary data and independent data vectors) for,

$$a = \frac{2}{m+M} \tag{2.54}$$

where, m and M denote the minimum and the maximum eigenvalues of R. However, in recent works [7, 27] it has been shown that the step length a must be restricted to an interval significantly smaller than the domain stated by relation (2.44), and it was shown that stability is ensured if and only if,

$$0 < a < \frac{1}{3M} \tag{2.55}$$

The step length, a, is not the only factor affecting the performance of the LMS algorithm and there is a second factor affecting its performance which is the spread of the eigenvalues of the data covariance matrix. As in the case of the steepest descent algorithm the maximum time constant is (see equation 2.33),

$$T^{V'}_{max} > \frac{M}{2m} \tag{2.56}$$

Thus, the larger the eigenvalue spread of the data covariance matrix, R, the longer it takes for the LMS algorithm to converge. This is a major disadvantage of the LMS algorithm, since the convergence can be very slow if the ratio M/m (known as condition number of the covariance matrix) is too large and generally speaking this is the case in most practical situations. Convergence of the LMS algorithm can be accelerated by the orthogonalizing technique (ie. making all the eigenvalues of the covariance matrix identical). Various techniques can be utilized to achieve orthogonalization and are explained in the next section.

2.6.4 Self orthonormalizing adaptive algorithms

With reference to the steepest descent algorithm, the mean squared error at iteration K can be expressed in terms of its optimum value and the excess mean squared error (EMSE), [5, 7, 8].

$$J(W_K) = J(W^*) + V_K^t R V_K \tag{2.57}$$

where,

$$V_K = (I - 2aR)V_{K-1} \tag{2.58}$$

using similarity transformation, equations (2.57) and (2.58) can be expressed in their decomposed forms, ie.,

$$J(W_K) = J(W^*) + V'_K D V'_K \tag{2.59}$$

and

$$V'_K = (I - 2aD)V'_{K-1} \tag{2.60}$$

From (2.59) and (2.60), it is easy to show that the EMSE evolves according to [5, 7, 8],

$$\text{EMSE} = \sum_{i=1}^{L} b_i m_i (1 - 2am_i)^{2K} \tag{2.61}$$

where the coefficients b_i of the L exponential modes are determined by the initial weight vector W_1. Obviously, the problem of selecting a value for a to yield fast convergence is affected by the eigenvalue spread m_i. The most successful modification of gradient descent algorithm to obtain fast convergence is to use the inverse covariance matrix R^{-1}, ie.,

$$W_{K+1} = W_K - aR^{-1}\nabla J(W_K) \tag{2.62}$$

which reduces to

$$W_{K+1} = ((1 - 2a)W_K + 2aR^{-1}P) \tag{2.63}$$

in which case (2.69) is modified to

$$\text{EMSE} = (1 - 2a)^{2K} \sum_{i=1}^{L} b_i m_i \tag{2.64}$$

Thus, for orthogonalized algorithm (2.62), convergence in (2.64) occurs in one step with the step size parameter $a = \frac{1}{2}$ [5,7,8,].

The algorithm (2.62) which is a non-stochastic modified gradient descent algorithm can be extended to stochastic modified gradient descent algorithm by replacing the deterministic deflected gradient with its stochastic version, $-2e_K R_{K-1}^{-1} X_K$, which results in a modified LMS algorithm, thus,

$$W_{K+1} = W_K + 2ae_K R_{K-1}^{-1} X_K \tag{2.65}$$

where, R_{K-1}^{-1} is obtained from some stochastic data sub-algorithm (see Appendix 1). The algorithm (2.65) is known as the LMS/Newton algorithm [8], or deflected-stochastic gradient (DSG) algorithm. The DSG algorithm is clearly much more complex than the LMS algorithm due to the requirement of estimating the inverse of the covariance matrix. The detailed convergence properties of algorithm (2.65) are not presented here, however, references [5, 7, 8, 28-33] give a good introduction to its underlying theory. Obviously the only computationally demanding procedure involved in (2.65) is to estimate R_{K-1}^{-1} recursively. It is shown in Appendix 1, that using matrix inversion lemma [5, 7, 8, 81], R_{K-1}^{-1} can be computed by $O(L^2)$ operations.

Using an estimate of the inverse of the covariance matrix is not the only technique to achieve orthogonalization. An alternative method of orthogonalization is to use a whitening filter in order to decorrelate the input data x_K. The most popular technique of decorrelating the input data involves Lattice structures which are explained in the next section.

2.6.5 The lattice gradient algorithms

Lattice configurations form an important class of architectures for signal processsing. They possess regularity of structures comprising of identical stages (sections), which have orthogonal properties and involve bounded coefficients [7, 34-36]. Lattices are thus inherently stable. These properties make them particularly attractive for adaptive processing. Digital lattice filters can be realized to have pole-zero transfer function, all pole transfer function, or all zero transfer function in which the latter is the most common configuration used in adaptive signal processing. Figure 2.4 represents an L-stage all zero digital lattice filter. The fundamental equations describing the lattice filter structure illustrated in Figure 2.4 are,

$$\begin{aligned} f_K^i &= f_K^{i-1} - k^i b_{K-1}^{i-1} \\ b_K^i &= b_{K-1}^{i-1} - k^i f_K^{i-1} \end{aligned} \qquad (2.66)$$

where, f_K^i and b_K^i are referred to as the forward (FD) and backward (BD) residuals at i-th stage, k^i are the lattice coefficients known as the reflection coefficients (RC), $i = 1, 2, \ldots\ldots L$, $K = 1, 2, \ldots\ldots N$, and $f_k^1 = b_K^1 = x_K$, where x_K = input data sequence.

The lattice structure shown in Figure 2.4 was originally proposed by Itakura and Saito [37] for performing speech analysis. The orthogonalization of x_K is done by recursion (2.66). Obviously the lattice filter has a more complex structure and requires more numerical operations to implement a transfer function than does the tap-delay line counterpart. However, the increased complexity is offset by several advantageous properties of the lattice structure, including a stage by stage orthogonalization of the input signal, good numerical properties when finite precision arithmetic is used and its suitability for VLSI implementation. The lattice filter structure naturally evolves from a prediction filter where orthogonality conditions are applied.

The most important advantage of the lattice structure is the fact that for each stage the BD prediction error at the output is orthogonal to both prediction errors at the input. This decouples successive stages, thereby enabling the optimization of each stage of the lattice independently [7, 34-37]. This is in contrast to the tap-delay line structure where the coefficients are adjusted jointly, leading to poor convergence properties (related to the statistics of the input signal) [5, 7, 8,].

There are a number of different algorithms available for calculating the RC of the lattice filter [7, 34-38]. The method of choice is generally the Burg's algorithm (also known as the harmonic mean algorithm) [38], which gives the optimum value of the reflection coefficients as being the ratio of the expectations of the negative crosspower and the mean output power of the FD and delayed BD prediction errors,

$$k^{i+1} = \frac{-2E[f_K^i * b_{K-1}^i]}{E[(f_K^i)^2] + E[(b_{K-1}^i)^2]} \qquad (2.67)$$

Burg's algorithm is chosen because it has a significant theoretical basis and minimizes a well defined, reasonable error criterion, namely, the sum of the FD prediction error power and BD prediction error power [38].

The algorithm (2.67) can be implemented recursively (coefficients recomputed with each new data sample) by means of either a gradient technique or a recursive least squares technique. This section presents recursive estimation of the RCs using gradient techniques and their recursive computation by means of RLS technique will be presented in a later section.

Consider both FD and BD residual error power at i-th stage,

$$E[(f_K^i)^2] = E[(f_K^{i-1})^2] - 2k^i E[(f_K^{i-1}) * (b_{K-1}^{i-1})]$$
$$+ (k^i)^2 E[(b_{K-1}^{i-1})^2]$$

$$E[(b_K^i)^2] = E[(b_{K-1}^{i-1})^2] - 2k^i E[(f_{K-1}^{i-1}) * (b_{K-1}^{i-1})]$$
$$+ (k^i)^2 E[(f_K^{i-1})^2] \qquad (2.68)$$

For a stationary process $E[(b_{K-1}^{i-1})] = E[(f_K^{i-1})]$.
Thus, FD residual power = BD residual power.
Now, consider the cost function for stage, i, with respect to both FD and BD residual energies,

$$J(k^i) = \left[E[(b_{K-1}^{i-1})^2] + E[(f_K^{i-1})^2] - 4k^i \left[E[b_{K-1}^{i-1}) * (f_K^{i-1})\right]\right]$$
$$+ (k^i)^2 \left[E[(b_{K-1}^{i-1})^2] + E[(f_K^{i-1})^2]\right] \qquad (2.69)$$

Taking the gradient of (2.69) with respect to k^i and equating it to zero yields,

$$k^i = \frac{-2E[(b_{K-1}^{i-1}) * (f_K^{i-1})]}{E[(b_{K-1}^{i-1})^2] + E[(f_K^{i-1})^2]} \qquad (2.70)$$

which is the Burg's method for computing the optimum value of k^i [38].

A recursive gradient descent algorithm can be used to calculate the RCs for the lattice when the statistics of the prediction errors are precisely known. This algorithm is based on the sum of the FD and BD prediction error gradients and is given by,

$$k_{K+1}^i = k_K^i - a^i \nabla J(k^i) \qquad (2.71)$$

It is easy to show that the step size parameter, a^i, is inversely proportional to the sum of the FD and BD prediction error power, thus (2.71) can be written as,

$$k_{K+1}^i = k_K^i - \frac{\alpha}{p^i} \nabla J(k^i) \qquad (2.72)$$

where, α = small positive constant and p^i = sum of the FD and BD prediction error power. Since the precise statistics of the gradient $\nabla J(k^i)$ and the prediction error power p^i are not known apriori, their estimates are used. Utilizing the instantaneous estimates of $\nabla J(k^i)$ yields the stochastic lattice gradient algorithm [7, 39-43], (see Appendix 2 for a detailed derivation),

$$k^i_{K+1} = k^i_K - \frac{\alpha}{\tilde{p}^i}\left[f^{i+1}_K * b^i_{K-1} + f^i_K * b^{i+1}_K \right] \tag{2.73}$$

where, $\tilde{p}^i$ = an estimate of the joint prediction error power. Various estimators can be used to calculate $\tilde{p}^i$ recursively of which the most common one is [7, 39-43],

$$\tilde{p}^i_K = \beta \tilde{p}^i_{K-1} + (1-\beta)\left[(f^i_K)^2 + (b^i_{K-1})^2\right] \tag{2.74}$$

where, β = positive constant (0.95 to 0.99) which controls the extent of smoothing [7, 39-43].

The algorithm (2.73 - 2.74) was presented by Griffith and utilizes two adaptation parameters, α and β . The adaptation parameter β can be set to the range $0 \leqslant \beta \leqslant 1-\alpha$, where α is selected in the range 0.95 to 0.99. However, $\beta = 1-\alpha$ represents the theoretical best case (closest to the Burg's formula) [7]. References [7, 39-43] provide a good introduction to the underlying theory of various lattice gradient descent algorithms for computing the RCs.

So far the techniques for computing the lattice coefficients have been discussed and it remains to outline the conventional algorithms for adaptive joint processing using a lattice structure which is shown in Figure 2.5.

Figure 2.5 is an equivalent representation of Figure 2.2 using a lattice structure mainly for orthogonality purposes. The lattice filter processes the input signal x_K using RCs and provides a set of BD orthogonal signals to the combiner (NB., initially of course, the BD signals will not be orthogonal [39-43]. Orthogonality of BD signals is only approached in steady state. This issue will be discussed in more detail in chapter 3). Generally, there are two gradient methods which may be used to update the combiner tap-weights. The first technique combines all the lattice outputs in a single global combining structure as shown in Figure 2.5. This results in the same error signal being used in each tap-weight update loop [42]. An alternative method [42] is to derive individual error signals for each of the update loops. This method is more suitable for hardware multiplexing than the global type, but suffers from an increased level of algorithm noise [42]. Ignoring the effects of algorithm self-noise, either technique should produce identical results. However, the results of some recent experiments carried out in [42] indicate that the self noise of method two may be considerably greater than that of method one. Thus, the gradient algorithm which implements the first technique is outlined below.

$$W_{K+1}^{i} = W_{K}^{i} + a_{K}^{i} * b_{K}^{i} * e_{K}^{i+1} \tag{2.75}$$

where,

$$e_{K}^{i+1} = d_{K} - \sum_{i=1}^{L+1} b_{K}^{i} * W_{K}^{i} \tag{2.76}$$

and the step size a_{K}^{i} is computed by,

$$\gamma_{K+1}^{i} = (1-\alpha)\,\gamma_{K}^{i} + (b_{K}^{i})^{2} \tag{2.77}$$

$$a^{i}_{K+1} = \frac{1}{\gamma^{i}_{K+1}} \qquad (2.78)$$

Note that in algorithms (2.75-2.78), different adaptive step sizes are used in order to keep the overall convergence rate of the adaptive lattice algorithm insensitive to the eigenvalue disparity [41-42]. For a detailed analysis of the lattice gradient algorithms see references [7, 39-43].

It has been shown by many [40-43] that lattice gradient algorithms offer a faster convergence than the gradient TDL algorithms. Although lattice gradient algorithms are more complex than their TDL counterparts, they have been widely used in channel equalization linear predictive coding (LPC) of speech waveforms for bandwidth compression, high resolution adaptive spectral analysis, etc.

2.6.6 The recursive least squares algorithms

The RLS algorithms are practical implementations of Newton's method. The Newton's method of searching the minimum of a quadratic function involves the first derivative and the inverse of the second derivative of the cost function and can be expressed as [5, 7, 13-15],

$$W_{K+1} = W_K - R^{-1} * \nabla J(W_K) \qquad (2.79)$$

For a quadratic function, Newton's method proceeds to the optimum solution, W^*, in one step [5, 7, 13-15].

In most practical situations, the exact value of the first derivative, $\nabla J(W_K)$, and the inverse of the second derivative, R, (data covariance matrix) of the cost function are not known apriori and their estimates must be used. The best estimate of the data covariance matrix, R, is its maximum likelihood estimate (MLE) [5, 7] which can be expressed by,

$$R_K = \frac{1}{K} \sum_{i=1}^{K} X_i X_i^t \qquad (2.80)$$

Using the estimator (2.80) and employing matrix inversion lemma [5, 7, 44, 81], (see Appendix 1), yields the RLS algorithm,

$$W_{K+1} = W_K + R_K^{-1} e_K X_K \qquad (2.81)$$

The RLS algorithm (2.81) converges to an optimal value of the weight vector, W^*, using any initial estimate W_1. However, the recommended value of the initial estimate of the weight vector is $W_1 = 0$ [5, 7]. The initial value of R_K^{-1} may be obtained simply by letting $R_1^{-1} = \delta I$, where, δ is a large positive number (eg. 100) [5, 7]. For a detailed derivation and explanation refer to Appendix 1. The estimator (2.80) is the best unbiased estimate of R under stationary conditions. In adaptive situations where x_K is non stationary, the estimator (2.80) would not be a good estimate of the data covariance matrix. Because of its infinite memory, this estimate would become insensitive to the data non-stationarities for large values of K. This problem can be overcome by introducing a factor, known as a forgetting factor, which provides a finite memory in the estimate of R. The modified estimator is [5, 7, 8],

$$R_K = \sum_{i=1}^{K} \alpha^{K-i} X_i X_i^t \tag{2.82}$$

Using the estimator (2.82) and matrix inversion lemma yields the exponentially weighted RLS algorithm (see Appendix 1).

Referring to algorithm (2.81), the output variance and the misadjustment are shown to be [7] (assuming stationary data, uncorrelated X_K and W_K, and $\alpha = 1$),

$$E(J(W_K)) = J(W^*) + J(W^*)\frac{L}{K}$$

$$\text{Misadjustment} = \frac{L}{K} \tag{2.83}$$

In the finite memory case, the misadjustment can be closely approximated by,

$$\text{Misadjustment} \doteq \left(\frac{L}{2\ln\alpha}\right) \tag{2.84}$$

The RLS algorithm suffers from excess complexity. Also, the RLS algorithm typically requires 32 bit floating point arithmetic in order to remain stable [7]. Some benefits in numerical stability may be obtained by the use of RLS lattice forms which will be discussed in the following section. On the positive side RLS algorithms do have fast, consistent convergence characteristics with convergence guaranteed within 2L input samples for a noise free signal. Performance, however, does degrade in the presence of interfering noise. In recent years there has been a renewed interest in the RLS algorithms, due to availability of its fast (computationally efficient) versions [45, 46, 47] (not applicable to spatial arrays).

2.6.7 The recursive least squares lattice algorithms

The recursive least squares lattice algorithms (LSL) have attracted much attention recently because of their excellent convergence behaviour and fast parameter tracking capability. The LSL algorithms allow the exact solution to the least squares problem to be updated for every newly observed data sample. These adaptive estimation techniques use the properties of the lattice structure to efficiently implement the adaptation. The LSL algorithms look similar to the lattice gradient algorithms except that optimal weighting factors are calculated. To understand the underlying structure of the LSL algorithms requires a general knowledge of linear prediction which is explained below.

The linear prediction model assumes that a data sample x_K at time K, can be approximated as y_K, a weighted sum of previous data samples. Thus, for a L-th order linear prediction with coefficients $(w_1, \ldots\ldots\, w_L)$,

$$y_K = -w_1 x_{K-1} - \ldots\ldots\, -w_L x_{K-L} \tag{2.85}$$

The coefficients are to be chosen so as to minimize the mean-square error between x_K and the estimate, y_K. The L-th order covariance matrix of the process x_K, is expressed as,

$$R = E[X_{K-i}X^t_{K-i}] \tag{2.86}$$

Minimizing the square of the prediction error with respect to the predictor coefficients, w_i, requires that the predictor coefficients satisfy,

$$RW^* = P \tag{2.87}$$

where, $R = E(X_K X^t_K)$, $P = E(x_K X_K)$, $X_K = (x_{K-1}, \ldots\ldots x_{K-L})$,

The equation (2.87) is referred to as Yule-Walker equation or normal equation. To solve (2.87) requires the inversion of the L by L covariance matrix R, which involves in the order of (L^3) computations (multiplications). However, assuming stationary data, then the covariance matrix is a Toeplitz Matrix. Using the Levinson algorithm [48], the normal equation (2.87) in Toeplitz form can be solved in the order of (L^2) computations. The Levinson algorithm is an order recursive technique that uses the solution for an L-th order predictor to generate the solution for (L+1)-th order predictor [7, 34, 48].

A detailed derivation of the Levinson algorithm is outlined in references [7, 34, 48]. In order to develop a complete LSL algorithm using Levinson recursion, it is necessary to define a suitable estimator for the various statistical quantities which are involved along with detailed time evolution of these estimates within the overall lattice structure. This is beyond the scope of this section and for a comprehensive derivation and explanation, the works of Morf and Lee [49], Satorious and Pack [50], Friedlander [34] and Ljung et al [51] should be consulted.

In addition to LSL algorithms developed as an extension of the Levinson recursion, other LSL algorithms have been developed that do not employ the Levinson algorithm and are presented in references [8, 52, 53].

To gain a general idea, this section reviews the concept of conventional LSL algorithms which arise from the basic work of Morf and Lee [49]. These algorithms are referred to as mixed time and order recursive LSL algorithms (for reasons that are clarified below). In the development of LSL algorithms, two aspects of the solution of the normal equation (2.87) are important. The first aspect is the efficient inversion of the covariance matrix that gives rise to the order update recursions. Secondly, the time update structure allows exact least-squares solutions to be computed in a recursive manner for each new data sample. This enables the LSL algorithm to achieve extremely fast convergence and excellent tracking capabilities.

Referring to Figure 2.6, the orthogonal signals are generated throughout the recursions,

$$b_K^{i+1} = b_{K-1}^{i} + k_{K-1}^{i}(f) * f_K^{i}$$

$$f_K^{i+1} = f_K^{i} + k_{K-1}^{i}(b) * b_{K-1}^{i} \tag{2.88}$$

where, $f_K^1 = b_K^1 = x_K$, and $k_{K-1}^i(f)$, $k_{K-1}^i(b)$ are called the forward and backward reflection coefficients, i = 1, 2, ... L. To meet the least square criterion, the RCs must be adjusted as follows [7, 34, 49, 50],

$$k_K^i(f) = \frac{-c_K^{i-1}}{p_{K-1}^{i-1}(b)}$$

$$k_K^i(b) = \frac{-c_K^{i-1}}{p_K^{i-1}(f)} \tag{2.89}$$

where,

$$p_{K-1}^{i-1}(b) = E[(b_{K-1}^{i-1})^t(b_{K-1}^{i-1})] \tag{2.90}$$

$$p_K^{i-1}(f) = E[(f_K^{i-1})^t(f_K^{i-1})] \tag{2.91}$$

$$c_K^{i-1} = E[(f_K^{i-1})^t(b_{K-1}^{i-1})] \tag{2.92}$$

are termed the backward, forward and cross residual energy and play a central role in the derivation of LSL algorithms. The order recursion for $p_K^{i-1}(b)$, $p_K^{i-1}(f)$ are obtained as follows,

$$p_K^i(f) = p_K^{i-1}(f) + k_K^i(f) * c_K^{i-1} \tag{2.93}$$

$$p_K^i(b) = p_{K-1}^{i-1} + k_K^i(b) * c_K^{i-1} \tag{2.94}$$

The order recursion for the interproduct c_K^{i-1} can not be established since it does not exist. Therefore, a time recursion is used in LSL algorithms as follows,

$$c_K^i = c_{K-1}^i + \delta c_K^i \tag{2.95}$$

The quantity δc_K^i can be interpreted as a time differential of the cross residual energy c_K^i expressing the innovation caused by the current sample of input data. The time recursion (2.95) is the underlying formula in all conventional LSL algorithms [7, 49].

δc_K^i is computed from the residual signals of the lattice form inverse filter [7, 49] as follows,

$$\delta c_K^i = \frac{f_K^i * b_{K-1}^i}{\gamma_K^i} \tag{2.96}$$

$$\gamma_K^i = \gamma_K^{i-1} - \frac{(b_{K-1}^{i-1})^2}{p_{K-1}^{i-1}(b)} \tag{2.97}$$

Finally, the residual signals f_K^i, b_K^i are computed by the recursion (2.88).
The factor γ_K^i obtained by recursion (2.97) is termed the likelihood variable and is limited to the range $0 \leqslant \gamma_K^i \leqslant 1$ [7, 49].

The recursions (2.88-2.97) outlined above express the concept of conventional mixed time and order recursion LSL algorithms and require in the order of (L^2) operations per time sample.

Exploiting the orthogonalizing nature of the lattice structure, an exact LSL algorithm can be established which requires only in the order of (L) operations to update per time sample [54-57].

2.7 Linearly constrained minimization problem and conventional adaptive algorithms

In some practical problems, not all possible values of the variables are acceptable and it is often necessary or desirable to impose constraints. A frequent form of constrained minimization problem is the linear equality constraints which can be stated as,

$$\text{minimize } J(W) \qquad \text{subject to } C^t W = f \tag{2.98}$$

where, W = L--parameter vector, the i-th row of m by L matrix C contains the coefficients of the i-th linear constraint, and f is an m-vector [13, 14, 15].

The function J(W) is twice continuously differentiable. Generally, it is assumed for simplicity that the rows of C are linearly independent. A necessary condition for W^* to be a minimum of linear equality constrained problem is

$$\nabla J(W^*) = C^t \gamma^* \tag{2.99}$$

where, γ^* is the m-vector of Lagrange multipliers [13, 14, 15].

Let Z denote a matrix whose columns form a basis for the set of vectors orthogonal to the rows of C. A condition equivalent to (2.99) is [13-15],

$$Z^t * \nabla J(W^*) = 0 \tag{2.100}$$

Another necessary condition for W^* to be a minimum of the linear equality constrained problem is that the matrix $Z^tR(W^*)Z$ should be positive semi definite [13-15], where the L by L matrix $R(W^*)$ is the second derivative of $J(W^*)$. The vector $Z^t\nabla J(W)$ is called the projected gradient of J(W) and the matrix $Z^tR(W)Z$ is called the projected Hessian matrix. For a detailed explanation and derivation of the optimality condition for linear equality constrained problems see references [13-15].

2.7.1 Linearly constrained LMS algorithm

Consider the augmented cost function,

$$J(W,\gamma) = \tfrac{1}{2}(W^tRW - 2P^tW + c) + \gamma^t(C^tW - f) \tag{2.101}$$

where, γ^t = vector of the Lagrange multipliers. The derivative of $J(W, \gamma)$ with respect to W, is given by,

$$\nabla J(W,\gamma) = RW - P + C\gamma \tag{2.102}$$

Equating (2.102) to zero yields,

$$W^* = R^{-1}P - R^{-1}C\gamma \tag{2.103}$$

Using the constraint equation $C^tW = f$ in equation (2.103) and solving for γ results in,

$$\gamma = (C^tR^{-1}C)^{-1}f - (C^tR^{-1}C)^{-1}C^tR^{-1}P \qquad (2.104)$$

or equivalently,

$$\gamma = (f - C^tR^{-1}C)(C^tR^{-1}C)^{-1} \qquad (2.105)$$

Substitute equation (2.105) into (2.103) to obtain the optimal solution for W subject to the linear constraints,

$$W^* = QR^{-1}P + F \qquad (2.106)$$

where,

$$Q = [I - R^{-1}C(C^tR^{-1}C)^{-1}C^t] \qquad (2.107)$$

is a projection operator to the constraint plane and

$$F = R^{-1}C(C^tR^{-1}C)f \qquad (2.108)$$

is the orthogonal vector to the constraint plane. In order to establish a linearly constrained LMS algorithm, replace all quantities in equation (2.102) by their instantaneous values at time K, and utilise the steepest descent approach to update the weight vector, ie.,

$$W_{K+1} = W_K + a(e_K X_K - C\gamma_K) \tag{2.109}$$

where, a is the gradient step size and γ_K is an estimate of the vector of the Lagrange multipliers at time K. Use the constraint equation $C^t W_{K+1} = f$ in equation (2.109) and solve for γ_K. Substitute the solution for γ_K back into equation (2.109) to obtain the linearly constrained LMS algorithm,

$$W_{K+1} = P[W_K - ae_K X_K] + C(C^t C)^{-1} f \tag{2.110}$$

where,

$$P = (I - C(C^t C)^{-1} C^t \tag{2.111}$$

is a projection matrix and operates on the instantaneous gradient vector $(W_K - ae_K X_K)$ in order to convert it to a form orthogonal to the constraint matrix C, and the term $C(C^t C)^{-1} f$ translates this orthogonally constrained vector so as to satisfy the desired constraint [58-59].

The constrained LMS algorithm is widely applied to the practical problem of sensor array processing for signal detection in the presence of spatial interference [58-59]. Generally, constrained adaptation is employed for sensor arrays when a desired signal is not available. The augmented cost function for this case can be expressed as,

$$J(W, \gamma) = \tfrac{1}{2} W^t R W + \gamma^t (C^t W - f) \tag{2.112}$$

The optimal solution (2.106) for this case is reduced to

$$W^* = F \qquad (2.113)$$

where, F is given by equation (2.108).

Utilizing the same approach as for the case with the desired signal available, the constrained LMS algorithm can be expressed as

$$W_{K+1} = P[W_K - ay_K X_K] + C(C^t C)^{-1} f \qquad (2.114)$$

where, y_K = output sequence. The algorithm (2.114) was devised and analysed in detail by Frost for multichannel adaptive array processing [5, 8, 60], shown in Figure 2.7.

2.7.2 Linearly constrained RLS algorithm

Consider the optimal solution (2.106) for W,

$$W^* = QR^{-1}P + F \qquad (2.115)$$

Express this solution in its stochastic form at iteration K to obtain the RLS solution, ie.,

$$W^C_{K+1} = Q_K R_K^{-1} P_K + F_K \qquad (2.116)$$

where,

$$Q_K = [I - R_K^{-1} C(C^t R_K^{-1} C)^{-1} C^t] \qquad (2.117)$$

and

$$F_K = R_K^{-1}C(C^tR_KC)^{-1}f \tag{2.118}$$

Substitution of the unconstrained solution $W_{K+1} = R_K^{-1}P_K$ into the constrained solution (2.116) yields the linearly constrained RLS algorithm (or known as the projected RLS algorithm),

$$W_{K+1}^C = Q_KW_{K+1} + F_K \tag{2.119}$$

where,

$$W_{K+1} = W_K + R_K^{-1}X_Ke_K \tag{2.120}$$

and R_K^{-1} is computed as shown in Appendix 1.

In general, the constrained RLS algorithm (2.119-2.120) requires storage of the previous deflection matrix (an estimate of the inverse of the covariance matrix) and the previous unconstrained weight vector W_K.

For the case of the constrained adaptation without a desired signal, the constrained RLS algorithm (2.119-2.120) is modified and reduced to

$$W_{K+1}^C = F_K \tag{2.121}$$

where, F_K is given by equation (2.118) [5, 7, 8, 61, 62].

The constrained RLS algorithms (2.119-2.120) and (2.121) are known to offer much faster convergence than the constrained LMS counterparts. However, the drawback is their higher computational requirement. References [5, 7, 8, 58-60, 63] and [5, 7, 8, 61, 62, 63] provide a good introduction to the underlying theory of the constrained LMS algorithm and constrained RLS algorithms respectively.

2.8 Quadratically constrained minimization problem and conventional adaptive algorithms

A frequently arising minimization problem in adaptive signal processing is minimizing a quadratic function subject to quadratic or norm constraints. The problem can be stated as,

$$\begin{aligned}&\text{minimize } J(W)\\&\text{subject to } W^tW = B\end{aligned} \tag{2.122}$$

where, W = L--parameter vector, J(W) is twice continuously differentiable and B is a positive constant usually set to 1. This type of minimization problem involves the determination of the minimum eigenvalue, m, of the covariance matrix R and the corresponding eigenvector. Consider the augmented cost function,

$$J(W,\gamma) = W^tRW - \gamma(W^tW - B) \tag{2.123}$$

where, γ = Lagrange multiplier. Equating the gradient of $J(W,\gamma)$ with respect to W to zero yields,

$$RW = \gamma W \tag{2.124}$$

The equation (2.124) is recognised as the classical eigen equation with the eigenvalue γ and eigenvector W. Note that the quadratic function W^tRW is bounded as

$$m|W|^2 \leqslant W^tRW \leqslant M|W|^2 \tag{2.125}$$

where, m and M are the minimum and maximum eigenvalues of the covariance matrix R respectively and $|W|^2$ is the two-norm, or Eueclidean norm, $|W|^2 = W^tW$. For B = 1, the relation (2.125) becomes [13-15],

$$m \leqslant W^tRW \leqslant M \tag{2.126}$$

The minimum of $J(W, \gamma)$ is the minimum eigenvalue of the covariance matrix R with the corresponding eigenvector defined by

$$RW^* = mW^* \tag{2.127}$$

where, W^* = optimum weight vector (eigenvector).

The quadratically constrained algorithms are frequently applied to array processing problems [5, 58]. Also, they are utilized in gain optimizing algorithms for adaptive arrays [64]. Recently the quadratically constrained algorithms have been used in high resolution adaptive spectral analysis, see Figures 2.8 and 2.9, [65-76].

2.8.1 Quadratically constrained LMS algorithm

Referring to the augmented cost function (2.123) and motivated by stochastic descent techniques, the quadratically constrained LMS algorithm can be expressed as [68-70], see Figure 2.8,

$$\tilde{W}_{K+1} = W_K - a y_K X_K \tag{2.128}$$

$$W_{K+1} = \tilde{W}_{K+1} / |\tilde{W}_{K+1}| \tag{2.129}$$

where, the normalizing step (2.129) implements the quadratic constraints. Alternatively, the algorithms (2.128-2.129) can be compactly expressed as a single update recursion [68-70].

The effectiveness of this adaptive technique was demonstrated in [68-70], where the algorithm was utilized in a high resolution adaptive spectral technique for estimating frequency of sinusoids corrupted by Gaussian white noise [68-70]. For global convergence behaviour of the quadratically constrained LMS algorithm, a useful insight can be found in references [68-70].

The constrained LMS algorithm (2.128-2.129) can be simply modified and utilized in adaptive directional spectral estimation for incoming narrow band signals [65], see Figure 2.9. It is a technique of spatial filtering for high-resolution spectral analysis of array data.

2.8.2 Quadratically constrained RLS algorithm

Because of the quadratic constraints on the weight vector, an exact least squares solution does not exist as clarified below.

The second derivative of the least squares cost function

$$J(\tilde{W}) = \tfrac{1}{2}\sum_{i=1}^{K} y_i^2 \qquad (2.130)$$

can be written as,

$$R = \frac{1}{|W|^2}\sum_{i=1}^{K} ((Z_i - y_i\tilde{W})(Z_i - y_i\tilde{W})^t - y_i^2 I) \qquad (2.131)$$

where,

$$\tilde{W} = \frac{W}{|W|}$$

$$y_i = \frac{W^t X_i}{|W|}$$

$$Z_i = X_i - \tilde{W}y_i$$

I = identity matrix

Due to the presence of the term $y_i^2 I$ which is a full rank matrix, it is not possible to establish a direct recursion for the second derivative. Therefore, an approximation of the second derivative must be used. As it is shown in [70], a suitable approximation is (assuming that $E(Z_i y_i) = 0$),

$$\bar{R} \simeq \frac{1}{|W|^2}\sum_{i=1}^{K} Z_i Z_i^t \qquad (2.132)$$

Using (2.132), an approximate quadratically constrained RLS algorithm can be developed [70].

$$\tilde{W}_{K+1} = a_{K+1}(\tilde{W}_K - \bar{R}^{-1}_{K+1}Z_{K+1}y_{K+1})$$

$$\bar{R}^{-1}_{K+1} = (\sum_{i=1}^{K} Z_i Z_i^t)^{-1}$$

$$Z_{K+1} = X_{K+1} - \tilde{W}_K y_{K+1}$$

$$a_{K+1} = |\tilde{W}_K| \;/\; |\tilde{W}_{K+1}|$$

where a_{K+1} is a scalar constant whose value is chosen such that the updated weight vector has a unit norm. Using the matrix inversion lemma [44], a recursion for $\bar{R}^{-1}_{K+1}$ can be obtained as shown in Appendix 1.

An alternative and more common approach to computing the eigenvector W, corresponding to the minimum eigenvalue m, is the inverse power technique [65, 77]. Consider the eigenvector-eigenvalue equation,

$$RW = \gamma W \tag{2.134}$$

The inverse power technique [65, 77] is an iterative method of solving equation (2.134) and can be expressed as,

$$\tilde{W}_{K+1} = R_K^{-1} W_K$$

$$W_{K+1} = \tilde{W}_{K+1} \;/ |\tilde{W}_{K+1}| \tag{2.135}$$

where, W_{K+1} = an estimate of the eigenvector at step K of the iteration cycle. Once again, the technique requires an estimate of the inverse of data covariance matrix R_K^{-1} which must be computed iteratively as data becomes available.

2.9 Discussion and conclusions

An ideal adaptive algorithm must have the following properties;

I Rapid initial convergence speed.

II Good tracking capability.

III Computationally efficient.

The former two of these properties are often in direct conflict with the latter, since rapidly converging algorithms with a good tracking capability tend to be computationally demanding, while computationally efficient algorithms lack rapid initial convergence speed and suffer from poor tracking capability.

Generally, adaptive algorithms can be divided into two broad categories;

I Computationally efficient but slowly converging algorithms (consequently poor tracking capability).

II Computationally demanding but rapidly converging algorithms (consequently good tracking capability).

Among all computationally efficient algorithms, the LMS algorithm is the most popular due to its simplicity and is used in a wide range of practical problems where the computational efficiency is of primary importance, not the speed of convergence. On the other hand, the LMS algorithm is not a suitable approach to those practical problems with rapid convergence requirements. In this case improved performance may be obtained by using lattice gradient algorithms which can be regarded as variants of the LMS algorithm (the trade-off is "of course" increased complexity due to lattice structure).

Although the lattice gradient algorithms offer higher speed of adaptation than the LMS algorithm, they can be classified as slowly converging algorithms when compared with RLS techniques.

The RLS techniques are attractive because of their rapid convergence properties. Their main disadvantage is their high computational requirement. The RLS algorithms require in the order of L^2 multiplications per adaptation cycle (approximately $2.5L^2 + 4L$). Furthermore, RLS techniques are sensitive to numerical round-off effects.

Complexity of RLS algorithms can be reduced to order L multiplications per adaptation cycle by making use of the shifting property of the input vector, ie. data redundancy technique. However, the data redundancy technique is only applicable to TDL linear combiners but not to spatial arrays.

Numerical stability of RLS algorithms may be improved by implementing them using a lattice structure. As a result of several research efforts [86, 87, 88] and extensive experimentation with least squares lattice (LSL) algorithms, it is now widely accepted that these algorithms possess better numerical properties than their direct structure (TDL) counterparts. Thus, there is a great interest in their application to practical problems. Alternatively, the numerical stability of RLS algorithms may be improved by introducing a step size parameter in order to smooth the deflected gradient, thus ensuring that excessively large steps are avoided. The adverse effect would be reduced convergence speed.

The main objective of this chapter was to outline various conventional adaptive algorithms along with a brief discussion of their properties, advantages and disadvantages. These algorithms are well documented and further insights into their properties can be obtained by referring to the technical literature.

Figure 2.10 represents a decision tree for designing adaptive algorithms in terms of computational efficiency, where CE and CD denote computationally efficient and computationally demanding respectively.
In this chapter conventional adaptive algorithms based on the steepest descent and Newton methods were discussed. However, there exist alternative minimization methods which are also descent methods. These techniques are widely known as accelerated gradient algorithms. Although there is a significant background literature available on these algorithms, there has been very little work reported on their application to adaptive signal processing. This is the main issue emphasised throughout the remainder of this thesis.

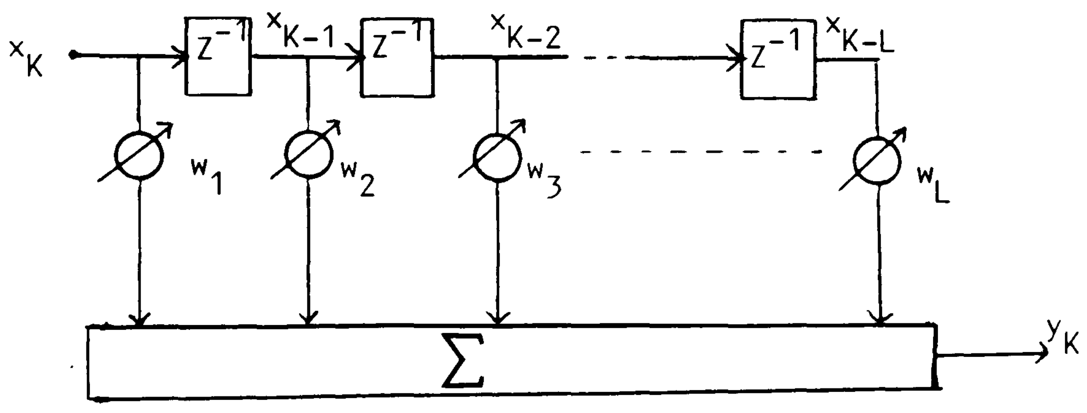


Figure 2.1 : Adaptive linear combiner in the form of single input adaptive transversal filter.

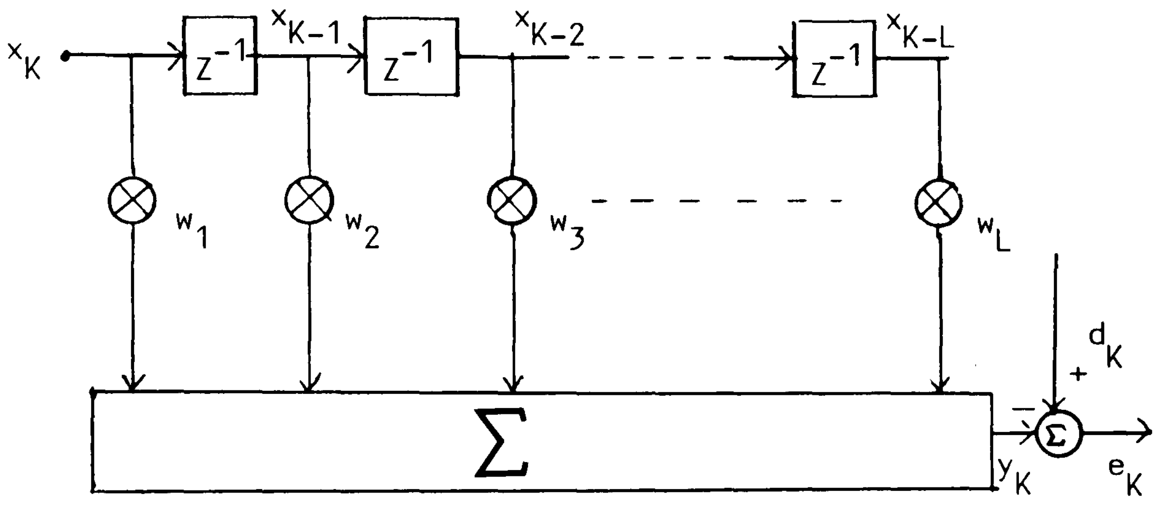


Figure 2.2 : Optimum non-recursive estimation or Wiener estimation.

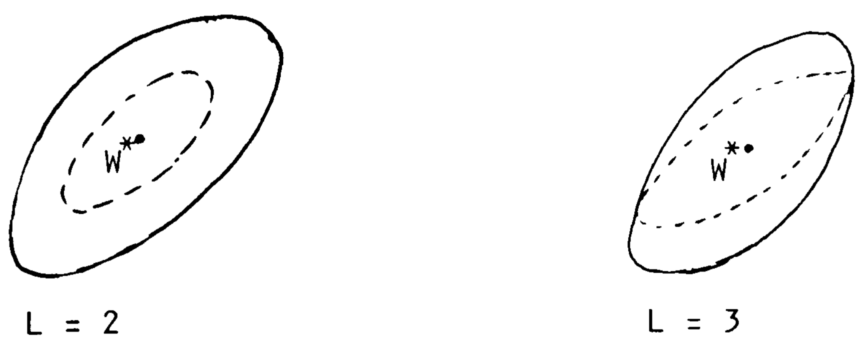


Figure 2.3 : (L-1)-- dimensional ellipsoids having W^* as their common centre.

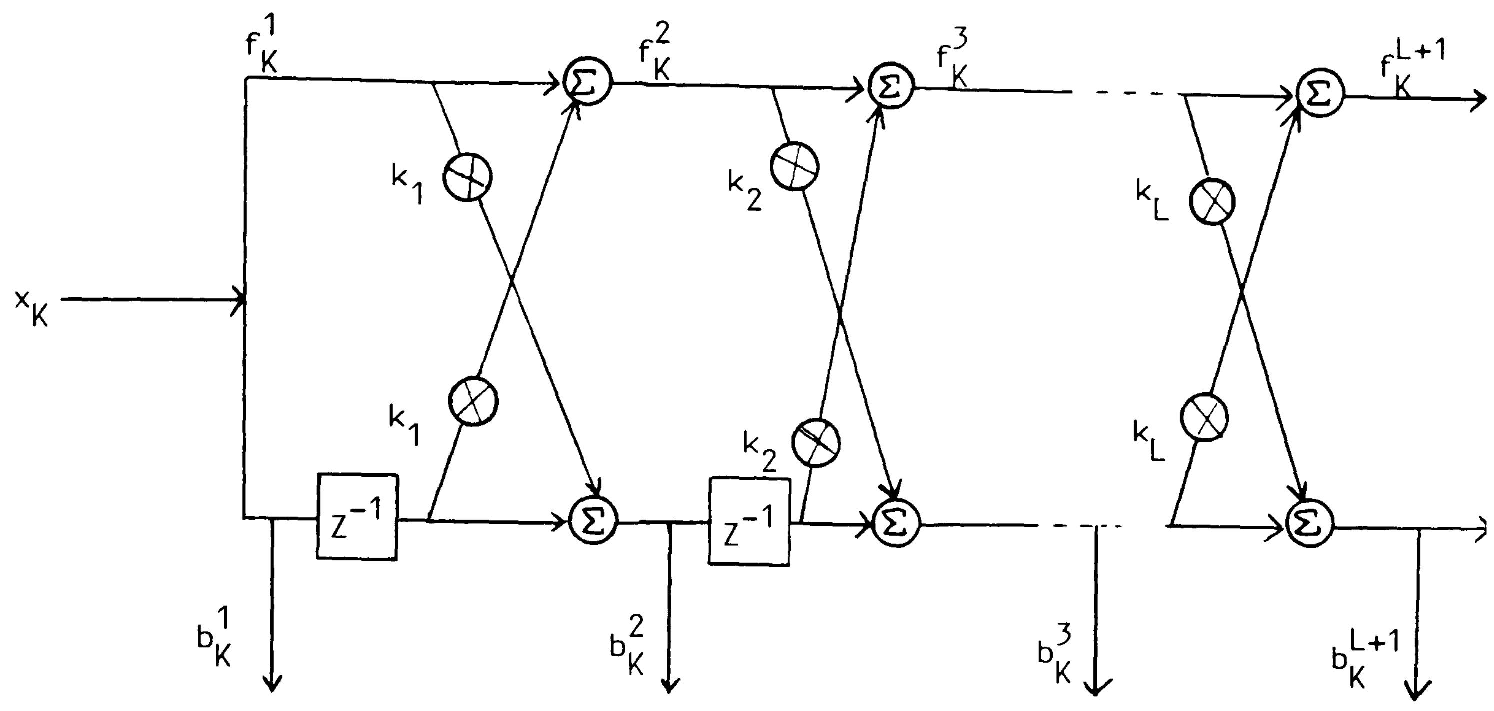


Figure 2.4 : Schematic diagram of an all zero digital Lattice filter.

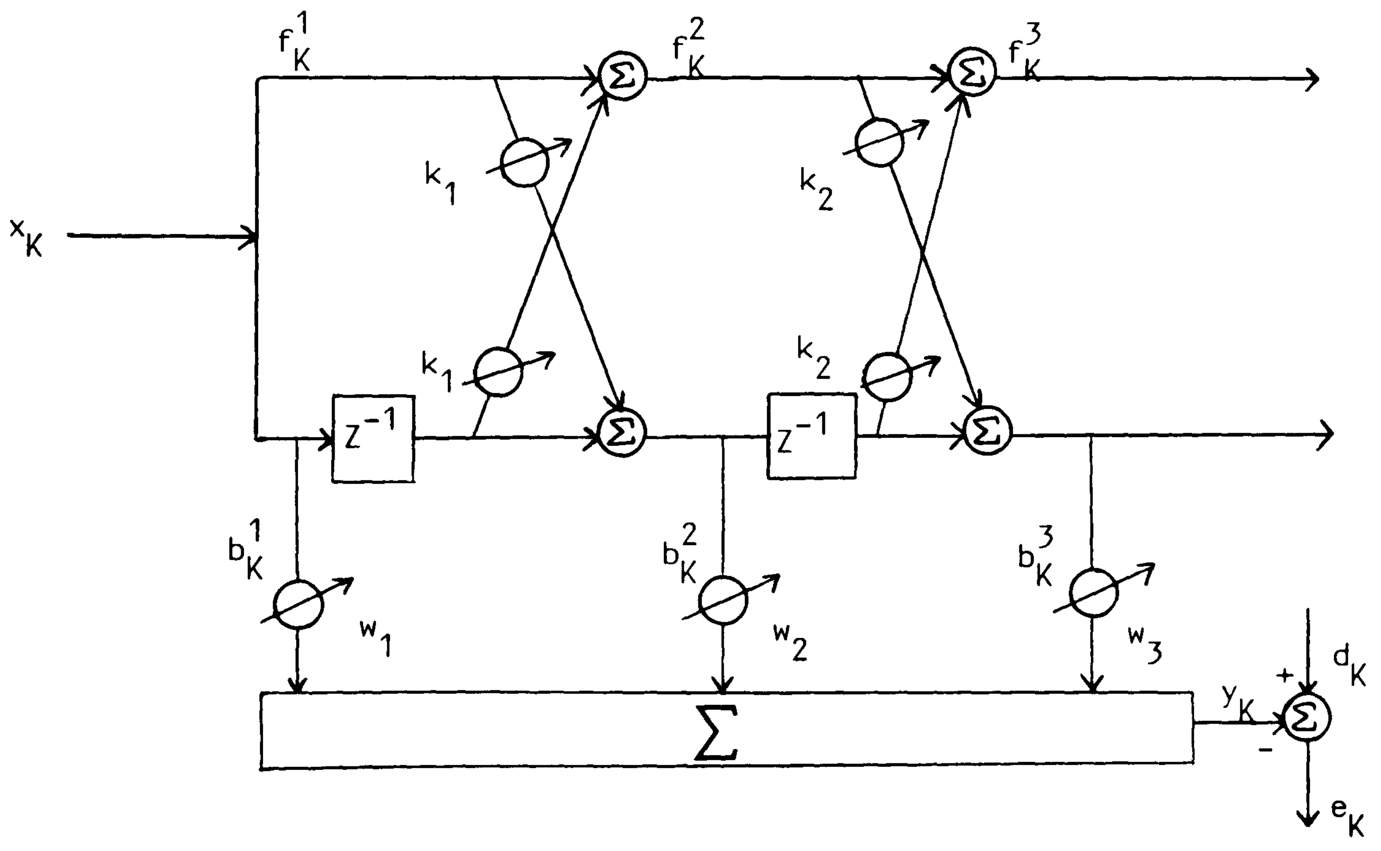


Figure 2.5 : Adaptive joint processing using Lattice structure.

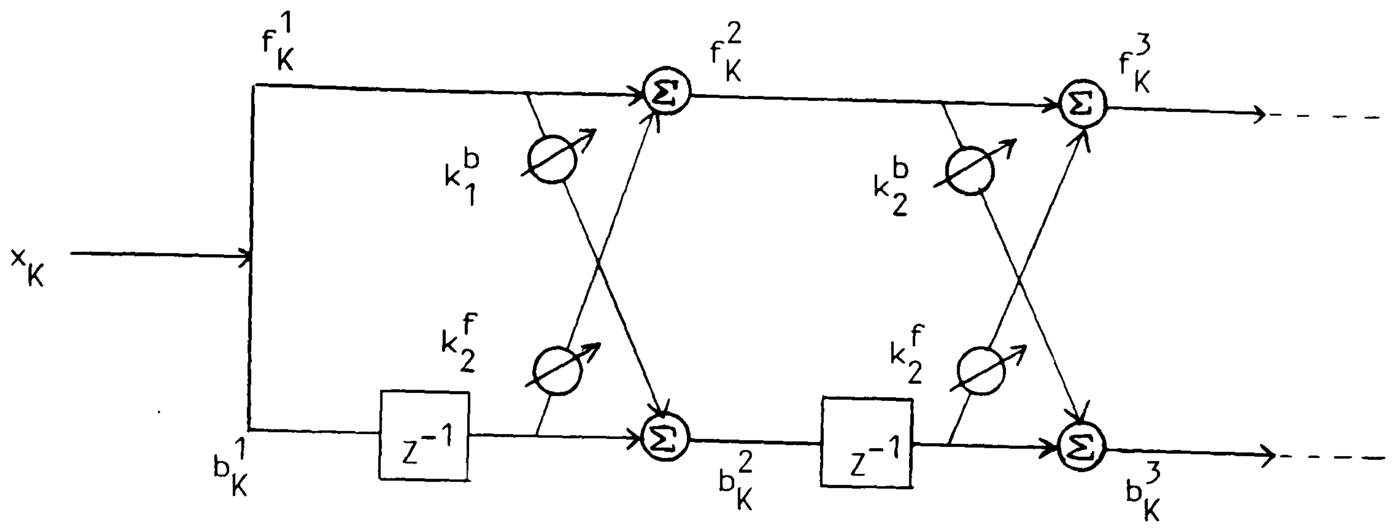


Figure 2.6 : Adaptive Lattice filter, all zero configuration.

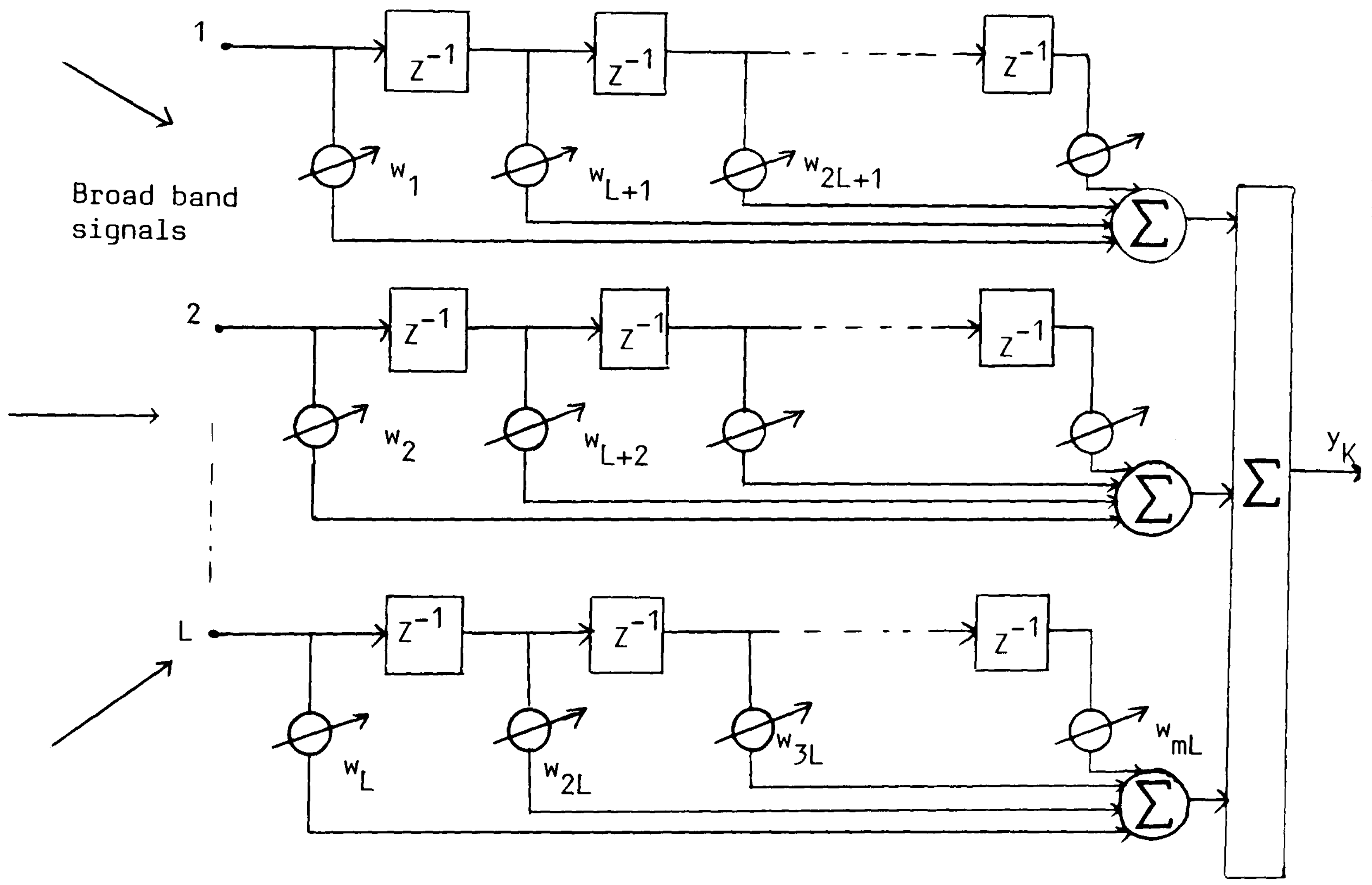


Figure 2.7 : Broad band antenna array processor.
L sensors, m tap per sensor.

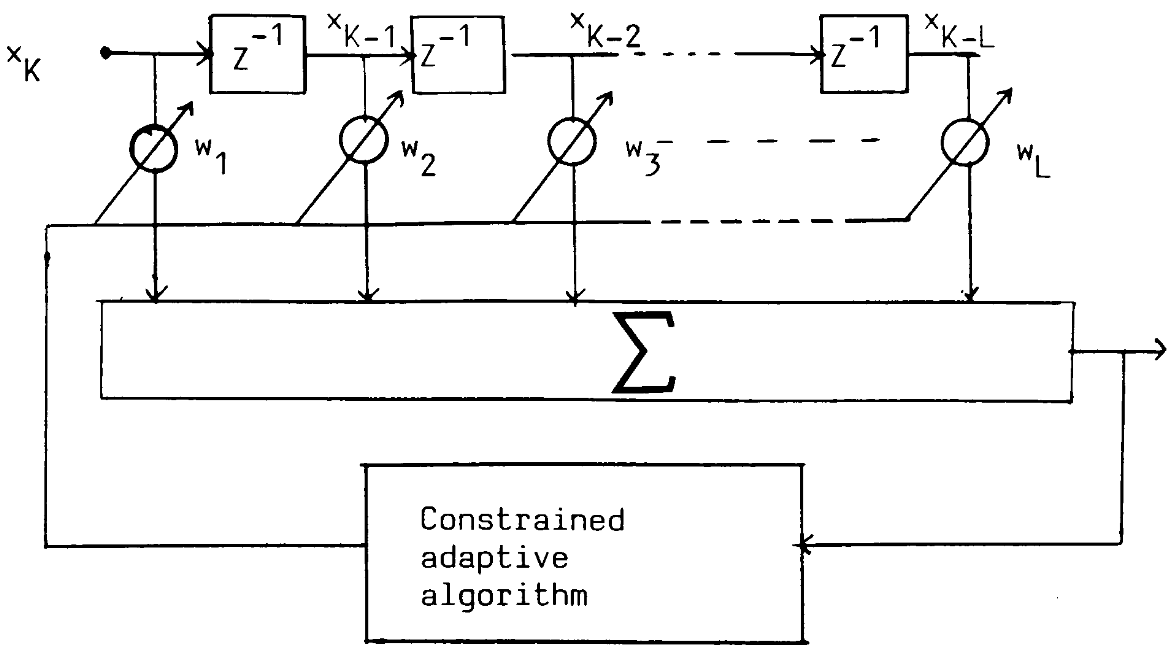


Figure 2.8 : Constrained spectral estimator.

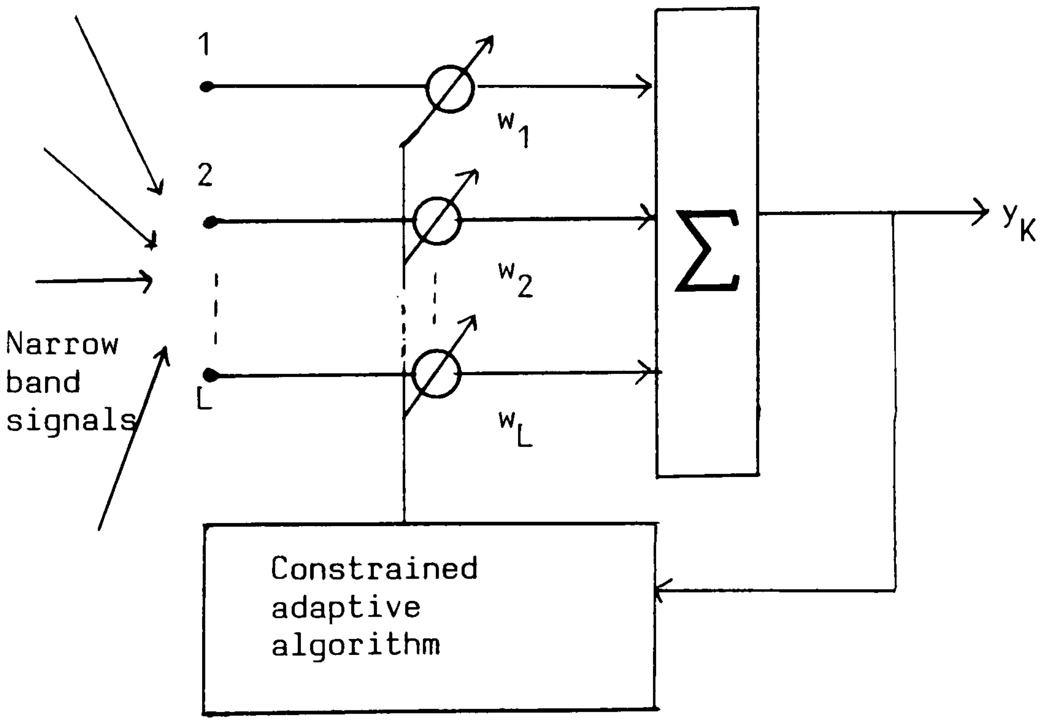
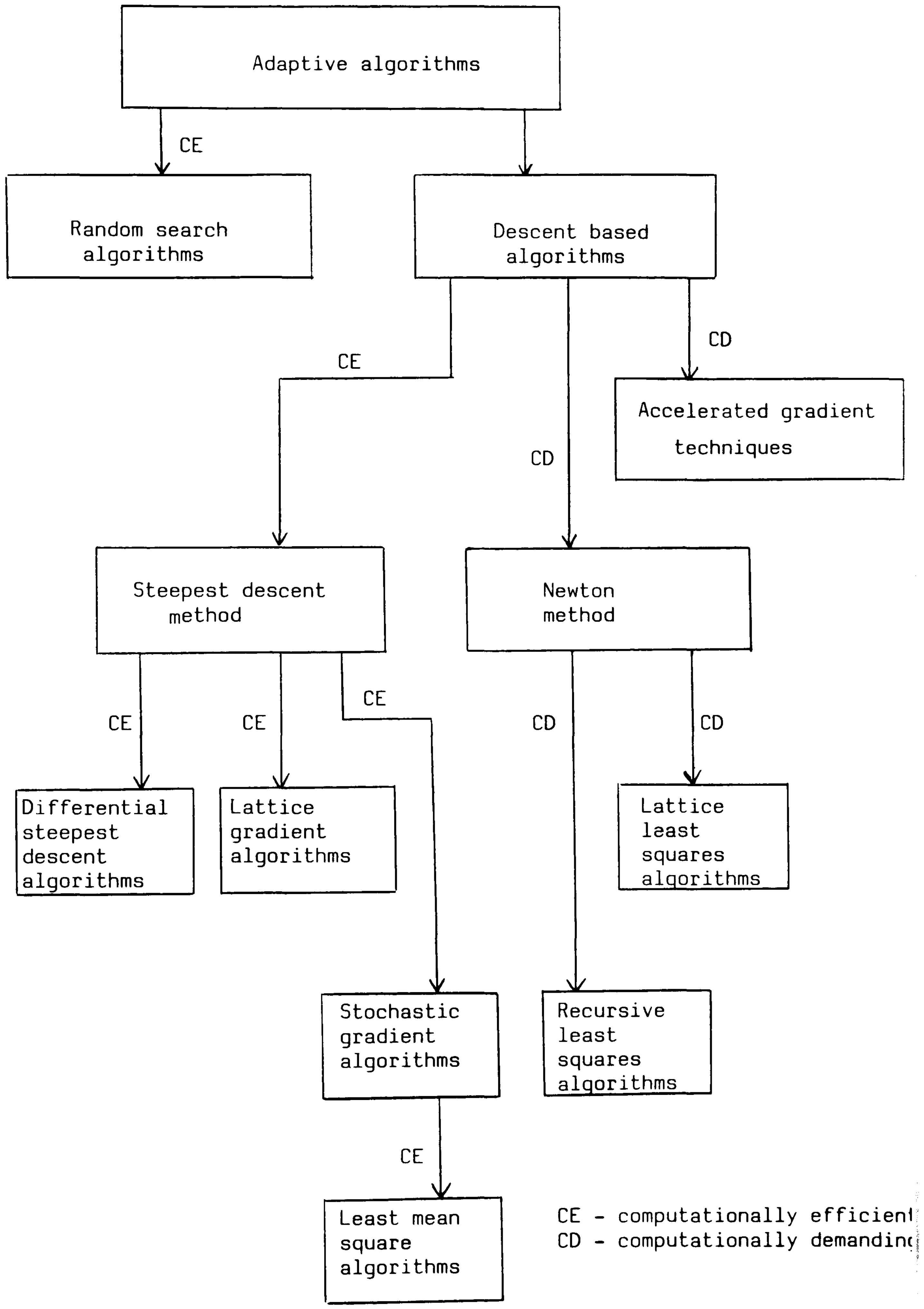


Figure 2.9 : Constrained directional spectral estimator.



Adaptive algorithms
CE
Random search algorithms
Descent based algorithms
CE
CD
CD
Accelerated gradient techniques
Steepest descent method
Newton method
CE
CE
CE
CD
CD
Differential steepest descent algorithms
Lattice gradient algorithms
Lattice least squares algorithms
Stochastic gradient algorithms
Recursive least squares algorithms
CE
Least mean square algorithms
CE - computationally efficient
CD - computationally demanding

Figure 2.10 : Decision tree for designing adaptive algorithms.

CHAPTER 3

ACCELERATED GRADIENT TECHNIQUES AND THEIR APPLICATIONS TO ADAPTIVE SIGNAL PROCESSING

3.1 Introduction

Generally, all gradient methods estimate the minimum of a function using a direction vector q_K which is the descent direction.

This chapter presents various gradient techniques that offer high speed of adaptation and which use second order statistics of the data to compute the direction vector q_K. These techniques will be referred to as the accelerated gradient (AG) techniques.

Throughout the chapter, the application of AG methods to three types of minimization problems (unconstrained, linearly constrained and quadratically constrained) is discussed along with a study of their convergence analysis, stability and computational complexity.

3.2 Un-constrained minimization

Consider the quadratic cost function (multivariate),

$$J(W) = W^t RW - 2P^t W + c \tag{3.1}$$

The process of finding the minimum of J(W) using gradient techniques which involve a search along the vector q_K from the current point W_K can be expressed as,

$$W_{K+1} = W_K + a_K q_K \tag{3.2}$$

where, a_K = step size and q_K = direction vector which is a descent direction.

Fundamental in the development of gradient methods is the generation of the descent direction vector q_K and the step size a_K so that $J(W_{K+1}) < J(W_K)$. In chapter 2 it was shown that when a_K is a small positive constant and $q_K = g_K = -\nabla J(W_K)$, then the convergence to the Wiener solution $W^* = R^{-1}P$ was guaranteed by the linear search (3.2). But the drawback was shown to be the very slow rate of convergence. Various techniques can be employed to increase the rate of convergence   which the most straightforward technique is finding the minimum of a quadratic cost function by the method of exact linear search (ELS). In this case a_K is chosen to minimize $J(W_K + a_K q_K)$ precisely for a given W_K and q_K.

3.2.1 Minimization by exact linear search (ELS)

Referring to the relationship (3.2) and using Taylor's expansion formula, the function $J(W_K + a_K q_K)$ can be expressed as,

$$J(W_{K+1}) = J(W_K + a_K q_K) = J(W_K) - 2a_K g_K^t q_K + a_K^2 q_K^t R q_K \qquad (3.3)$$

where, $g_K = -\nabla J(W_K)$ with respect to W_K. The optimum value of a_K that minimizes (3.3) precisely is obtained by equating the gradient of (3.3) (with respect to a_K) to zero.

$$\nabla J(W_K + a_K q_K) = 0 \qquad (3.4)$$

It follows,

$$a_K = g_K^t q_K \;/\; q_K^t R q_K \tag{3.5}$$

When a_K is obtained by equation (3.5), the procedure is said to be an exact linear search [13-14].

To characterise q_K algebraically consider the Taylor series expansion of $J(W_{K+1})$ in terms of $J(W_K)$ and g_K, as $a_K^2 \simeq 0$.

$$J(W_{K+1}) \simeq J(W_K) - 2a_K g_K^t q_K \tag{3.6}$$

To satisfy $J(W_{K+1}) < J(W_K)$ with $a_K > 0$, it follows immediately that $g_K^t q_K > 0$. Equation (3.6) can be written in the form,

$$J(W_{K+1}) - J(W_K) \simeq - a_K |g_K| |q_K| \cos \theta \tag{3.7}$$

where, θ = the angle between g_K and q_K. Holding a_K, $|g_K|$, $|q_K|$ constant and varying θ, then the right hand side of (3.7) is most negative when $\theta = 0^o$ or 360^o. Thus, the greatest reduction in the function value is obtained in the direction $q_K = g_K = -\nabla J(W_K)$, which is the negative gradient direction or steepest descent direction. Using $q_K = g_K$ in (3.3) yields,

$$a_K = g_K^t g_K \;/\; g_K^t R g_K \tag{3.8}$$

Thus (3.2) can be written as,

$$W_{K+1} = W_K + a_K g_K \tag{3.9}$$

which is the method of steepest descent with exact linear search.

3.2.2 Convergence of the method of steepest descent with exact linear search

With exact linear search it can be shown that (see Appendix 3), [14].

$$J(W_K + a_K g_K) - J(W^*) \leqslant l(J(W_K) - J(W^*)) \tag{3.10}$$

where, $J(W^*)$ = minimum value of the quadratic function and $l = ((M - m) / (M + m))^2$, M and m being the maximum and the minimum eigenvalue of R. Rearranging (3.10) we have,

$$J(W_K + a_K g_K) \leqslant l\ J(W_K) + (1 - l)\ J(W^*) \tag{3.11}$$

It is obvious from,

$$l = ((M - m) / (M + m))^2 \tag{3.12}$$

that, in general, the convergence rate decreases as the condition number M/m increases.

A simple explanation of the slow rate of convergence for a large condition number is given by the fact that in the ELS method the successive search directions g_K are orthogonal. Successive iterations generate a zig-zag path to the minimum. This can result in very small steps being taken in a region away from the minimum if the condition number is large corresponding to substantially ellipsoidal contours of the cost function. This situation is illustrated in Figure 3.1.

Referring to the above convergence discussion, if M/m is close to unity the contours of the cost functions are hyperspherical in the region of the minimum and the method of ELS exhibits quadratic termination. As it is illustrated in Figure 3.1, a large condition number corresponds to ellipsoidal contours thus slow convergence near the minimum. In this case the ELS does not appear to be a practical technique!

However, in most practical adaptive signal processing situations we are dealing with non-stationary data and as a result a variable optimum point. The fundamental requirement in a non-stationary situation is the initial speed of response of the adaptive algorithm to data non-stationarities. In other words, it is the tracking property of the adaptive algorithm which must be taken into consideration, not its final rate of convergence.

Figure 3.1 shows the convergence behaviour of the ELS method for a situation in which M/m is assumed to be large (ellipsoidal contours).

Considering the zig-zag path it is clear that very large steps are taken initially, corresponding to the rapid initial rate of convergence of the algorithm which is the main requirement in most practical situations (ie. rapid tracking of the data non-stationarities).

This interesting property of the method of steepest descent with ELS is utilized to establish adaptive algorithms which will be presented in various sections of this chapter.

Furthermore, the concept of conjugacy and linear independence is used to generate a set of search vectors said to be mutually conjugate vectors which, when used sequentially with ELS, result in quadratic termination [13, 14].

3.2.3 Exact linear search adaptive algorithms

The algorithm outlined in sections (3.2.1) and (3.2.2) can be carried over to adaptive filtering problems by simply choosing a statistical estimator for covariance matrix and cross-correlation vector, ie. a sample covariance matrix R_K and a sample cross-correlation vector P_K. In this case R_K and P_K are updated by one or more new data samples at each iteration.

Assuming that all signals can be modelled as zero mean Gaussian processes, then the maximum likelihood estimates of the covariance matrix and cross-correlation vector can be formed using the sample covariance matrix and sample cross-correlation vector given by [5, 8],

$$R_K = \frac{1}{K} \sum_{i=1}^{K} X_i X_i^t$$
$$P_K = \frac{1}{K} \sum_{i=1}^{K} d_i X_i \qquad (3.13)$$

where, $X_i^t = (x_1, x_2, \ldots\ldots, x_L)^t$ is the vector of tap signals and d_i = the desired signal (or primary signal, refer to Figure 2.2).

Using the estimators given by (3.13) and employing ELS minimization technique results in the following adaptive algorithm.

$$R_K = (R_{K-1}*(K-1) + X_K X_K^t) / K$$

$$P_K = (P_{K-1}*(K-1) + d_K X_K) / K$$

$$g_K = P_K - R_K W_K \tag{3.14}$$

$$a_K = g_K^t g_K / g_K^t R_K g_K$$

$$W_{K+1} = W_K + a_K g_K$$

The algorithm (3.14) is the adaptive version of the method of steepest descent using ELS technique. The convergence is guaranteed for any arbitrary initial estimate W_1. The sample covariance matrix R_K can be assumed to be positive definite yielding the optimal step length a_K in successive iterations K. This statement will be discussed in more detail in the following section.

The algorithm (3.14) has little practical significance, since it is only suitable for stationary situations and its tracking capability is limited. This is obvious as the estimators (3.13) treat all signal samples equivalently. Therefore, as K increases the tracking capability of the algorithm decreases. This problem can be overcome by employing windowing techniques which provide a finite memory in the estimates of the covariance matrix and cross-correlation vector thus regarding the new data samples as being more important than the previous ones.

A common windowing technique is the exponential windowing which results in the following estimators of the covariance matrix and cross-correlation vector,

$$R_K = \sum_{i=1}^{K} \alpha^{K-i} X_i X_i^t$$
$$P_K = \sum_{i=1}^{K} \alpha^{K-i} d_i X_i \qquad (3.15a)$$

where, $0 < \alpha < 1$. Employing the estimators (3.15a) the ELS adaptive algorithm can be outlined as follows,

$$R_K = \alpha R_{K-1} + X_K X_K^t$$

$$P_K = \alpha P_{K-1} + d_K X_K \qquad (3.16)$$

$$g_K = P_K - R_K W_K$$

$$a_K = g_K^t g_K \,/\, g_K^t R_K g_K$$

$$W_{K+1} = W_K + a_K g_K$$

The algorithm (3.16) is an exponentially weighted ELS adaptive algorithm that regards the new data samples more important than the previous ones. Note that estimators (3.15a) yield biased estimates of R and P. This bias can be removed by scaling (3.15a) in accord with,

$$R_K = \frac{1-\alpha}{1-\alpha^K} \sum_{i=1}^{K} \alpha^{K-i} x_i x_i^t$$

$$P_K = \frac{1-\alpha}{1-\alpha^K} \sum_{i=1}^{K} \alpha^{K-i} d_i x_i \qquad (3.15b)$$

which result in the following update recursions,

$$R_K = \frac{(1-\alpha^{K-1})\alpha}{(1-\alpha^K)} R_{K-1} + \frac{(1-\alpha)}{(1-\alpha^K)} x_K x_K^t$$

$$P_K = \frac{(1-\alpha^{K-1})\alpha}{(1-\alpha^K)} P_{K-1} + \frac{(1-\alpha)}{(1-\alpha^K)} d_K x_K \qquad (3.15c)$$

An alternative data weighting scheme is the one that allows for past data samples to be regarded as more or less important than the new data samples [5]. The sample covariance matrix and sample cross-correlation vector that employ this type of data weighting scheme are obtained by,

$$R_K = (1-\beta)R_{K-1} + \beta x_K x_K^t$$

$$P_K = (1-\beta)P_{K-1} + \beta d_K x_K \qquad (3.17)$$

where, $0 < \beta < 1$. For $\beta = \frac{1}{K}$ each sample is equally weighted. Employing the estimators (3.17) the ELS adaptive algorithm can be outlined as follows,

$$R_K = (1-\beta)R_{K-1} + \beta x_K x_K^t$$

$$P_K = (1 - \beta)P_{K-1} + \beta d_K X_K$$

$$g_K = P_K - R_K W_K$$

$$a_K = g_K^t g_K \;/\; g_K^t R_K g_K \qquad (3.18)$$

$$W_{K+1} = W_K + a_K g_K$$

Note that, for K smaller than L, the ELS adaptive algorithms have a growing dimension, which means a singular matrix R_K. To ensure that R_K for K less than L be positive definite, R_1 can be initialized as $R_1 = X_1 X_1^t + \delta I$, where, δ is a positive quantity and I is the identity matrix.

3.2.4 On the optimality of the convergence factor a_K

In section (3.2.1) it was shown algebraically that $a_K = g_K^t g_K \;/\; g_K^t R g_K$ minimizes $J(W_K)$ precisely yielding a convergence rate given by $l = ((M - m) / (M + m))^2$. This is true if and only if R is a positive definite matrix thus ensuring the following well known inequalities [14-15], (see Appendix 3),

$$m|g_K|^2 \leqslant g_K^t R g_K \leqslant M|g_K|^2$$

$$\frac{|g_K|^2}{M} \leqslant g_K^t R^{-1} g_K \leqslant \frac{|g_K|^2}{m} \qquad (3.19)$$

$$\frac{1}{M} \leqslant \frac{g_K^t g_K}{g_K^t R g_K} \leqslant \frac{1}{m}$$

Sample covariance matrices given by (3.13), (3.15) and (3.17) are all positive definite matrices except for K less than L. Therefore, to satisfy the inequalities (3.19), R_K can be initialized as $R_1 = X_1X_1^t + \gamma I$ as was mentioned previously.

Figures 3.2 and 3.3 illustrate the relative behaviour of the convergence factor a_K and the inverse of the maximum and the minimum eigenvalues of the covariance matrix when the ELS technique was used to minimize a quadratic function of the form,

$$J(W) = W^tRW - 2P^tW + c \qquad (3.20)$$

with the known R, P and c. Figure 3.2 shows the behaviour of a_K for R having a small condition number and Figure 3.3 illustrates this behaviour for a large condition number.

Figures 3.4, 3.5, ... 3.9 demonstrate the variation of the step length a_K with respect to the inverse of the maximum and inverse of the minimum eigenvalues of the estimated covariance matrix. In this case the same quadratic function (3.20) had to be minimized but to illustrate the effect of various estimators (3.13, 3.15b, 3.17) on behaviour of a_K , R and P were assumed to be unknown and were estimated adaptively as the data samples became available (an adaptive noise cancelling scheme) . Figures 3.4-3.5, 3.6-3.7, and 3.8-3.9 show the variation of a_K (for small and large condition numbers) with respect to the estimated eigenvalue boundary when estimators 3.13, 3.15b and 3.17 were used respectively.

From these results it is clear that the inequality

$\frac{1}{M_K} \leqslant a_K \leqslant \frac{1}{m_K}$, where, M_K and m_K are the maximum and minimum eigenvalues of the sample covariance matrix at each iteration K, is satisfied irrespective of the condition number or type of estimator. These results are used to extend the convergence behaviour of the deterministic algorithm outlined in sections (3.2.1) and (3.2.2) to convergence behaviour of its adaptive versions given by (3.14), (3.16) and (3.18).

3.2.5 Convergence of adaptive algorithms with ELS

To study the convergence of the ELS adaptive algorithms, it is essential to make the fundamental assumption that estimates of the covariance matrix are positve definite at each iteration. This assumption will enable the derivation of convergence relationships in terms of the maximum and minimum eigenvalues of the estimate of the covariance matrix at each iteration K.

Referring to algorithms (3.14, 3.16, 3.18), the cost function at each iteration can be expressed as a scaled sum of squares of error.

$$J(W_K) = W_K^t R_K W_K - 2P_K^t W_K + c_K \tag{3.21}$$

Since R_K is a positive definite matrix then the quadratic function (3.21) has a unique minimum $J(W_K^*)$. Also using the Taylor expansion formula, the updated cost function $J(W_K + a_K g_K)$ can be expressed as,

$$J(W_{K+1}) = J(W_K) - 2a_K g_K^t g_K + a_K^2 g_K^t R_K g_K \qquad (3.22)$$

The derivative of (3.22) is equal to zero if a_K is given by,

$$a_K = g_K^t g_K \;/\; g_K^t R_K g_K \qquad (3.23)$$

which is the convergence factor for algorithms (3.14, 3.16, 3.18).

Now with reference to convergence analysis of the method of steepest descent with exact linear search outlined in section (3.2.2), we have,

$$J(W_K + a_K g_K) - J(W_K^*) \leqslant l_K(J(W_K) - J(W_K^*)) \qquad (3.24)$$

or

$$J(W_{K+1}) - J(W_K^*) \leqslant l_K(J(W_K) - J(W_K^*)) \qquad (3.25)$$

where,

$$l_K = ((M_K - m_K) \;/\; (M_K + m_K))^2 \qquad (3.26)$$

and M_K, m_K are the maximum and minimum eigenvalues of the estimate of covariance matrix R_K.

The above discussion can be repeated for successive iterations. Therefore, it can be stated that, successive cost functions : $J(W_K)$, $J(W_{K+1})$, ..., etc. have unique minimum points

$J(W_K^*)$, $J(W_{K+1}^*)$, $J(W_{K+2}^*)$, ..., etc and as a result of ELS adopted at each iteration,the following relationships can be established.

$$J(W_{K+1}) - J(W_K^*) \leqslant l_K(J(W_K) - J(W_K^*))$$
$$J(W_{K+2}) - J(W_{K+1}^*) \leqslant l_{K+1}(J(W_{K+1}) - J(W_{K+1}^*)) \qquad (3.27)$$
$$J(W_{K+3}) - J(W_{K+2}^*) \leqslant l_{K+2}(J(W_{K+2}) - J(W_{K+2}^*))$$

.

.

.

etc.

where,

$$l_K = ((M_K - m_K) / (M_K + m_K))^2$$
$$l_{K+1} = ((M_{K+1} - m_{K+1}) / (M_{K+1} + m_{K+1}))^2 \qquad (3.28)$$
$$l_{K+2} = ((M_{K+2} - m_{K+2}) / (M_{K+2} + m_{K+2}))^2$$

.

.

.

etc.

and M_K, M_{K+1}, M_{K+2} ...... etc., m_K, m_{K+1}, m_{K+2} ...... etc. are the maximum and minimum eigenvalues of R_K, R_{K+1}, R_{K+2}, ..., etc.

Rearranging (3.27) results in,

$$J(W_{K+1}) \leqslant l_K J(W_K) + (1 - l_K)J(W_K^*)$$

$$J(W_{K+2}) \leqslant l_{K+1} J(W_{K+1}) + (1 - l_{K+1})J(W_{K+1}^*) \qquad (3.29)$$

$$J(W_{K+3}) \leqslant l_{K+2} J(W_{K+2}) + (1 - l_{K+2})J(W_{K+2}^*)$$

.

.

.

etc.

Solving (3.29) in terms of the initial value of the cost function yields,

$$J(W_{K+1}) \leqslant (\prod_{i=1}^{K} l_i)J(W_1) + \sum_{j=2}^{K+1} (\prod_{h=j}^{K} l_h)(1 - l_{j-1})J(W_{j-1}^*) \qquad (3.30)$$

where, $J(W_1)$ = initial value of the cost function. If $j = K+1$, then $(\prod_{h=j}^{K} l_h) = 1$. Equation (3.30) relates the value of the cost function $J(W_{K+1})$ to its initial value and unique optimum values $J(W_1^*)$, $J(W_2^*)$, ..., $J(W_K^*)$.

3.2.6 Convergence of ELS adaptive algorithms in a stationary environment

Assuming stationary inputs, then the resultant quadratic cost function has a single fixed minimum point. In this case there exists one optimum weight vector W^* that minimizes the cost function.

In stationary situations all data samples can be equally weighted. This is achieved by employing the estimates of covariance matrix and cross-correlation vector given by (3.13) which result in unbiased estimates of R and P as $K \to \infty$.

It is well known and also experienced (computer simulations) by the author that estimators (3.13) result in a good estimate of R and P approximately after twenty to thirty iterations.This can be verified by referring to Figures 3.2, 3.3, 3.4, 3.5 which show the true and estimated inverse of the maximum and the minimum eigenvalues of R and its estimates R_K given by (3.13) respectively.

In the case of stationary data, the variable optimum point of $J(W_K^*)$ in (3.29) and (3.30) can be represented by the unique optimum value of the cost function $J(W^*)$, (neglecting the first few iterations before convergence of R_K and P_K to R and P). Therefore, as $K \to \infty$the first term on the right hand side of (3.30) vanishes since $l_i < 1$ and $\prod_{i=1}^{\infty} l_i \ll 1$. The second term on the right hand side of (3.30) reduces to $(1 - l^{\infty})J(W^*)$ since estimates $R_K \to R$ and thus M_K, $m_K \to M,m$ as $K \to \infty$. Therefore, equation (3.30) reduces to

$$J(W_{K+1}) \simeq J(W^*) \text{ as } K \to \infty. \qquad (3.31)$$

With regard to the above discussion we may state that, assuming stationary data, the ELS adaptive algorithm (3.14) minimizes the quadratic cost function and results in the Wiener solution as $K \to \infty$.

Due to unbiased estimates $R_K = R$ and $P_K = P$ at $K = \infty$, the algorithm (3.14) at $K = \infty$ behaves like the deterministic steepest descent method with ELS and therefore results in zero steady state misadjustment (neglecting the effects of finite word length implementation).

3.2.7 Convergence of ELS adaptive algorithms in a non-stationary environment

In the case of non-stationary signals, the minimum point and orientation of the cost function could be changing over time. Figure 3.10 illustrates a two dimensional time varying cost function for a non-stationary process.

Due to data non-stationarities and as a result a variable optimum point, the data samples can not be equally weighted and alternative data weighing schemes must be employed. Two of the most well known data weighting schemes are those given by (3.15) and (3.17).

For a non-stationary cost function a logical approach would be to define an optimum parameter vector (weight vector) W_K^* that minimizes the cost function at time K [17, 24, 78]. This is the approach taken in section (3.2.5) in order to establish the relationship (3.30).

Equation (3.30) relates the value of the estimate of the cost function to its initial value and its optimal value at each iteration. The tracking capability of the algorithm is determined by the variable factor l_i which in turn is related to the variable condition number of the estimated covariance matrix.

Due to the data non-stationarities, it is meaningless to study the convergence of an adaptive algorithm with regard to its steady state solution. This is because there is no steady state solution as data statistics vary over time. In this case, it is the tracking capability of the algorithm, at any time K, that must be taken into consideration (ie. the response of the algorithm to a sudden statistical change).

The tracking capability of the ELS adaptive algorithm is demonstrated by simulating a non-stationary adaptive noise cancelling scheme which is presented in chapter 5.

3.2.8 On the convergence of the weight vector

It is shown in section 3.2.2 [Appendix 3] that using the method of steepest descent with ELS, the quadratic function $J(W_K)$ converges to its optimal value by a constant rate 1. Since $J(W_K)$ is a quadratic function of W_K, then it implies that W_K converges to its optimum W^* with a rate $1^{\frac{1}{2}}$. The discussion can be extended to ELS adaptive algorithms simply by considering the relationship,

$$J(W_{K+1}) - J(W_K^*) \leqslant l_K(J(W_K) - J(W_K^*)) \qquad (3.32)$$

The equation (3.32) implies that after ELS the cost function is closer to its optimum value at time K by a factor l_K. Since $J(W_K)$ is a quadratic function of W_K, then after each adaptation

W_K would be closer to its optimum value W_K^* by a factor $1_K^{\frac{1}{2}}$. In other words W_K tracks W_K^* by a factor $1_K^{\frac{1}{2}}$. Note that W_K^* is the optimum value of W_K at iteration K.

3.2.9 Stochastic interpretation and convergence in the mean

Consider the equally weighted least squares optimization problem,

$$\text{minimize } \frac{1}{K} \sum_{i=1}^{K} (d_i - W_{K+1} X_i)^2 \tag{3.33}$$

The solution to this problem is given by,

$$W_{K+1} = \left(\frac{1}{K} \sum_{i=1}^{K} X_i X_i^t \right)^{-1} \left(\frac{1}{K} \sum_{i=1}^{K} d_i X_i \right) \tag{3.34}$$

or

$$W_{K+1} = R_K^{-1} P_K \tag{3.35}$$

Minimization of (3.33) by ELS results in one step approximation of the solution (3.35) at each iteration K.

Assuming stationary stochastic data, then all time averages can be replaced in the limit by expected values. It follows that,

$$\lim_{K \to \infty} \frac{1}{K} \sum_{i=1}^{K} X_i X_i^t \triangleq R \tag{3.36}$$

$$\lim_{K \to \infty} \left(\frac{1}{K} \sum_{i=1}^{K} X_i X_i^t \right)^{-1} \triangleq R^{-1} \tag{3.37}$$

$$\lim_{K\to\infty} \frac{1}{K} \sum_{i=1}^{K} d_i X_i \triangleq P \tag{3.38}$$

and therefore,

$$\lim_{K\to\infty} W_K = W^* \tag{3.39}$$

As a result of (3.36 to 3.39), in the limit ELS adaptive algorithm behaves like its non-stochastic version and converges to the Wiener solution.

A useful measure of an adaptive algorithm's performance is to study its behaviour in the mean (ie. ensemble averaging). In this case the output variance (the mean squared error) can be expressed in terms of its minimum value and the excess mean squared error.

$$E(J(W_K)) = J(W^*) + E(V_K^t R_K V_K) \tag{3.40}$$

where, $V_K = W_K - W^*$. Assume that the ensemble averaging is taken over a very large number of individual runs with equal initial weight/vector W_1, so that the expected error vector behaves like its non stochastic version. Therefore, the expectation operation on V_K in (3.40) may be deleted.

$$E(J(W_K)) = J(W^*) + V_K^t R V_K \tag{3.41}$$

Expanding by Taylor series and taking $q_K = g_K = -RV_K$, the optimum value of a_K that maximizes the convergence speed is obtained by equating the derivative (with respect to a_K) of the expanded function to zero.

$$a_K = \frac{V_K^t R^2 V_K}{V_K^t R^3 V_K} \qquad (3.42)$$

Using the bound,

$$m V_K^t R^2 V_K \leqslant V_K^t R^3 V_K \leqslant M V_K^t R^2 V_K$$

in the denominator yields,

$$\frac{1}{M} \leqslant a_K \leqslant \frac{1}{m} \qquad (3.43)$$

where, m and M are the minimum and the maximum eigenvalues of R.

If the adaptive algorithm is convergent (which it is as long as a_K satisfies (3.43)), then in the limit (3.40) reduces to

$$\lim_{K \to \infty} E(J(W_K)) = J(W^*) \qquad (3.44)$$

with a convergence rate given by

$$1 = (M - m)^2 / (M + m)^2 \qquad (3.45)$$

3.2.10 Computational complexity of ELS adaptive algorithms

A complexity measure of any adaptive algorithm is the required number of multiplications (divisions) per iteration. That is the higher the number of required multiplications the more complex the adaptive algorithm.

With regard to ELS adaptive algorithms proposed in section (3.2.3) we can see that a direct realization of these algorithms requires $4L^2 + 5L + 1$ multiplications per cycle of adaptation. Thus, the algorithms appear to be unsuitable for on line adaptive signal processing applications.

However, utilizing the symmetric property of R_K, $X_K X_K^t$ and $g_K^t R_K g_K$ a great reduction in number of multiplications can be achieved. This procedure is discussed and outlined below.

We start with $X_K X_K^t$ which is a symmetric matrix and it is only essential to compute its upper triangular matrix $(X_K X_K^t)^{\blacktriangledown}$ (for ease of presentaion assume L = 4).

$$(X_K X_K^t)^{\blacktriangledown} = \begin{bmatrix} x_1x_1 & x_1x_2 & x_1x_3 & x_1x_4 \\ & x_2x_2 & x_2x_3 & x_2x_4 \\ & & x_3x_3 & x_3x_4 \\ & & & x_4x_4 \end{bmatrix} \qquad (3.46)$$

The required number of multiplications to form $(X_K X_K^t)^{\blacktriangledown}$ is equal to $\sum_{i=0}^{L-1} (L-i)$. Therefore, there is a saving of

$\sum_{i=1}^{L-1}$ (L-i) multiplications. Note that to weight the new data vector X_K (in accord with algorithms 3.14, 3.18) requires an additional L multiplications.

Next consider the symmetric matrix R_{K-1}. Because of its symmetrical structure it is only necessary to store its upper triangular elements. Therefore, we have (assuming L=4),

$$\overset{\nabla}{R}_{K-1} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & r_{14} \\ & r_{22} & r_{23} & r_{24} \\ & & r_{33} & r_{34} \\ & & & r_{44} \end{bmatrix} \tag{3.47}$$

To weight the upper triangular matrix $\overset{\nabla}{R}_{K-1}$ requires $\sum_{i=0}^{L-1}$ (L-i) multiplications which correspond to a saving of $\sum_{i=1}^{L-1}$ (L-i) multiplications if all elements of R_{K-1} were to be weighted.

With regard to $\overset{\nabla}{(X_K X_K^t)}$ and $\overset{\nabla}{R}_{K-1}$, the equation for updating the covariance matrix R_K will take a form as follows,

$$\begin{aligned} \overset{\nabla}{R}_K &= H_{K-1} + A_K \\ R_K &= \overset{\nabla}{R}_K + \overset{\triangle}{R}_K \end{aligned} \tag{3.48}$$

where, H_{K-1} = weighted R_{K-1}^{∇} in accord with algorithms 3.14, 3.16 and 3.18. $A_K = (\frac{1}{K} X_K X_K)^{\nabla}$, or $(X_K X_K^t)^{\nabla}$, or $(\beta X_K X_K^t)^{\nabla}$ in accord with algorithms 3.14, 3.16 and 3.18 respectively. $\triangle$ represents the upper triangular elements in R_K^{∇} (with the exception of the diagonal terms) written in its lower triangular section.

Finally, consider the scalar $g_K^t R_K g_K$ written in its expanded form for L=4.

$$\begin{array}{ccccccc}
g_1 r_{11} g_1 & + & \boxed{g_1 r_{12} g_2 \quad + \quad g_1 r_{13} g_3 \quad + \quad g_1 r_{14} g_4 \;+} \\
\boxed{g_2 r_{21} g_1} & + & g_2 r_{22} g_2 & \boxed{+ \quad g_2 r_{23} g_3 \quad + \quad g_2 r_{24} g_4 \;+} \\
\boxed{g_3 r_{31} g_1} & + & \boxed{g_3 r_{32} g_2} & + & g_3 r_{33} g_3 & + & \boxed{g_3 r_{34} g_4 \;+} \\
\boxed{g_4 r_{41} g_1} & + & \boxed{g_4 r_{42} g_2} & + & \boxed{g_4 r_{43} g_3} & + & g_4 r_{44} g_4
\end{array} \qquad (3.49)$$

where, g_1, g_2, g_3, g_4 are elements of g_K, ie. $g_K^t = (g_1, g_2, g_3, g_4)_K$.

It can be observed in summation (3.49) that several of the product terms occur twice. This is demonstrated by encircling these terms. Therefore, the scalar $g_K^t R_K g_K$ can be formed by a fewer number of multiplications than $L^2 + L$, as follows.

$$g_K^t R_K g_K = \sum_{j=1}^{L-1} 2g_j \sum_{i=j+1}^{L} r_{ji} g_i + \sum_{h=1}^{L} (g_h g_h) r_{hh} \qquad (3.50)$$

The products $g_h g_h$ are formed by operation $g_K^t g_K$ in the numerator of $g_K^t g_K \ / \ g_K R_K g_K$. This means that the second term on the right hand side of equation (3.50) only involves L multiplications.

Utilizing equation (3.50) to calculate $g_K^t R_K g_K$, there will be a great reduction in the required number of multiplications in order to form the convergence factor a_K. For instance, if L=10, then the total number of multiplications to form a_K would be 75 which corresponds to a saving of 121-75 = 46 multiplications.

Table (a) outlines the total number of operations - multiplication (division) - for direct realization of algorithms 3.14, 3.16 and 3.18 and their realization using the techniques explained in this section. It is obvious that there is an increasing reduction in the required number of operations as dimension L increases.

It must be noted that direct realization of ELS adaptive algorithms in low dimensional systems is not a great concern with today's computing power cost. However, for large dimensional systems there is a severe technological limitation due to the large number of required multipliers. In this case, using techniques outlined in this section result in a significant reduction in the required number of operations thus reducing the complexity of the system.

3.2.11 Block implementation of ELS adaptive algorithms

Block implementation of adaptive algorithms involves the calculation of a block or a finite set of filter outputs from a block of input values and adjustment of the weight vector once per block of data. Block implementation of adaptive algorithms allow efficient use of parallel processors and as a result , high speed signal processing.

It must be noted that updating the weight vector for each data sample yields the highest rate of convergence of the adaptive algorithm, if convergence rate is measured in terms of the number of data samples. However, if the convergence rate is measured in terms of algorithm iterations, then speed of convergence increases as the number of data samples in each block increases.

The block data adaptive algorithms are suitable for applications in which the adjustment of the weight vector can be time consuming and costly (eg. Satellite antenna array adaptation using micro-wave power dividers).

Block LMS adaptive algorithms are discussed and analysed extensively in [16, 18, 79, 80].

The application of a block processing scheme to ELS adaptive algorithms yields a great reduction in computational requirement. Furthermore, employing a block adaptation technique facilitates generation of mutually conjugate search vectors which, when used sequentially with ELS method, results in quadratic termination.

Consider the estimated cost function over N data samples.

$$J^N(W_K) = W_K^t R_K^N W_K - 2P_K^{Nt} W_K + c_K^N \tag{3.51}$$

where, R_K^N = estimate of covariance matrix over N data samples, P_K^N = estimate of cross-correlation vector over N data samples and c_K^N = constant.

Taking the gradient of $J^N(W_K)$ with respect to W_K yields,

$$g_K^N = -\nabla J^N(W_K) = P_K^N - R_K^N W_K \tag{3.52}$$

Employing ELS technique, the update equation for W_K can be written as,

$$W_{K+1} = W_K + a_K g_K^N \tag{3.53}$$

where,

$$a_K = \frac{(g_K^N)^t (g_K^N)}{(g_K^N)^t R_K^N (g_K^N)} \tag{3.54}$$

Equations (3.52), (3.53) and (3.54) represent the block implementation of the ELS, (BIELS), adaptive algorithm. Obviously a direct realization of BIELS involves the computation R_K^N, P_K^N and g_K^N over a block of data which is costly. Also, computation of a_K requires a large number of multiplications.

The computational requirement of BIELS can be greatly reduced by assuming stationary data. So that,

$$R_K^N = R_{K+1}^N \qquad (3.55)$$

$$P_K^N = P_{K+1}^N \qquad (3.56)$$

As a consequence of (3.55) and (3.56) the following procedure can be established.

I. For W_K constant over N data samples compute g_K^N by

$$g_K^N = \frac{1}{N} \sum_{i=1}^{N} e_i X_i \qquad (3.57)$$

II. For W_K constant over another N data samples, $(g_K^N)^t R_K^N (g_K^N)$ can be computed by,

$$(g_K^N)^t R_K^N (g_K^N) = \frac{1}{N} \sum_{i=1}^{N} ((g_K^N)^t X_j)(X_j^t (g_K^N)) \qquad (3.58)$$

III. Compute a_K by

$$a_K = \frac{(g_K^N)^t (g_K^N)}{\frac{1}{N} \sum_{j=1}^{N} ((g_K^N)^t X_j)(X_j^t (g_K^N))} \qquad (3.59)$$

Computation of a_K requires 2N data samples, N data samples due to g_K^N and N data samples due to $(g_K^N)^t R_K^N (g_K^N)$.

IV. Update W_K by,

$$W_{K+1} = W_K + a_K g_K^N \qquad (3.60)$$

Having updated the weight vector by (3.60), it is held constant over 2N more data samples to allow the computation of g_{K+1}^{N}, $(g_{K+1}^{N})^{t}R_{K+1}^{N}(g_{K+1}^{N})$ and a_{K+1}. Note that for a_K equal to a constant the algorithm outlined above reduces to the conventional stochastic gradient descent algorithm (2.38) [8, 16, 18, 79, 80].

It is worth mentioning that the larger the block size the closer the estimates R_K^N and P_K^N are to R and P respectively. In this case BIELS adaptive algorithm behaves like that of deterministic steepest descent method with ELS, with a convergence rate affected by the condition number of R. Therefore, the convergence speed deteriorates as the condition number of R increases. This problem can be overcome by employing a conjugate gradients method (CGM) which yields the solution after a finite number of iterations.

3.2.12 Minimization by a conjugate gradients method

The CGM was first introduced by Hestenes and Stiefel [82] for solving a set of simultaneous linear equations having a symmetric positive definite matrix of coefficients and it was later extended to non-linear functions by Fletcher and Reeves [83]. There are many variations of the CGM proposed by various authors. Reference [14] presents a full treatment of CGM techniques.

Generally, CGM is a modification of the method of steepest descent with ELS in which the successive direction vectors q_K are chosen so that q_K is a linear combination of the negative gradient at the K-th iteration and previous direction vectors, q_{K-1}, q_{K-2}, ..., q_1, such that the R-orthogonality condition,

$$q_K^t R q_j = 0 \qquad K \neq j \tag{3.61}$$

is satisfied. As a consequence of (3.61) any point in an L-dimensional space can be reached by a suitable linear combination of at most L of them. A brief discussion of the concept of conjugacy and linear independence, quadratic termination and the construction of conjugate search vectors is given in Appendix 4. For a detailed discussion see [13, 14].

It is shown in Appendix 4 that the successive direction vectors can be obtained by,

$$q_K = g_K + b_K q_{K-1} \tag{3.62}$$

where, b_K is a positive scalar given by,

$$b_K = \frac{|g_K|^2}{|g_{K-1}|^2} \tag{3.63}$$

Using the direction vector (3.62) in (3.9) the following algorithm can be established [13, 14],

$$\begin{aligned} g_K &= -\nabla J(W_K) = P - RW_K \\ b_K &= g_K^t g_K \,/\, g_{K-1}^t g_{K-1} \\ q_K &= g_K + b_K q_{K-1} \\ a_K &= q_K^t g_K \,/\, q_K^t R q_K \\ W_{K+1} &= W_K + a_K q_K \end{aligned} \tag{3.64}$$

where, $b_1 = 0$ and $q_1 = g_1$.

The scalar a_K can be computed by

$$a_K = g_K^t g_K \; / \; q_K^t R q_K \tag{3.65}$$

since $q_{K-1}^t g_K = 0$ (q_{K-1} is orthogonal to g_K as it is shown in Appendix 4) which results in

$$\begin{aligned} q_K^t g_K &= (g_K + b_K q_{K-1})^t g_K \\ &= g_K^t g_K \end{aligned} \tag{3.66}$$

Minimization by algorithm (3.64) yields the minimum after L-iterations (neglecting the round-off error). This is true if the condition number is not too large and in the case of an ill conditioned situation the algorithm (3.64) takes longer to converge (2L to 3L iterations) [13, 14].

Clearly the adaptive implementation of CGM algorithm (3.64) involves a great amount of computation if the weight vector W_K is to be updated for each data sample. This high computational requirement is a drawback for on line signal processing.

The computational requirement of algorithm (3.64) can be substantially reduced if block data adaptation is employed along with the assumption of stationary data.

3.2.13 Adaptive implementation of a conjugate gradients algorithm

Assume stationary data so that (3.55) and (3.56) are satisfied. The the same techniques used to compute g_K^N and $(g_K^N)^t R_K^N (g_K^N)$ in the BIELS adaptive algorithm can be used to compute g_K^N and $q_K^t R_K^N q_K$ in algorithm (3.64). This results in the following block CGM (BICGM) adaptive algorithm.

$$g_K^N = \frac{1}{N} \sum_{i=1}^{N} e_i X_i$$

$$b_K = (g_K^N)^t (g_K^N) \ / \ (g_{K-1}^N)^t (g_{K-1})$$

$$q_K = g_K^N + b_K q_{K-1} \tag{3.67}$$

$$a_K = (q_K)^t (g_K^N) \ / \ \frac{1}{N} \sum_{j=1}^{N} ((q_K)^t X_j)(X_j^t (q_K))$$

$$W_{K+1} = W_K + a_K q_K$$

where, $b_1 = 0$ and $q_1 = g_1^N$.

Algorithm (3.67) is the adaptive version of algorithm (3.64) in which the weight vector W_K is held constant over 2N data samples, N data samples to allow the computation of g_K^N and N more data samples to allow the computation of $q_K^t R_K^N q_K$.

Therefore, each algorithm iteration requires 2N data samples to allow the computation of $a_K q_K$. Note that BIELS and BICGM adaptive algorithms outlined in section (3.2.11) and the present section are valid only for processing of stationary data. Also in the case of BICGM adaptive algorithm, the estimates R_K^N and P_K^N must be a realistic estimate of R and P to enable L-step convergence. In other words, BICGM adaptive algorithm (3.67) will converge after L iterations (L2N data samples) if the size of each block of data is large enough so that the following conditions are satisfied.

$$R_K^N = R_{K+1}^N = \;\ldots\ldots\; \simeq R \tag{3.68}$$

$$P_K^N = P_{K+1}^N = \;\ldots\ldots\; \simeq P \tag{3.69}$$

An accelerated gradient method using the exact linear search and Schmidt orthogonalization technique is proposed in [5] for adaptive array processing. The technique is an adaptive version of the algorithm proposed by Powell [14, 84], for finding stationary values of a function. The technique requires far more iterations than L-iterations to converge which contradicts the basic definition of L-step convergence. This is due to the lack of an appropriate restarting technique and small blocks of data.

It will be verified by simulations that the BICGM algorithm outlined in this section is superlinearly convergent (L to 2L-step convergence) in a well condition situation, a property that can be guaranteed even in an ill condition situation by adopting a restarting method.

3.2.14 Computational complexity of BIELS and BICGM adaptive algorithms

Considering BIELS and BICGM algorithms it can be seen that by adopting block data adaptation, computational requirement of

ELS and CGM adaptive algorithms is significantly reduced. This is because of elimination of all matrix vector products which require L^2 multiplications. Also, using block adaptation eliminates the need for storing a matrix (or upper triangular matrix) which simplifies the implementation even further. BIELS and BICGM algorithms require in the order of L multiplications per algorithm iteration.

Obviously if block data adaptation is to be adopted, then BICGM algorithm is far more efficient than BIELS algorithm since it guarantees L-step convergence for a slight increase in complexity of the system.

Note that to implement the BIELS and BICGM adaptive algorithms parallel processing is required, one processor with the weights equal to W_K and one processor with the weights equal to q_K (or g_K in the case of BIELS).

3.2.15 <u>On the choice of the initial estimate of the weight vector in accelerated gradients algorithms</u>

Defining an error vector $V_K = (W_K - W^*)$, a quadratic function can be expressed by

$$J(W_K) = J(W^*) + V_K^t R V_K \qquad (3.70)$$

If W_1 is chosen equal to W^*, then any iterative algorithm terminates in one step since $W_1 = W^*$ and $J(W_1) = J(W^*)$. But this is hardly the case because W^* is not known apriori. Considering the second term on the right hand side of (3.70), it is clear that the larger $|V_1|$, the larger would be the error term $V_1^t R V_1$. A suitable choice for W_1 would be $W_1 = 0$ which results in a maximum initial error.

$$V_1^t R V_1 = (W^*)^t R (W^*) \tag{3.71}$$

Although all descent algorithms do converge for any arbitrary initial value of the weight vector, an arbitrary W_1 can give rise to a large initial value of error $V_1^t R V_1$ which is undesirable. The value of $W_1 = 0$ is the recommended value for all descent algorithms.

3.2.16 On the implementation of the exact linear search gradient technique using lattice structure

A linear combiner using lattice structure instead of a TDL involves two minimization procedures, one due to that of finding an optimum set of reflection coefficients and the other due to finding a set of optimum combiner weights. As it was discussed in section (2.6.5), the most important advantage of using the lattice structure instead of TDL is its orthogonalizing property which decouples successive stages enabling optimization of each stage of the lattice independently.

Minimization of individual cost functions at each stage of the lattice by adopting ELS yields a step length which is inversely proportional to the sum of the FD and BD prediction error power p^i at that stage. Since $\bar{p}^i$ is not known, an estimate of it $\tilde{p}^i$ can be used (see equation 2.74) which is the procedure adopted in most conventional gradient algorithms.

These algorithms use an instantaneous estimate of the gradient at each iteration K (see equation (2.73) and Appendix 2), which has an adverse effect on the speed of convergence. To overcome this problem, similar estimators to (3.13), (3.15) and (3.17) (one dimensional) can be used to generate a more realistic estimate of the gradient. Clearly the trade-off would be the increased computational requirement.

Due to orthogonalized signals generated by lattice structure, the combiner weights can be updated individually using a step length inversely proportional to the signal power at the corresponding lattice output and an instantaneous (or a more realistic) gradient estimate of one dimensional quadratic cost function due to that weight. This is the method adopted in conventional lattice adaptive gradient algorithms (see equations 2.75 - 2.78).

However, the lattice filter outputs are a set of orthogonal signals only after complete convergence of the reflection coefficients. Thus there exists an initial degree of correlation between these signals which decreases as the reflection coefficients approach their optimal value. The larger the condition number of the process, the higher is the initial degree of correlation between the lattice output signals. Therefore, the initial convergence rate of the conventional gradient algorithms deteriorates as the eigenvalue spread gets larger. This problem can be overcome by using a combined update recursion for the combiner weights and adopting ELS method.

It should be noted that the most interesting property of the ELS technique is its rapid initial rate of convergence "irrespective of the eigenvalue spread of the covariance matrix" (although its final rate of convergence deteriorates as eigenvalue spread gets larger). Therefore, there would be very little to gain (or none) to implement the ELS technique by a lattice linear combiner instead of implementing it using a TDL linear combiner. This is as a result of increased algorithm noise and higher computational requirement due to the lattice structure. The above discussion will be verified by simulations in chapter 5.

3.3 Linearly constrained minimization by ELS

Referring to the sections 2.7 and 2.7.1, the augmented cost function for a linearly constrained problem can be written as,

$$J(W,\gamma) = \tfrac{1}{2}(W^tRW - 2P^tW + c) + \gamma^t(C^tW - f) \qquad (3.72)$$

where, γ^t = vector of the Lagrange multipliers, C^t = m by L matrix of the linear constraints and the factor $\frac{1}{2}$ is added for simplifying the analysis.

The negative gradient of $J(W_K, \gamma)$ with respect to W_K is

$$- \nabla J(W_K, \gamma) = g_K - C\gamma_K \qquad (3.73)$$

where, g_K = negative gradient of the quadratic function. Equating (3.73) to zero yields,

$$C\gamma_K = g_K \qquad (3.74)$$

Multiply both sides of (3.74) by C^t, then

$$C^tC\gamma_K = C^tg_K \qquad (3.75)$$

Therefore,

$$\gamma_K = (C^tC)^{-1}C^tg_K \qquad (3.76)$$

where, γ_K is an estimate of the Lagrange multipliers corresponding to W_K. Substitute (3.76) into (3.73) to obtain,

$$\begin{aligned} -\nabla J(W_K, \gamma) &= g_K - C(C^tC)^{-1}C^tg_K \\ &= (I - C(C^tC)^{-1}C^t)g_K \\ &= Zg_K \qquad (3.77) \end{aligned}$$

where, $Z = (I - C(C^tC)^{-1}C^t$ is a projection matrix and operates on the negative gradient g_K in order to convert it to a form orthogonal to the constraint matrix C. The projection matrix Z represents the null space of the constraints, so that CZ= 0 (ie. rows of the constraint matrix C are orthogonal to the columns of the projection matrix). The negative gradient of the augmented function is a projected gradient of the quadratic function.

Adopting the steepest descent method, the update recursion for W_K can be expressed by,

$$W_{K+1} = W_K + a_K Z g_K \tag{3.78}$$

Equation (3.78) represents the projection gradient algorithm (PGA).

An alternative form of PGA can be established by considering the update recursion for W_K using the negative gradient of the augmented function.

$$\begin{aligned} W_{K+1} &= W_K + a_K \nabla J(W_K, \gamma_K) \\ &= W_K + a_K(g_K + C\gamma_K) \end{aligned} \tag{3.79}$$

Substitute for W_{K+1} from (3.79) into the constraint equation, $C^t W_{K+1} = f$ to obtain,

$$C^t W_K + a_K C^t(g_K + C\gamma_K) = f \tag{3.80}$$

Solve (3.67) for γ_K,

$$\gamma_K = (C^t C)^{-1} C^t g_K + \frac{1}{a_K}(C^t C)^{-1}(C^t W_K - f) \tag{3.81}$$

Substituting γ_K back into (3.79) results in

$$\begin{aligned} W_{K+1} &= (I - C(C^tC)^{-1}C^t)(W_K + a_K g_K) + C(C^tC)^{-1}f \\ &= Z(W_K + a_K g_K) + C(C^tC)^{-1}f \qquad (3.82) \end{aligned}$$

which is an alternative PGA.

A simple geometrical interpretation of the algorithms (3.78) and (3.82) is illustrated by Figure 3.11. A fundamental difference between these algorithms is that, the former algorithm projects the gradient g_K onto the subspace and adds the projected vector to the past weight vector (Figure 3.11a). However, the algorithm (3.82) makes the unconstrained step, projects onto the subspace and then adds the vector $C(C^tC)^{-1}f$, producing a new weight vector W_{K+1} that satisfies the constraints (Figure 3.11b).

This different updating procedure has a practical significance as explained below.

Practical implementation of constrained algorithms involves computational errors causing deviation of the weight vector from the constrained plane which deteriorates the performance of the algorithm. Consequently constrained algorithms must have an error correcting capability and this is a useful property of algorithm (3.82) as illustrated geometrically by Figure 3.12 (W_K is assumed to be off constraints).

In order to implement PGA using ELS technique, a_K and g_K can be computed by the same method outlined for unconstrained ELS adaptive algorithms in section (3.2.3). It follows that,

$$g_K = P_K - R_K W_K$$

$$a_K = g_K^t g_K \; / \; g_K^t R g_K \qquad (3.83)$$

$$W_{K+1} = Z(W_K + a_K g_K) + C(C^t C)^{-1} f$$

with $W_1 = C(C^t C)^{-1} f$. R_K and P_K must be computed according to stationary or non-stationary data (estimators; 3.13, 3.15, 3.17). Note that in the absence of a desired signal, $P_K = 0$ and $g_K = -R_K W_K$.

Application of algorithm (3.83) to the broad band array processing with look direction constraints is discussed in chapter 5 and simulation results are presented to illustrate its performance.

3.3.1 Linearly constrained minimization by CGM

To apply CGM to a linearly constrained problem requires that the problem be transformed to an unconstrained problem and be represented by a set of linear equations. The procedure usually involves the projection of the second derivative and the projection of the first derivative of the function [13].

Alternatively, a linearly constrained problem can be transformed to an unconstrained problem as explained below.

Referring to the augmented function (3.72), compute the gradient of $J(W, \gamma)$ with respect to W and γ, so that,

$$\begin{aligned} \nabla J(W, \gamma)_W &= RW - P + C\gamma \\ \nabla J(W, \gamma)_\gamma &= CW - f \end{aligned} \qquad (3.84)$$

Equating (3.71) to zero yields the linear equations

$$\begin{aligned} RW + C\gamma &= P \\ CW + 0 &= f \end{aligned} \qquad (3.85)$$

For simplicity, write (3.85) as,

$$\tilde{R}\tilde{W} = \tilde{D} \qquad (3.86)$$

where,

$$\tilde{R} = \begin{bmatrix} R & C \\ C^t & 0 \end{bmatrix} \qquad (m + L \text{ by } m + L) - \text{matrix}$$

$$\tilde{W}^t = (W, \gamma) \qquad (m + L) - \text{vector}$$

$$\tilde{D}^t = (P, f) \qquad (m + L) - \text{vector}$$

Given R, C, P and f, the linear equation (3.86) can be solved by a CGM to obtain $\tilde{W}^*$ which contains the optimum weight vector W^* and the optimum vector of the Lagrange multipliers γ^*. It should be emphasised that the augmented matrix $\tilde{R}$ is not necessarily a positive definite matrix and CGM may terminate giving a wrong solution. This complicates the problem since the CGM method must be modified to ensure (m + L) - step convergence to a right solution. A modified CGM is proposed in [14] which guarantees (m + L) - step convergence but involves a great amount of computation per iteration and even a block implementation does not appear to be a practical approach. However, the above procedure is outlined only to demonstrate a theoretical approach to the problem.

Generally, the application of the CGM to a constrained problem causes substantial increases in computational requirement, in particular if one sample adaptive implementation is adopted. Some reduction in the required computations may be achieved if a block implementation is employed with stationary data.

3.4 Quadratically constrained minimization by ELS

Referring to section (2.8), the augmented function for a quadratically constrained problem can be written as,

$$J(W, \gamma) = W^tRW - \gamma(W^tW - B) \qquad (3.87)$$

where, γ = Lagrange multiplier and B is a positive constant usually set to 1 so that $W^tW = 1$. Taking the negative gradient of (3.87) with respect to W and equating it to zero yields the eigenvalue-eigenvector relationship.

$$RW = \gamma W \tag{3.88}$$

where, γ is the eigenvalue corresponding to eigenvector W. Multiply both sides of (3.88) by W^t to obtain,

$$W^tRW = \gamma W^tW$$
$$\gamma = \frac{W^tRW}{W^tW} \tag{3.89}$$

Note that if $W^tW = 1$, then

$$\gamma = W^tRW \tag{3.90}$$

Equation (3.89) is known as Rayleigh quotient. Minimum eigenvalue m of the symmetric positive definite matrix R is formed by minimizing the Rayleigh quotient [13, 14, 15]. Minimization of (3.87) or (3.89), (with B = 1), by ELS technique results in the following constrained algorithm (based on the same principles discussed in section (3.2.1)),

$$\begin{aligned} g_K &= m_KW_K - RW_K \\ a_K &= g_K^tg_K \,/\, g_K^tRg_K \\ \tilde{W}_{K+1} &= W_K + a_Kg_K \\ W_{K+1} &= \tilde{W}_{K+1} \,/\, |\tilde{W}_{K+1}| \end{aligned} \tag{3.91}$$

where, $m_K = W_K^tRW_K$.

The last step is added to satisfy the norm constraints $W_K^t W_K = 1$. The initial estimate of the weight vector W_1 must satisfy the norm constraints, eg. $W_1^t = (1, 0, 0 \ldots\ldots)^t$. Adaptive implementation of algorithm (3.91) involves an estimate of the covariance matrix R_K, that must be estimated by a suitable estimator (eg. 3.13, 3.15, 3.17).

Application of algorithm (3.91) to the field of adaptive spectral estimation is discussed in detail in chapter 5 and simulation results are presented to illustrate its performance.

3.4.1 Quadratically constrained minimization by CGM

Minimization of (3.87) by CGM requires the modification of the algorithm (3.91) so that

$$\begin{aligned} \tilde{W}_{K+1} &= W_K + a_K q_K \\ q_K &= g_K + b_K q_{K-1} \\ a_K &= g_K^t q_K \,/\, q_K^t R q_K \end{aligned} \tag{3.92}$$

where, b_K must be chosen such that q_K be a linear combination of g_K and $q_{K-1}, q_{K-2}, \ldots\ldots$. The algorithm is not a practical approach for adaptive signal processing due to its computational requirement. A block implementation of the algorithm may be useful if a long data record is available. For a typical quadratically constrained CGM algorithm see [74].

3.5 Discussion and conclusions

The main objective of this chapter was to design rapidly converging robust adaptive algorithms for various fields of adaptive signal processing. Throughout the chapter the emphasis was on the method of exact linear search. Due to their rapid initial speed of convergence and moderate computational requirement, the ELS adaptive algorithms appear to be attractive alternatives to RLS algorithms for relatively well condition problems.

The final convergence rate of the ELS algorithm deteriorates as the condition number increases. This is hardly a problem in the case of adaptive signal processing since it is the initial convergence speed which plays a crucial role in a non-stationary situation, not the final convergence speed.

The ELS adaptive algorithms involve only one matrix-vector product which is the price to pay for a realistic gradient estimate (residual vector) of the cost function.

Adopting block adaptation (for stationary data) and a parallel processing technique, the computational requirement of the ELS technique can be reduced to an order of L. Also block adaptation enables the generation of a set of mutually conjugate search vectors which, when used sequentially with ELS technique result in quadratic termination (with a computational requirement in the order of L).

The application of accelerated gradient techniques to the fields of adaptive noise cancelling, broad band adaptive array processing with look direction constraints and adaptive spectral estimation is discussed in chapter 5 and simulation results are presented to illustrate their performance.

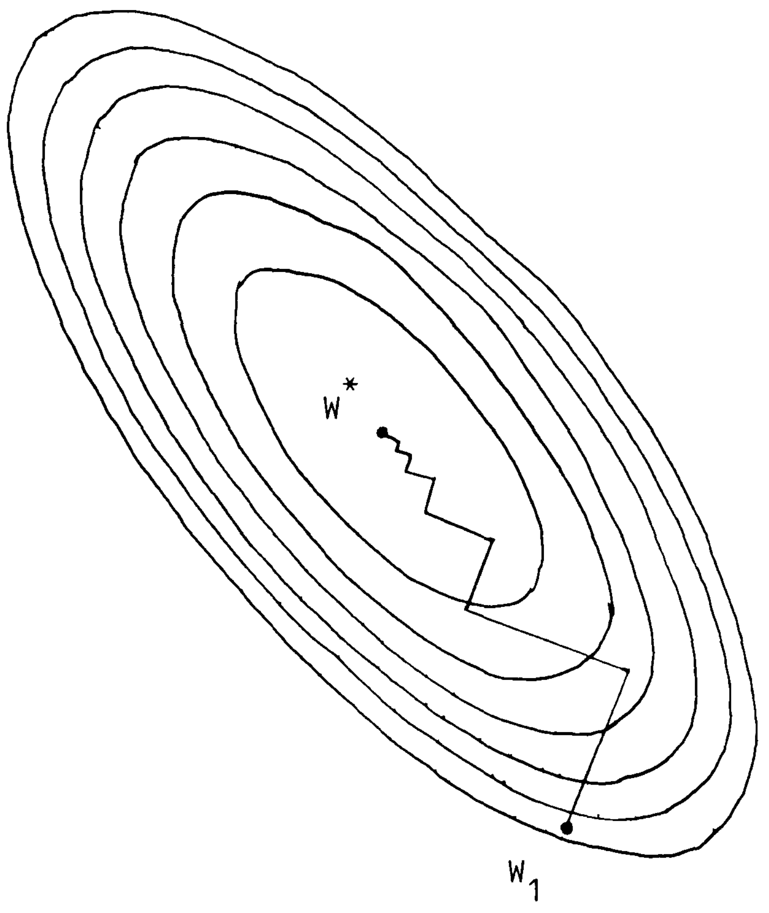


Figure 3.1 : The zig-zag progress of the steepest descent method with ELS on a typical quadratic function with ellipsoidal contours.

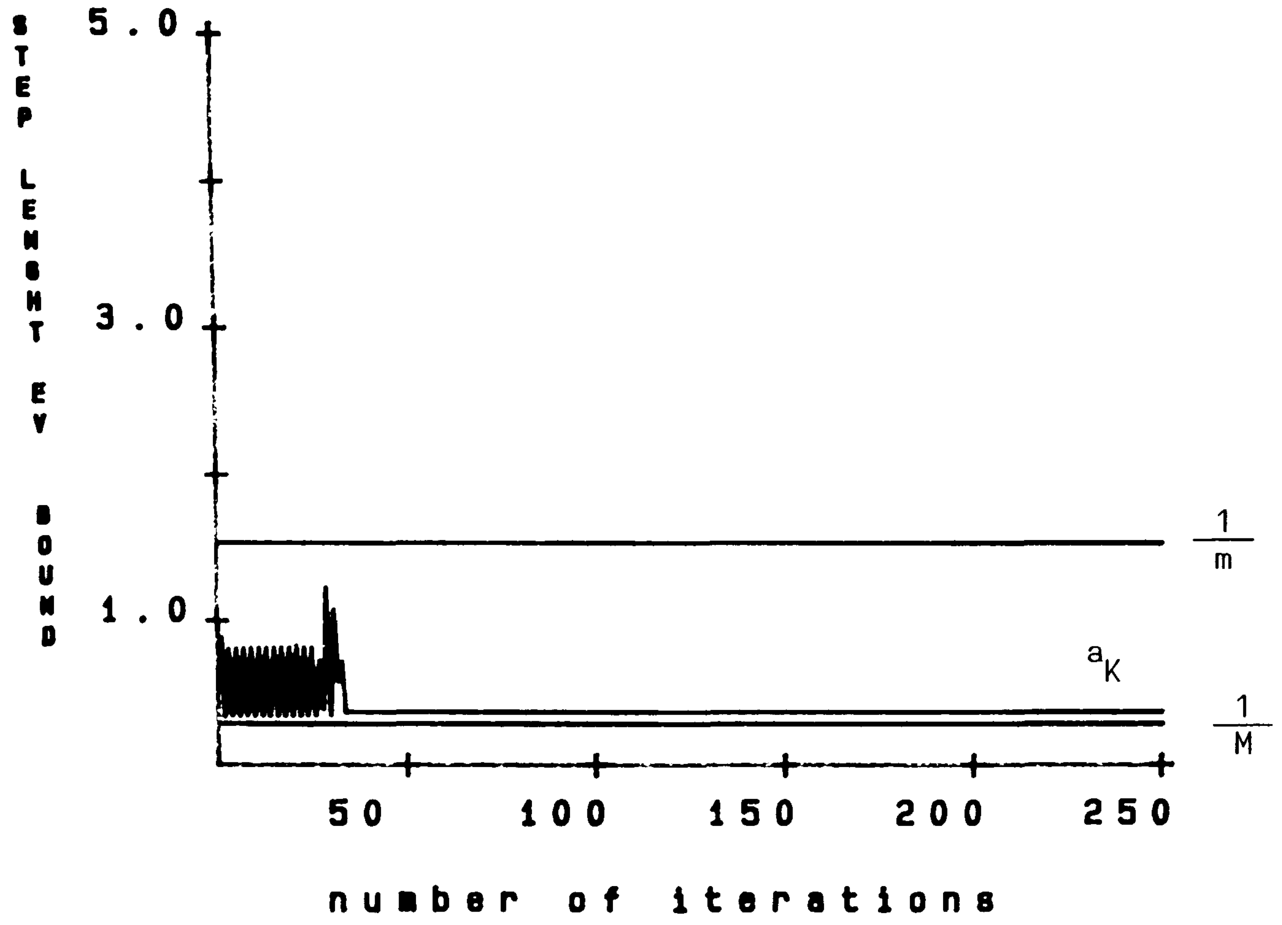


Figure 3.2 : The relative behaviour of the convergence factor and eigenvalue bounds, condition number = 5

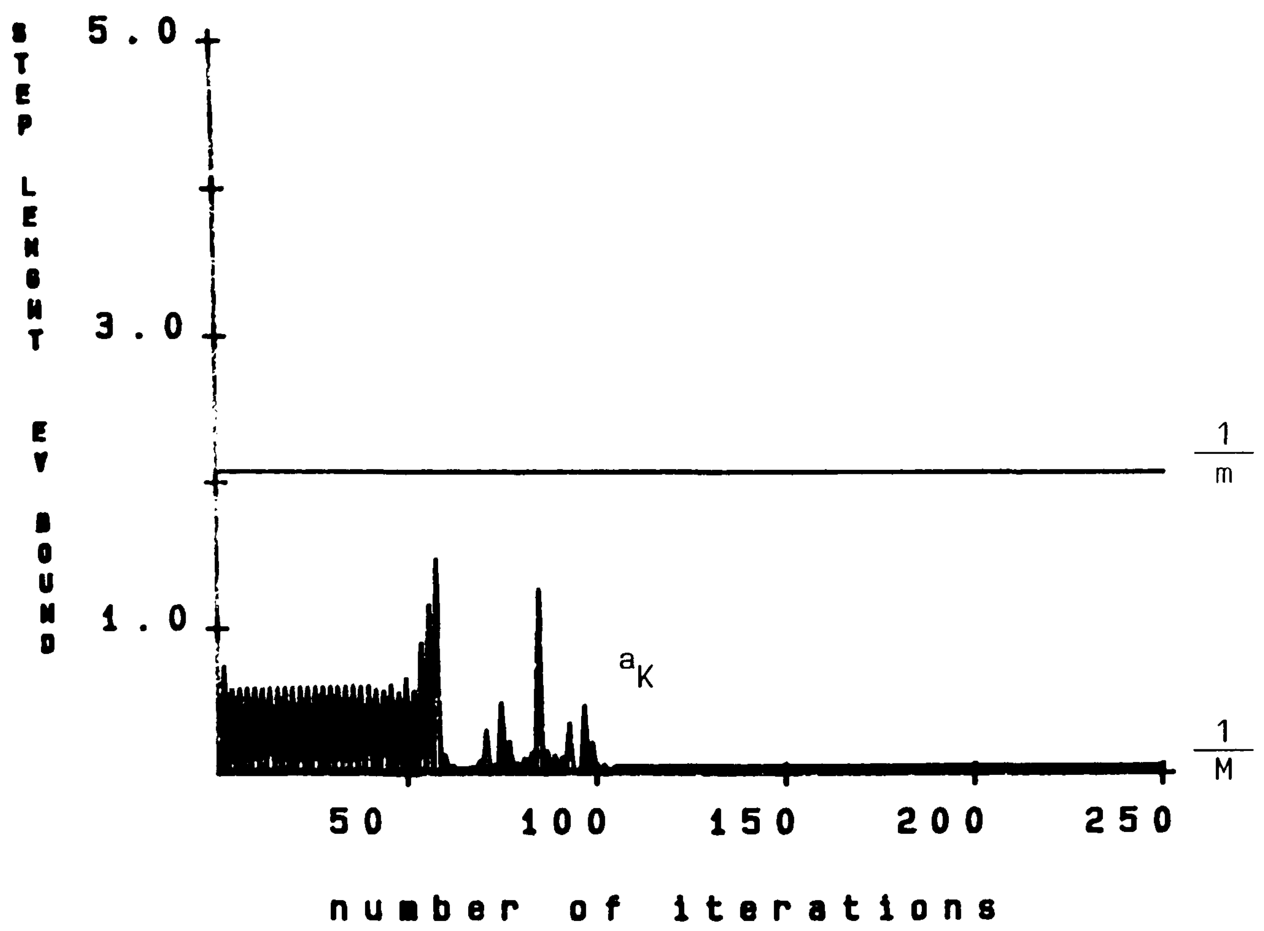


Figure 3.3 : The relative behaviour of the convergence factor and eigenvalue bounds, condition number = 100

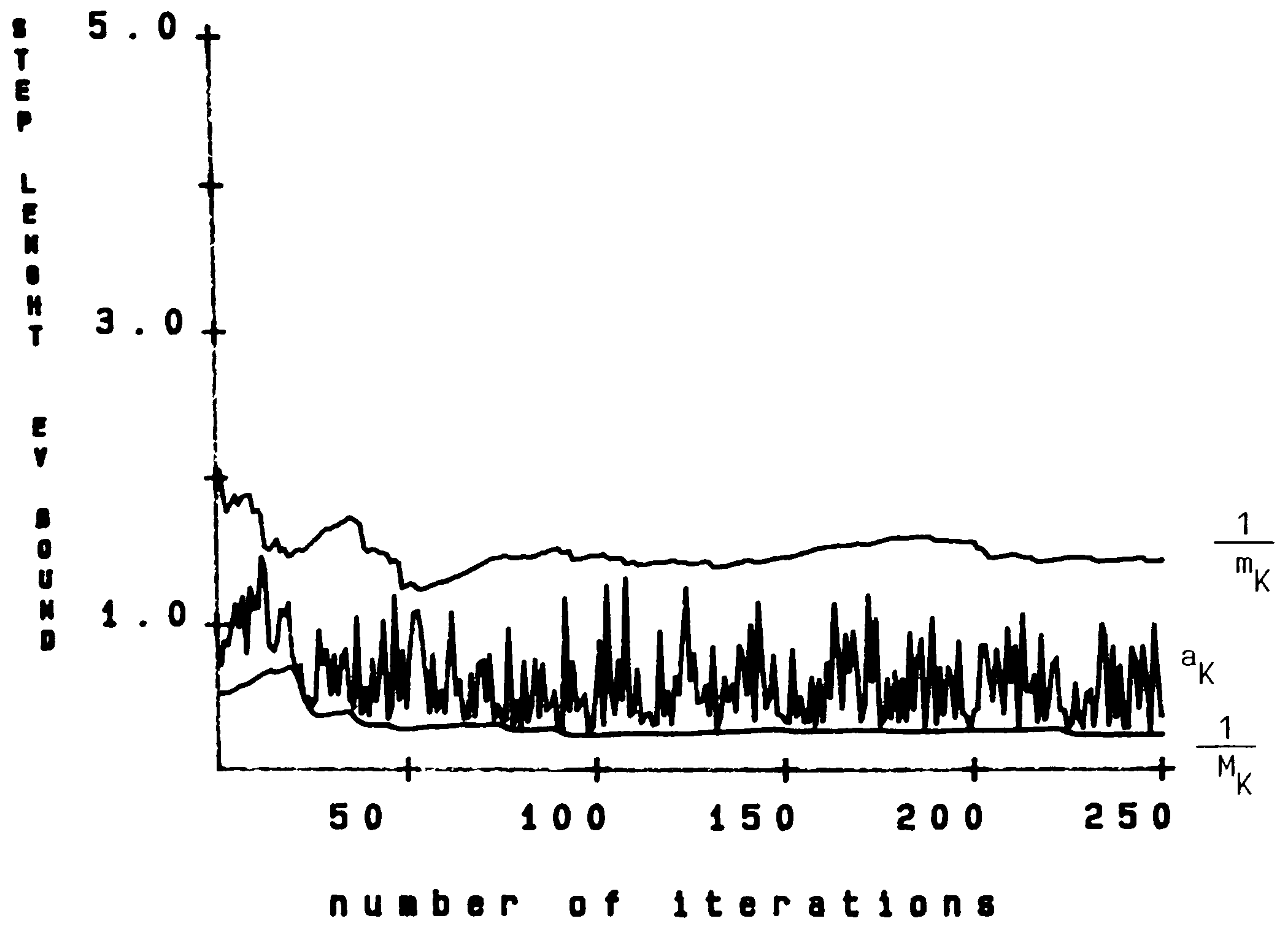


Figure 3.4 : The relative behaviour of the convergence factor and eigenvalue bounds. Estimator 3.13, condition number = 5.

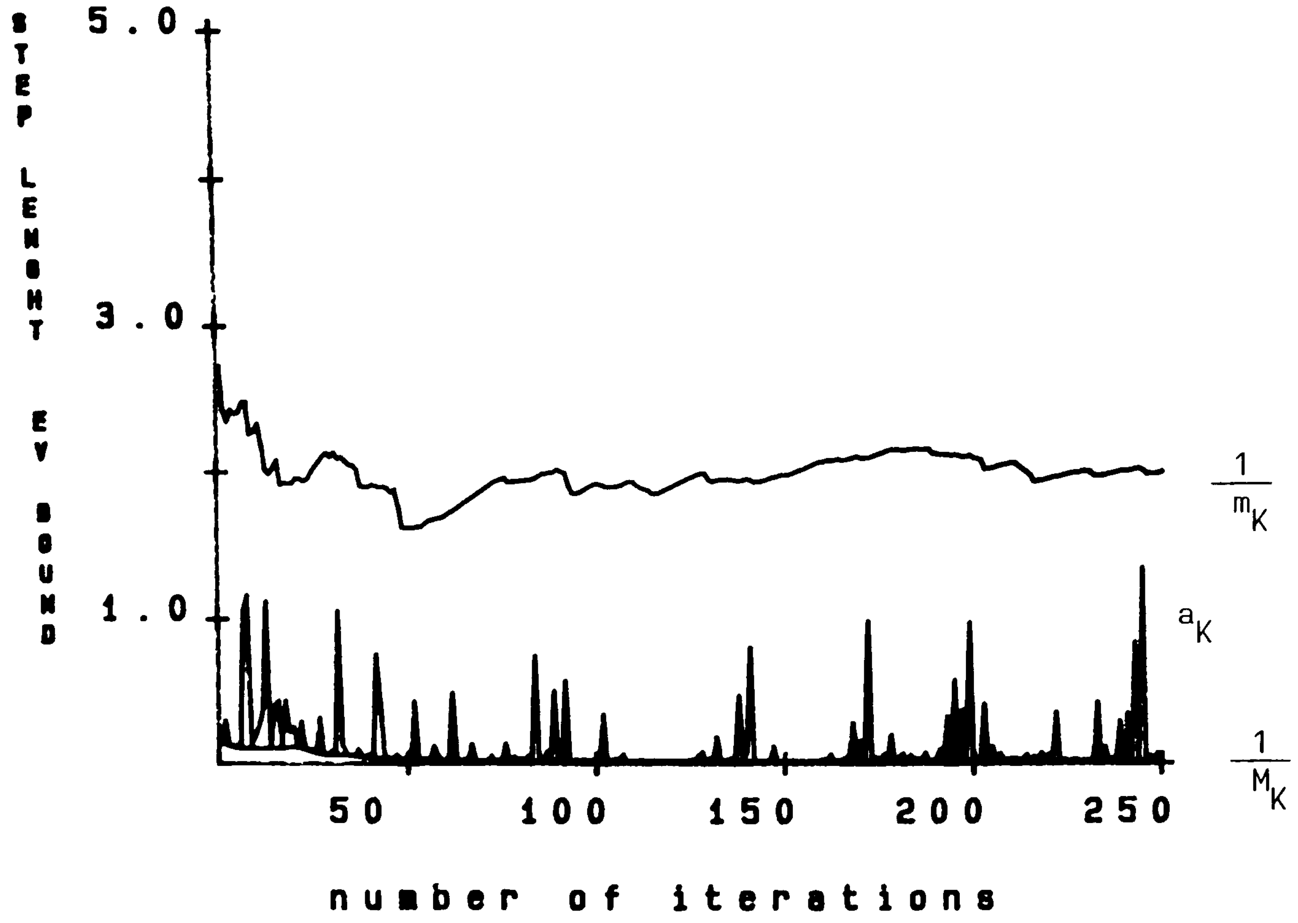


Figure 3.5 : The relative behaviour of the convergence factor and eigenvalue bounds. Estimator 3.13, condition number = 100.

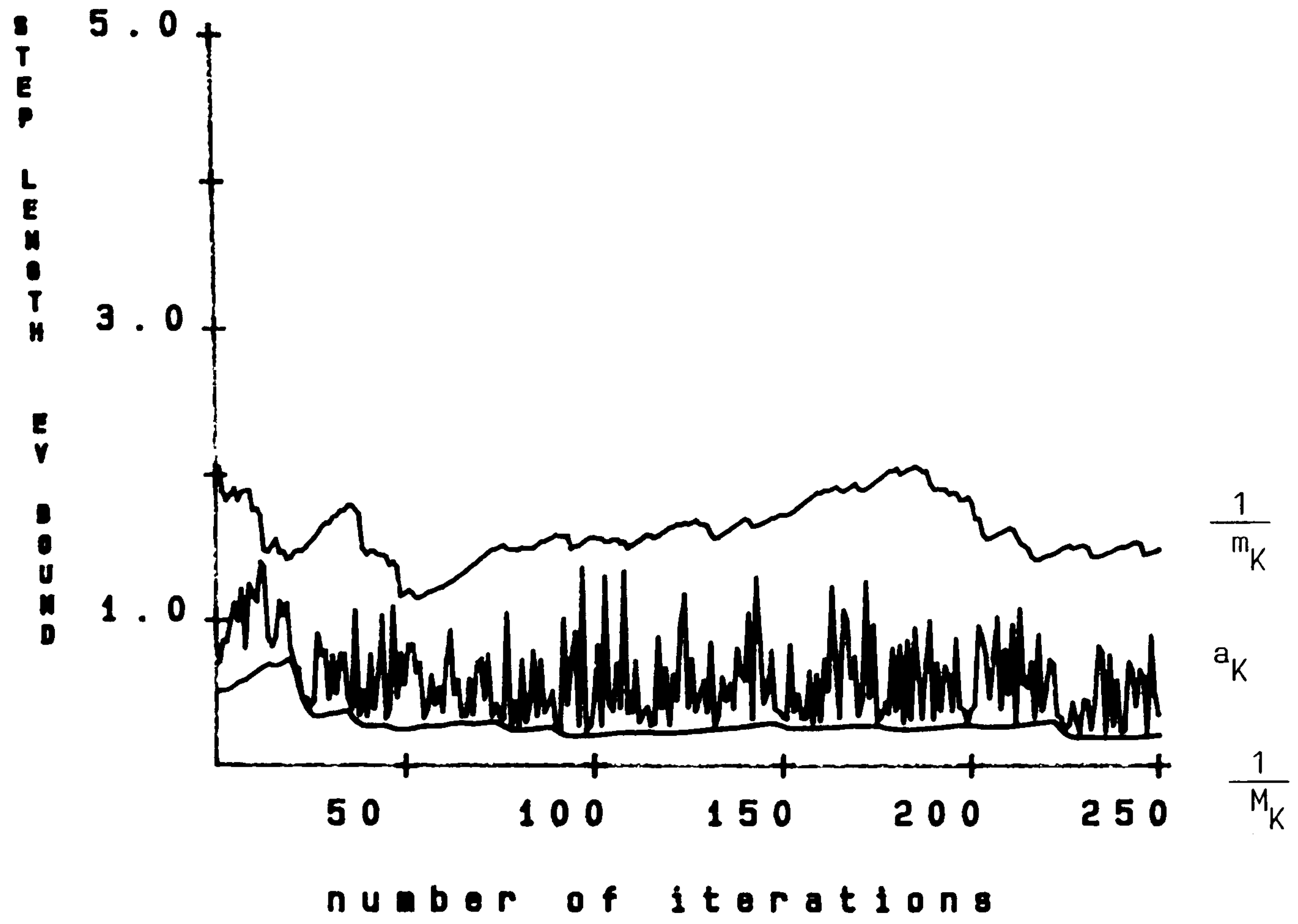


Figure 3.6 : The relative behaviour of the convergence factor and eigenvalue bounds. Estimator 3.15b, condition number = 5.

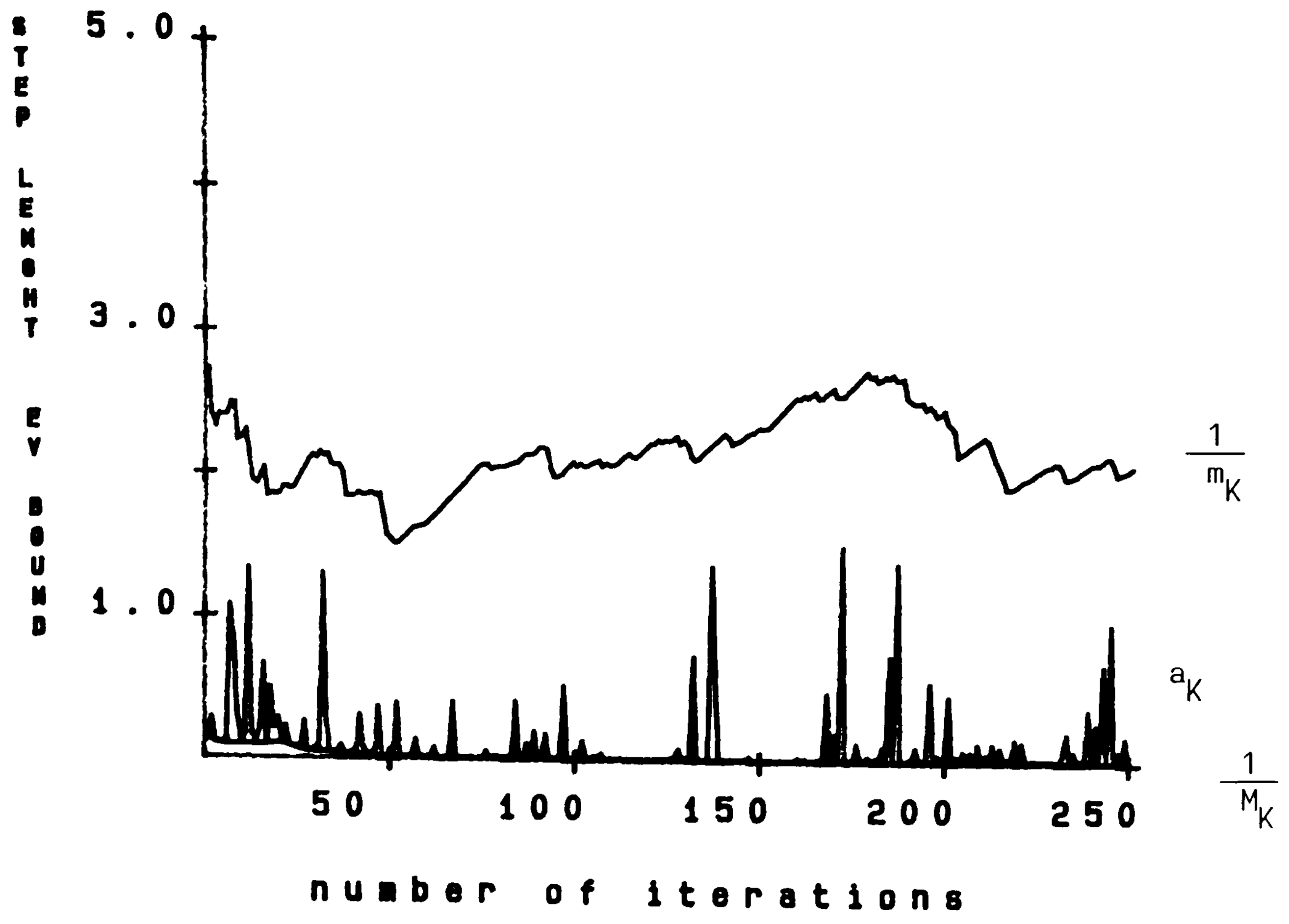


Figure 3.7 : The relative behaviour of the convergence factor and eigenvalue bounds. Estimator 3.15b, condition number = 100.

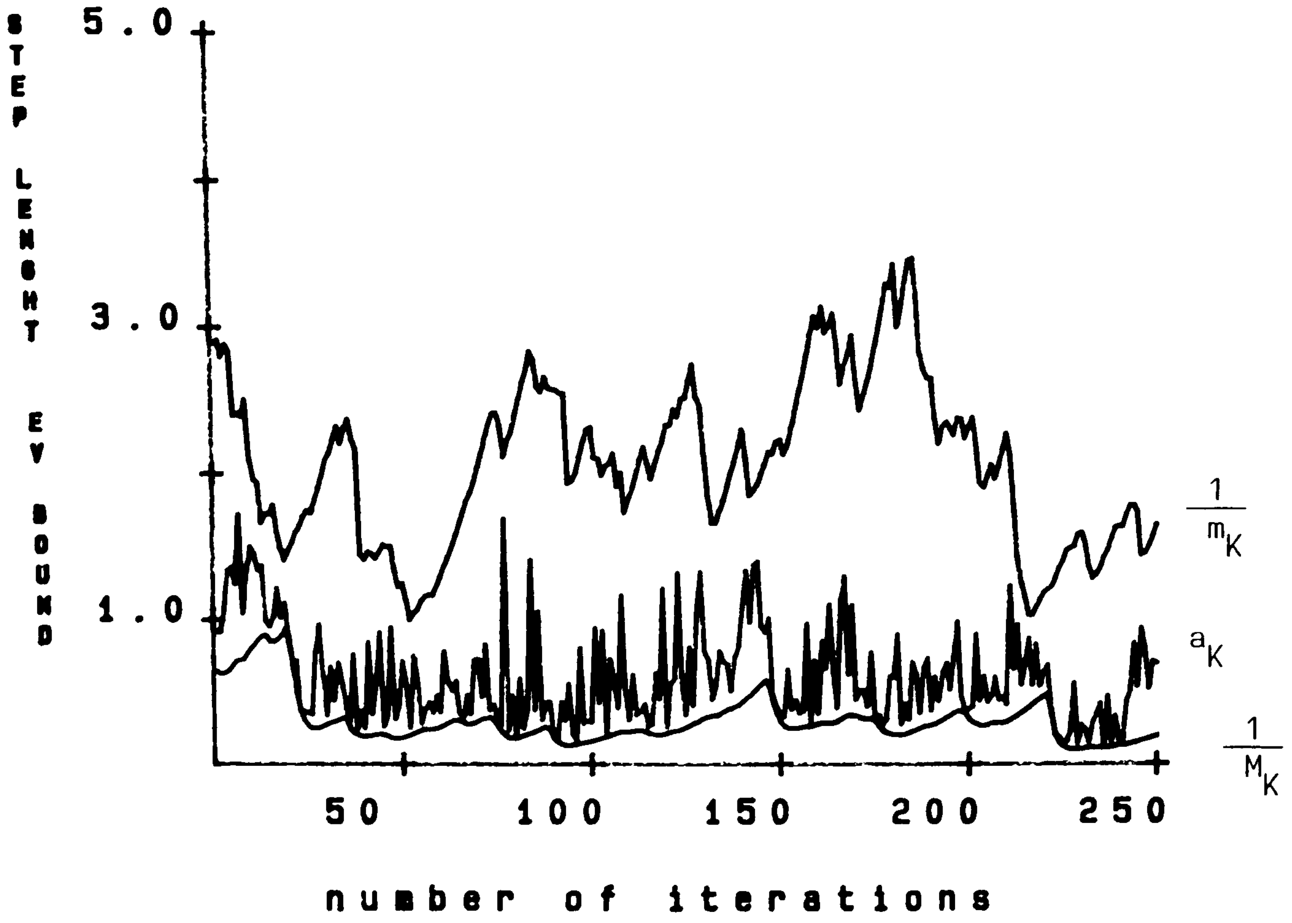


Figure 3.8 : The relative behaviour of the convergence factor and eigenvalue bounds. Estimator 3.17, condition number = 5.

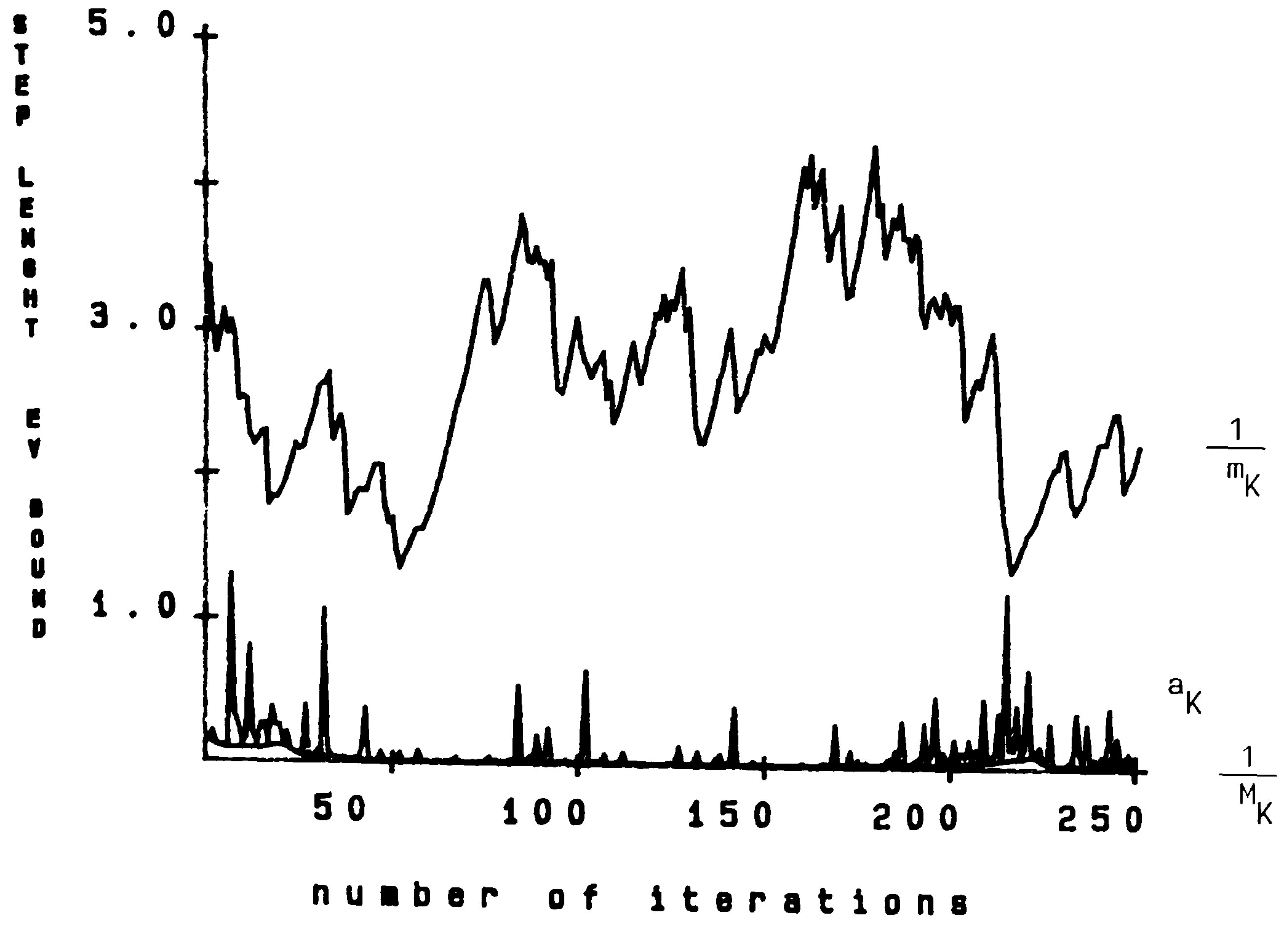


Figure 3.9 : The relative behaviour of the convergence factor and eigenvalue bounds. Estimator 3.17, condition number = 100.

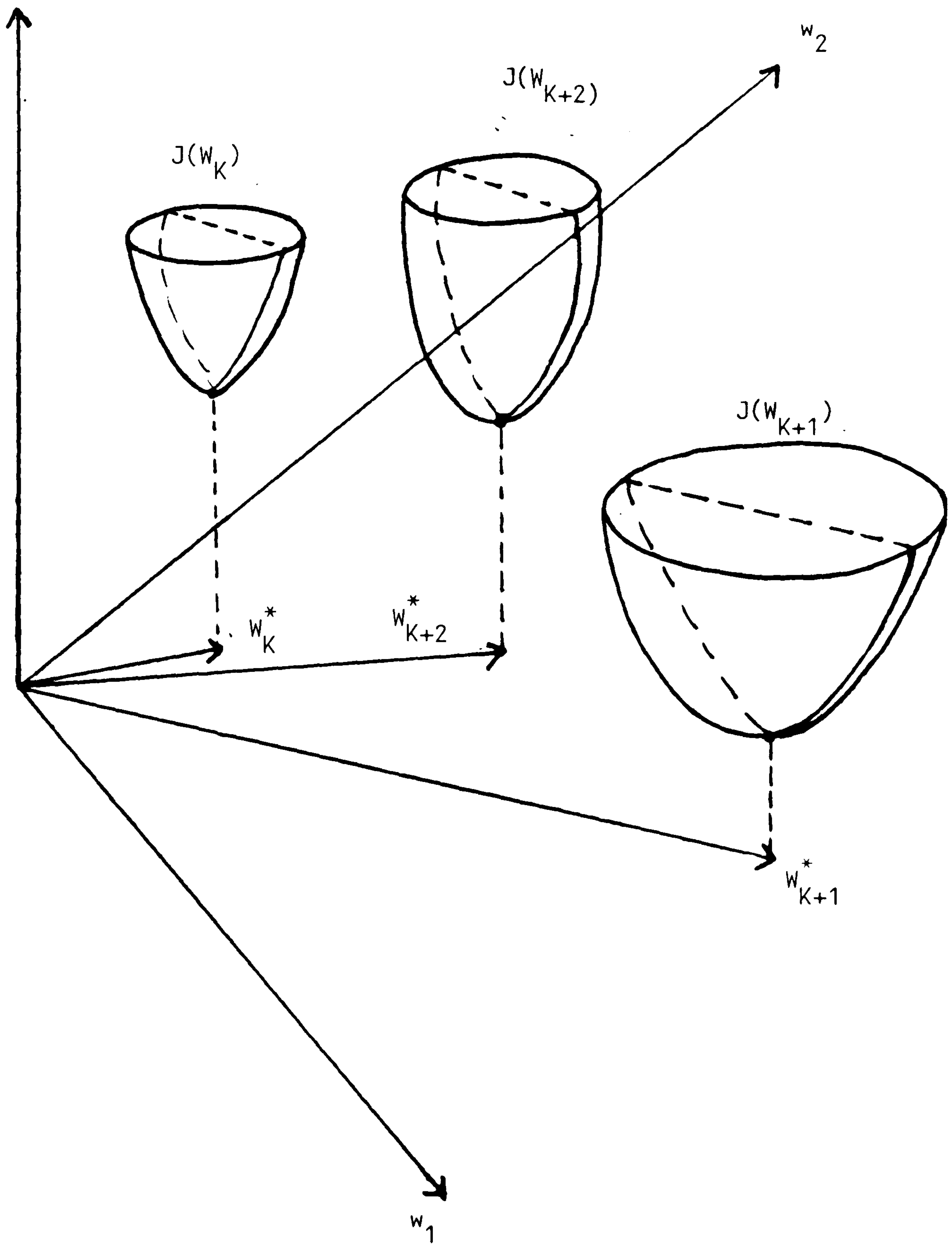
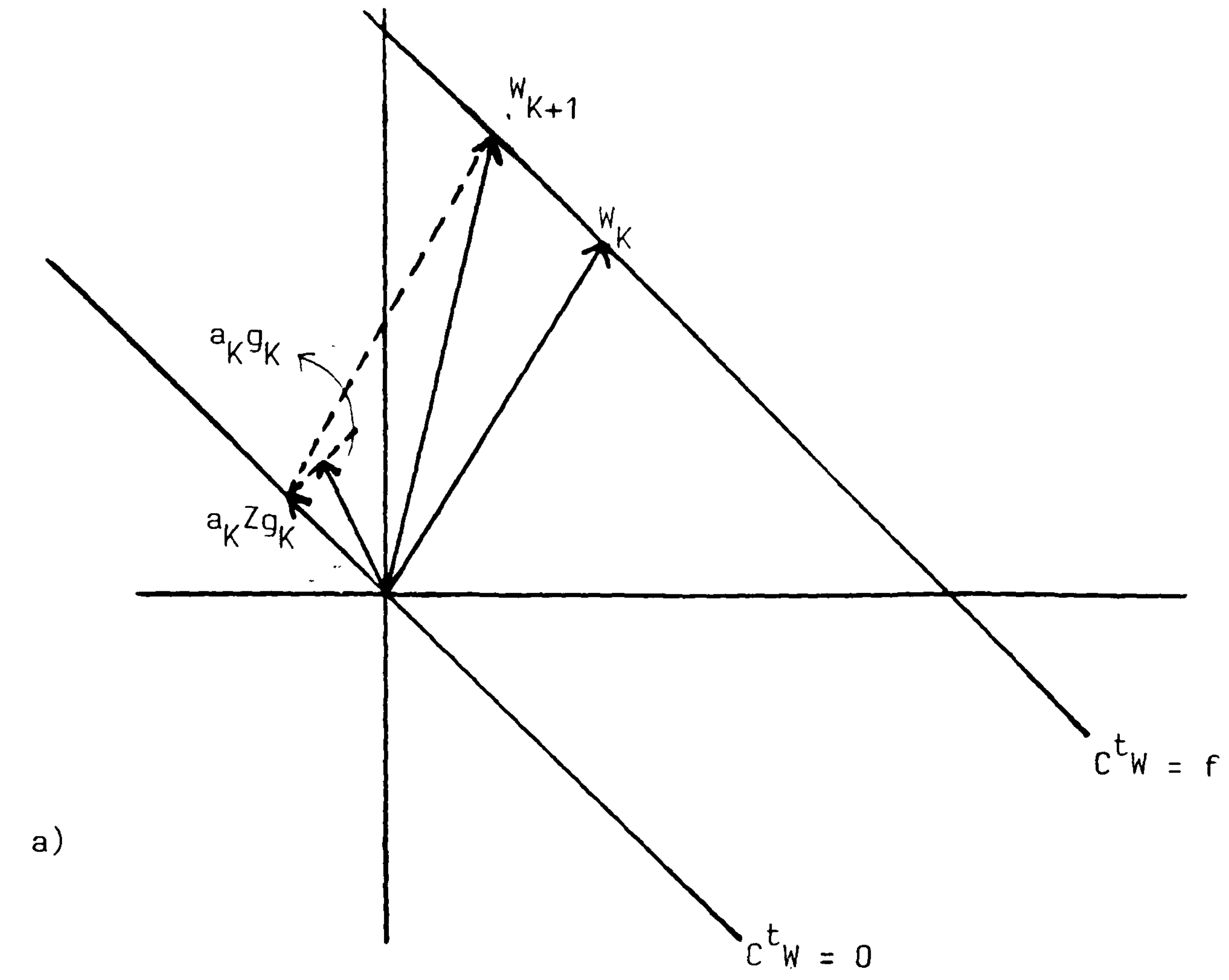


Figure 3.10 : Time varying cost function due to a non-stationary process



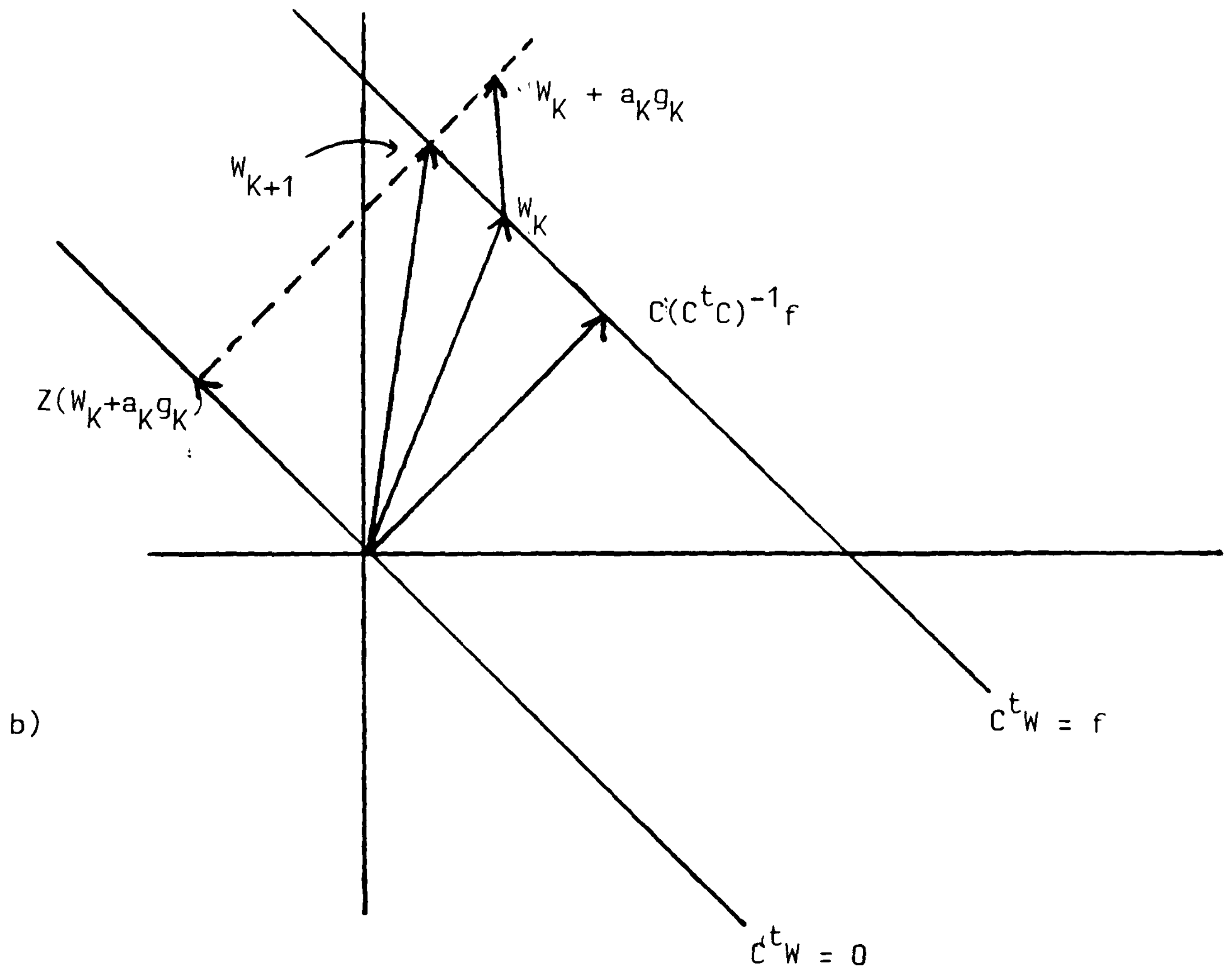
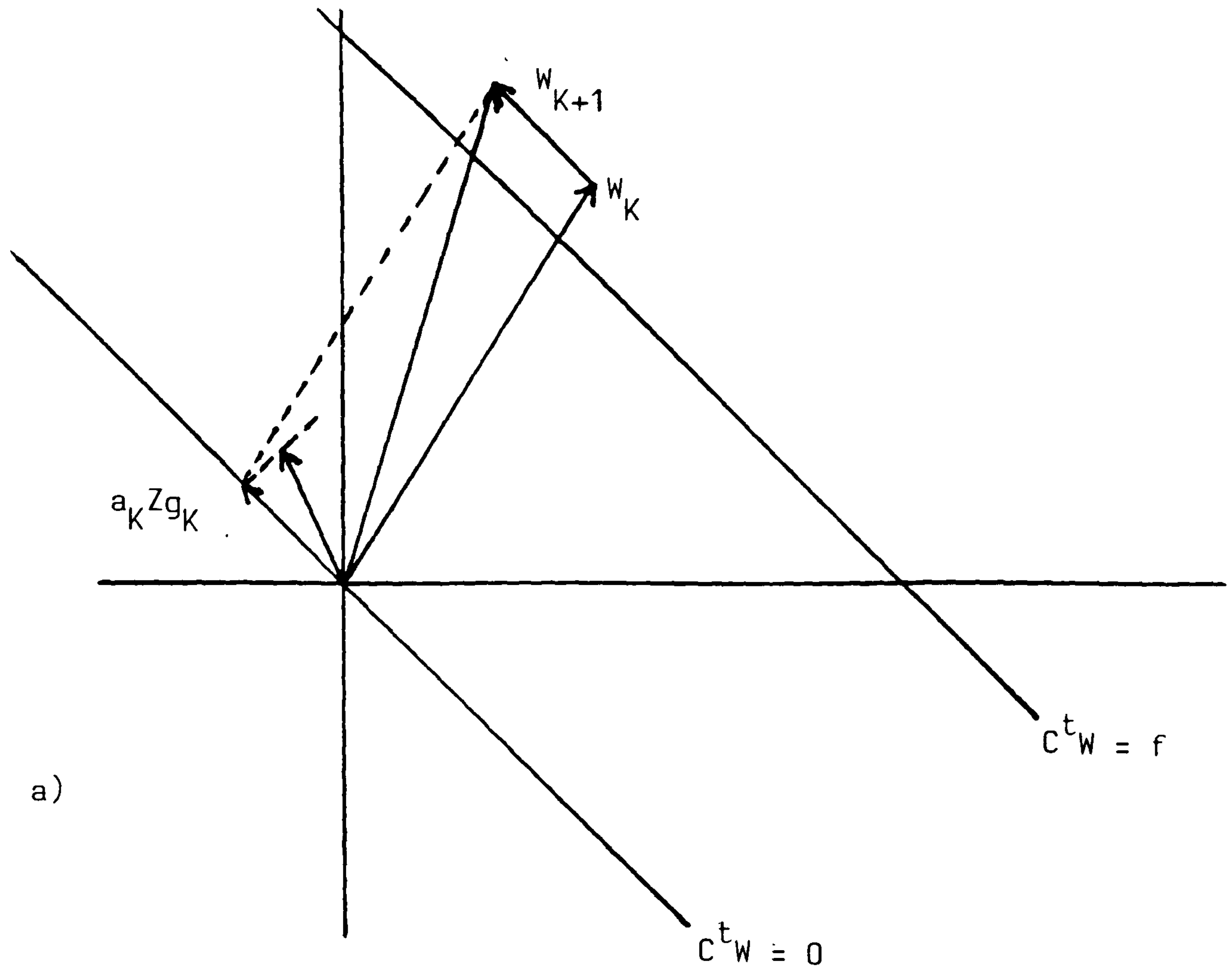


Figure 3.11 : Geometrical interpretations of the PGA algorithms.
a) algorithm 3.78   b) algorithm 3.82



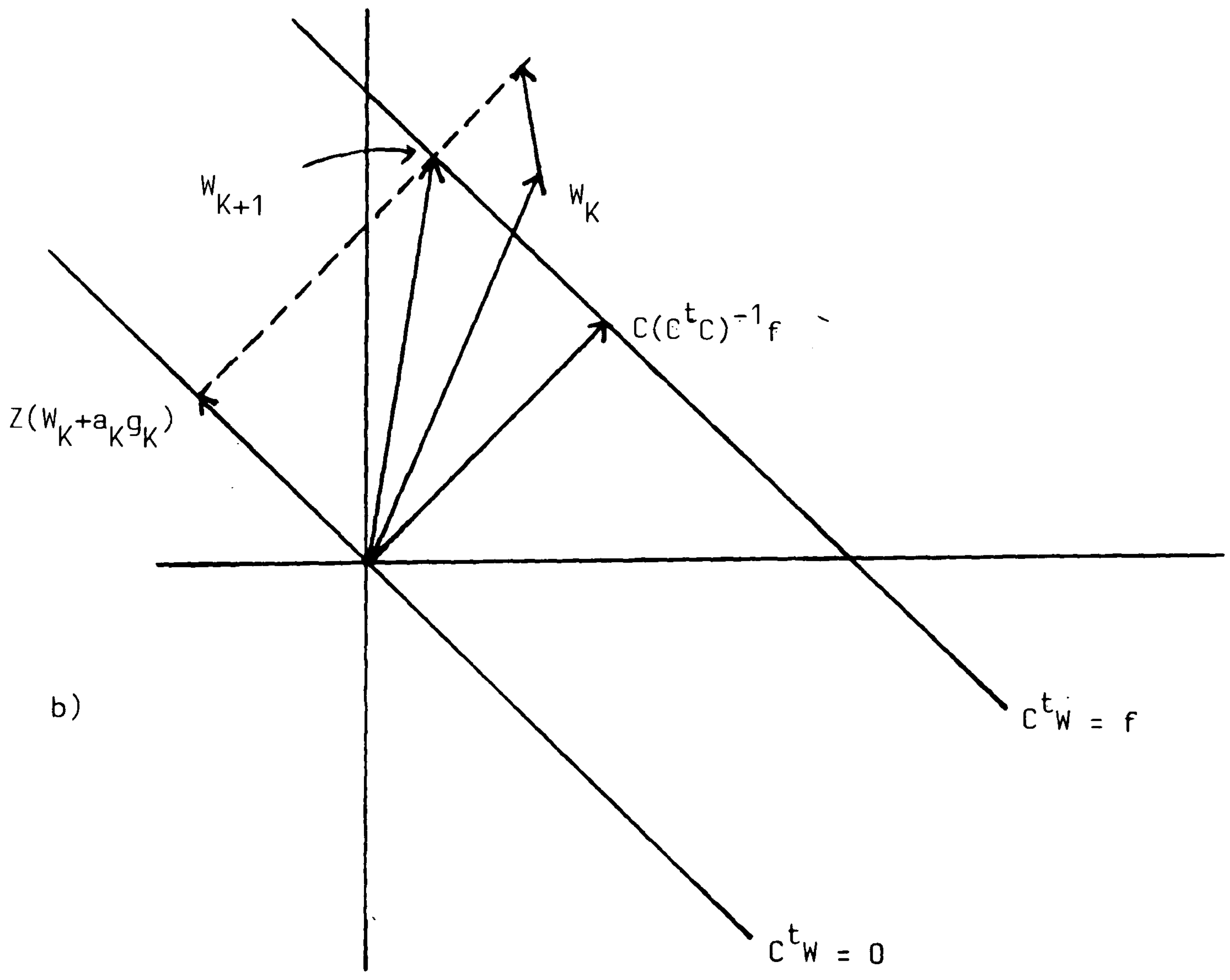


Figure 3.12 : Geometrical interpretations of the PGA algorithms. W_K is assumed off-constraints.

a) algorithm 3.78 (allows the error to accumulate)

b) algorithm 3.82 (corrects deviations from the constrai

L	DIRECT REALIZATION OF ELS ALGORITHMS	REALIZATION OF ELS ALGORITHMS USING THE SYMMETRICAL PROPERTY
3	55	45
5	131	100
10	461	325
20	1721	1150
30	3781	2475
50	10301	6575

Table a : The required number of multiplications of ELS algorithms

CHAPTER 4

NUMERICAL STABILITY OF AG ALGORITHMS

4.1 Effects of finite word length

Various algorithms were developed in chapter 3, assuming infinite precision implementation and no reference was made to the possible performance degradation due to the effects of finite word length. Practical implementation of these algorithms involves finite precision arithmetic. This results in performance degradation due to the quantization of the input signals, round-off errors as a result of each arithmetic operation and quantization of the digital filter coefficients.

4.2 Numerical stability of ELS adaptive algorithms

The ELS adaptive algorithms (for one sample update) involve the computation of estimates of the covariance matrix and cross-correlation vector (if it exists) and then performing an exact linear search. The quantization of the input signals and round-off errors due to finite precision arithmetic result in biased estimates of R and P. This has no effect on the stability of the algorithm (as long as the properties of R are preserved) simply because, instead of the original cost function, its approximation is minimized which yields an approximated solution.

Regarding the step length a_K, it is assumed to be the optimal step (at iteration K) satisfying the optimality condition. Due to finite precision arithmetic, the optimal step length cannot be determined exactly and its computed value might violate the optimality condition $\frac{1}{M_K} \leqslant a_K \leqslant \frac{1}{m_K}$.

Note that a_K can have any value within this bound. Thus the larger the eigenvalue bounds the faster would be the accumulation of errors (due to successive computation of g_K, a_K, W_{K+1}).

Logically, the violation of the lower bound has no effect on the stability of the algorithm, while resulting in the reduced convergence speed. It is the upper bound on a_K that plays a crucial role in the stability of the ELS adaptive algorithms. The algorithm will be unstable for $a_K > \frac{1}{m_K}$. To ensure that a_K does not exceed the upper bound, a relaxation factor $0 < \alpha < 1$ can be introduced, so that

$$\tilde{a}_K = \alpha a_K \qquad (4.1)$$

Obviously, the smaller the α, the slower would be the convergence speed of the algorithm.

Further errors are introduced because of finite precision implementation of $R_K W_K$, g_K, $a_K g_K$, W_{K+1}, y_K, and $d_K - y_K$ which result in performance degradation and possibly instability of the algorithm. The smaller the number of bits, the faster would be the accumulation of errors. A possible approach to decrease the accumulation of errors is to reduce the relaxation factor α in (4.1). The adverse effect would be reduced convergence speed.

Due to the time limitations of this project, it was not possible to carry out a detailed analytical study of the numerical stability of the ELS adaptive algorithms. However, simulation results are presented in chapter 5 which were carried out by considering the effects of finite word length implementation.

4.3 Numerical stability of block data implementation of ELS and CGM algorithms

In sections (3.2.11) and (3.2.13), it was shown that the computational load of BIELS and BICGM algorithms can be reduced substantially by assuming the stationary data and adopting a parallel processing technique. As a consequence of reduced computational load, the errors (due to finite precision implementation) do not accumulate as rapidly as the errors due to one sample iteration.

In BIELS and BICGM algorithms, a_K is computed using two different sets of data samples. For these algorithms to be stable, a_K must satisfy the eigenvalue bounds as mentioned in the previous section. Therefore, it is important that the size of the data block, over which a_K is computed, be large enough to enable a_K to preserve its optimality.

Numerical stability of CGM is discussed in detail in [14, 82] and various correction procedures are proposed. Theoretically, the CGM is L-step convergent. A property that can hardly be met in practice due to the propagation of round-off errors. The reasons for this are;

a. successive search directions lose conjugacy,

b. a_K cannot be computed exactly. While the former results in reduced convergence speed, the latter may cause instability.

It is shown in [14, 82], that if the initial estimate W_1 is chosen such that the residual vector g_1 is close to the eigenvector corresponding to the minimum eigenvalue of R, then the ratio a_K/a_{K-1} is always less than unity and the CGM is always stable with respect to round-off errors. However, this is not applicable in the case of adaptive implementationof CGM, since knowledge of such an initial estimate is not known apriori. In this case an applicable and simpler method to ensure stability, would be to modify the CGM by setting $b_K = 0$ [in algorithms (3.64) and (3.67)] so that $q_K = g_K$ and selecting a_K in accord with (4.1). This results in a relaxed ELS algorithm.

CHAPTER 5

RESULTS

5.1 Introduction

In this chapter application of the accelerated gradient techniques (developed in chapter 3) to the fieldsof adaptive noise cancelling, broad band adaptive array processing and adaptive spectral estimation is discussed and results are presented to illustrate their performance. The results are obtained by simulations performed on a VAX/VMS computer using single precision FORTRAN.

The numerical stability of the ELS adaptive algorithm when applied to an adaptive noise cancelling problem is demonstrated (and compared with the numerical stability of the RLS algorithm) by considering the effects of floating point round-off errors due to the finite precision implementation.

5.2 Adaptive noise cancelling (ANC)

A standard problem which occurs frequently in various fields such as control, communication, biomedical engineering, etc. is that of extracting a signal buried in noise (the term "noise" signifies all forms of interference). The usual method of estimating a corrupted signal is to pass it through a filter which tends to suppress the noise while leaving the signal relatively unchanged.

With regard to the type of signals and the available information about their statistics, different techniques like match filtering, kalman filtering can be employed to detect the signal. However, adaptive noise cancelling technique is used when the only available information is the fact that the signal is uncorrelated with the noise and if a secondary source is available which yields to a reference input correlated in some unknown way with the primary noise.

Figure 5.1 shows the basic problem and its adaptive noise cancelling solution [89]. A signal S is transmitted over a channel to a sensor which also receives a noise n_o uncorrelated with the signal. A second sensor receives a noise x uncorrelated with the signal but correlated in some unknown way with the noise n_o. This sensor provides a reference input to the canceller.

Assume that S, n_o, x, and the filter output y are statistically stationary with zero means and S is uncorrelated with n_o and x. It follows,

$$e = S + n_o - y$$

$$e^2 = (n_o - y)^2 + 2S(n_o - y) + S^2$$

$$E(e^2) = E((n_o - y)^2) + 2E(S(n_o - y)) + E(S^2)$$

$$E(e^2) = E((n_o - y)^2) + E(S^2) \qquad (5.1)$$

The signal power $E(S^2)$ will be unaffected as the filter is adjusted to minimize $E(e^2)$.

Therefore,

$$\min E(e^2) = \min E((n_0 - y)^2) + E(S^2) \qquad (5.2)$$

As a result of (5.2), minimizing $E(e^2)$ corresponds to minimizing $E((n_0 - y)^2)$. The filter output y is then the best least squares estimate of n_0 and e is the best least squares estimate of S. In other words, minimizing the total output power minimizes the output noise power and maximizes SNR [89].

In a non-stationary situation, the signal S, the noise n_0 and the reference signal x can be statistically non-stationary. In this case minimizing the mean squared error corresponds to tracking of a variable minimum point.

The advantage of the ANC technique is that no apriori knowledge of S, n_0, x or their interrelationship is required. ANC technique has been applied to various problems such as ECG and EEG in biomedical engineering, echo cancellation on long distance telephone lines, inter-symbol interference suppression, flight deck communication systems to remove the aircraft noise, suppression of high level transients and broad-band noise transducers in process control, improvement of auditorium accoustics, cancelling antenna sidelobe interference, cancelling periodic interference (eg. in the playback of speech or music in the presence of tape hum or turntable rumble), etc. [2,3,5,7,8,89].

The cost function for the system shown in Figure 5.1 is,

$$J(W) = W^tRW - 2P^tW + c \tag{5.3}$$

where,

$J(W) = E(e^2)$	Expected value of the output power
$R = E(XX^t)$	L by L data covariance matrix
$P = E(dX)$	L-vector of cross-correlation
$c = E(d^2)$	Constant scalar

W = L-weight vector of the adaptive filter

X = L-vector of the tap voltages (data vector)

d = Primary or desired signal

The main task of an adaptive algorithm is to minimize (5.3) as data becomes available. Since R and P are not known apriori, their statistical estimates are used.

5.2.1 Statistically stationary data

In order to compare the speed of convergence of the ELS adaptive algorithm (3.14) and the RLS algorithm, computer simulations were carried out using single precision FORTRAN.

The primary input (the desired signal) was taken to be a sine wave of amplitude 4.5 corrupted by additive Gaussian white noise of relative power 20dB (ie. SNR=-10dB). The reference signal was obtained by a second order autoregressive (AR) filter with its input Gaussian white noise of power 20dB, so that,

$$x(K) = n_o(K) + u_1 x(K-1) + u_2 x(K-2) \qquad (5.4)$$

where, u_1 and u_2 are the autoregressive (AR) parameters.

The size of the TDL adaptive filter was equal to ten. The RLS and ELS adaptive algorithms were initialized according to,

RLS ; $\alpha = 1.0$, $R_1^{-1} = 100I$, $W_1 = (0,0, \dots\dots)$

ELS ; $R_1 = X_1 X_1^t + I$, $W_1 = (0,0, \dots\dots)$

Note that in the case of ELS adaptive algorithm, the error signal e_K is not involved in updating the weight vector. Therefore, the block diagram for adaptive noise canceller is modified as is shown in Figure 5.2.

Example 1

With AR parameters equal to $u_1 = 0.6$ and $u_2 = 0.05$, the corresponding eigenvalue ratio (EVR) was 20 (ie. the ratio of the maximum to the minimum eigenvalue of the 10 by 10 data covariance matrix).

Figures 5.3-5.4, 5.5-5.6, 5.7-5.8 show the learning curves (averaged over 20 individual runs), the convergence of the norm of the weight vector $|W_K|$ and the output of the noise canceller (ie. e_K) for ELS and RLS algorithms respectively.

Example 2

To demonstrate the effect of EVR on the initial speed of convergence of the ELS algorithm, the AR parameters were set to $u_1 = 0.9$ and $u_2 = 0.05$. This resulted in EVR = 402.

Figures 5.9 through to 5.14 present the results for ELS and RLS algorithms (based on the same order of presentation as example 1).

5.2.2 Statistically non-stationary data

The tracking capability of the ELS algorithm (3.16) and the RLS algorithm was tested by the following experiments.

Example 3

Signal, noise and reference inputs were generated as explained in the previous examples, but this time the signal amplitude, noise power and AR parameters were abruptly changed after a specific number of iterations. Table 5.a outlines the values of the various parameters over 500 iterations.

Figures 5.15 through 5.20 show the learning curves (averaged over 20 individual runs), tracking capability of $|W_K|$ and the output waveforms for ELS algorithm (3.16) (R and P were estimated by 3.15C) and RLS algorithm.

Example 4

The same as example 3 but the noise power was kept constant.

Table 5.b outlines the values of the various parameters during 500 iterations.

Figures 5.21 through 5.26 depict the results for ELS algorithm (3.16) and RLS algorithm (based on the same order of presentation as example 3). Note that the tracking behaviour of the ELS and RLS algorithms depend on the value of factor α . The smaller the α , the noisier would have been the results.

5.2.3 Adaptive noise cancelling using lattice filter

One of the advantages of using a lattice filter instead of a TDL is to decorrelate the input signal (reference signal), thus reducing the EVR. This is the main reason that the lattice gradient algorithms converge faster than their TDL counterparts.

As mentioned in section (3.2.16), using a lattice filter (instead of a TDL) and employing an ELS adaptive algorithm would have no effect on the initial speed of convergence for reasons that were discussed in that section.

However, to compare the performance of ELS and a conventional lattice gradient algorithm (see equations 2.73 - 2.78), the following simulation was carried out.

Example 5

The same as example 2 but TDL was replaced by a 10-stage lattice filter. Figures 5.27 and 5.28 show the learning curves (averaged over 20 individual runs) for ELS and a conventional lattice gradient algorithm (see equations 2.73 - 2.78) respectively.

From these results, it is clear that, although the ELS algorithm performs better than a conventional lattice gradient algorithm, its performance is degraded with respect to its TDL implementation (see Figure 5.9). This verifies the remarks made in section (3.2.16).

5.2.4 Adaptive noise cancelling using BIELS and BICGM algorithms

To examine the performance of BIELS and BICGM algorithms (3.60 and 3.67) simulations were performed as is explained in the following examples.

Example 6

The signal amplitude (a sine wave) = 4.5, noise power = 10dB, filter size = 5, AR-parameters u_1 and u_2 = 0.6 and 0.08, EVR = 12, and the block size = 50 data samples. The weight vector was initialized as W_1 = (0,0, ...). Figures 5.29 and 5.30 illustrate the behaviour of the norm of the weight vector for BIELS and BICGM respectively. As it is expected, the convergence behaviour of both algorithms is relatively similar at low EVR.

Example 7

The same as example 6, but the AR-parameters were, u_1 = 0.9, u_2 = 0.08 and EVR = 365. Figures 5.31 and 5.32 show the convergence of the norm of the weight vector for BIELS and BICGM algorithms respectively.

These results clearly demonstrate the superior performance of BICGM algorithm and its quadratic convergence behaviour. Also, these results reveal the most interesting property of BIELS algorithm which is its rapid initial rate of convergence irrespective of EVR (consider the convergence behaviour of the algorithm over the first ten iterations).

Note that in these simulations the size of the block of data was 50. The larger the block size, the closer would be the performance of the BIELS and BICGM algorithms to the performance of their non-stochastic versions. A small block size deteriorates the performance and may result in instability, mainly because the step length, a_K, is computed using two different blocks of data.

5.2.5 Finite precision implementation effects

Recall that, the results presented so far in this chapter were obtained by simulations performed on a VAX/VMS computer using single precision FORTRAN, ie. 32 bit floating point arithmetic operations.

To examine the numerical stability of any algorithm the simulations can be performed using a micro-computer with smaller processor size or writing the programs using Assembly language thus allowing control of precision.

Alternatively, the simulations can be carried out using a high level language on any computer system but taking into account the effects of the errors that would have been introduced as a result of lower precision operations. This technique can demonstrate (although not precisely) the performance degradation as a result of finite precision operations.

The error introduced by floating point operations can be modelled as follows [96].

$$\begin{aligned} fl(xy) &= xy(1+e_m) \\ fl(x+y) &= (x+y)(1+e_a) \end{aligned} \tag{5.5}$$

where, e_m and e_a denote errors due to the floating point multiplication and floating point addition and can be modelled as zero mean white noise independent of x and y, xy. It is shown in [97] that the variance of e_m and e_a is approximately given by $0.18 * 2^{-2B}$, where B is the number of bits used to present the mantissa. Note that the e_m and e_a are statistically independent.

To illustrate the effects of floating point round-off errors on ELS adaptive algorithm, the following simulation was carried out.

Example 8

The signal amplitude (a sine wave) = 4.5, noise power = 20dB, filter size = 5, AR-parameter = 0.9 and 0.08, EVR =365 and the weight vector was initialized as $W_1 = (0,0, \ldots\ldots)$. Two independent white noise sequences of variance $0.18 * 2^{-16}$, (ie. B = 8bit) were added to all required arithmetic operations (in accord with (5.5)) in order to illustrate the effects of floating point round-off errors. Figure 5.33 shows the learning curve (averaged over 20 individual runs) for ELS adaptive algorithm (3.16) --- R_K and P_K were computed by (3.15C) with $\alpha = 0.99$. From this result it is clear that the algorithm is stable.

The same experiment was repeated for RLS algorithm with forgetting factor equal to 0.99. The algorithm was unstable (rapidly diverging) for B = 8 and B = 16 bits. Figure 5.34 shows the learning curve for B = 20 bits verifying the sensitivity of the RLS algorithm to the effects of finite precision arithmetic. This is a well recognised feature of the RLS algorithm.

The above simulations by no means demonstrate the precise effects of floating point errors (and quantization errors due to quantizing the input signals and filter coefficients). However, they give an indication of relative numerical stability of ELS adaptive algorithms with respect to standard RLS algorithms.

5.3 Broad band adaptive array processing

Adaptive arrays are becoming increasingly important in such fields as radar, sonar and communications. The fundamentals of adaptive arrays are by no means new, but applications in practice have been limited both by technology and lack of robust algorithms suitable for real time operations. However, today's computing power cost enables complicated signal processing in real time at an economical cost. This has led to an increased interest in designing robust and computationally demanding algorithms for adaptive array processing (eg. RLS,ELS).

In order to examine the performance of an ELS adaptive algorithm, when applied to a linearly constrained problem, the practical broad band array processing system with the look direction constraints [60] is simulated.

The advantage of imposing the look direction constraints on array processing is that the only apriori/required information is the desired "look" direction (or the direction of the arrival of the desired signal). The technique ensures that a chosen frequency response of the overall array processor is maintained while the interference and the background noise are minimized. For detailed explanations see [5, 60].

Referring to Figure 2.7, the augmented cost function for the broad band antenna system under look direction constraints can be expressed by ,

$$J(W,\gamma)= W^tRW + \gamma^t(C^tW - f) \qquad (5.6)$$

where,

$R = E(XX^t)$ mL by mL data covariance matrix

γ^t = L-vector of the Lagrange multipliers

C^t = mL by m matrix of the linear constraints

f = m-vector of the look direction

X = mL-vector of the tap voltages (data vector)

W = mL-weight vector

L = number of sensors

m = number of taps per sensor

Computer simulations were performed for a linear array of five elements, each being processed through a filter of five tap weights. The environment had three broadband point noise sources of power 1.0, 10.0 and 10.0 located at 0, 30, 45 degrees respectively. The background and sensor noise was set to 0dB per sensor. The look direction was assumed to be 0 degrees (normal to array).

The broad band signals were generated by passing Gaussian white noise sequences through three digital bandpass filters with different passbands and centre frequencies. There was no correlation between the desired signal (incident at 0 degrees) and two strong interferences (incident at 30 and 45 degrees). Figure 5.35 shows the power spectrum of the incident signals (128 data samples).

Simulations were carried out using the constrained LMS algorithm (2.114) [5, 60] and the constrained ELS algorithm (3.83). In the case of the LMS algorithm, the convergence factor, a, was set to 0.00001 (note that the eigenvalues of the covariance matrix were not computed and the step size was chosen by trial and error to achieve the fastest convergence and at the same time ensuring the stability of the constrained LMS algorithm). In the case of the constrained ELS algorithm (3.83), the covariance matrix was estimated by (3.13), an equal data weighting. Both algorithms were initialized according to $W_1 = C(C^tC)^{-1}f$.

Figures 5.36 and 5.37 show the power spectrum of the array output sequence (128 data samples, ie. 128 iterations) obtained from LMS and ELS algorithms respectively. Complete suppression of the interferences by ELS algorithm and far less noisier output spectrum (unlike the output spectrum when the LMS algorithm was used) verify the rapid speed of convergence of the algorithm. However, the price to pay for this result is increased complexity which can be a drawback for applications with simplicity requirement.

5.4 Adaptive spectral analysis by eigenvalue-eigenvector decomposition of the data covariance matrix

Eigenvalue-eigenvector decomposition (EED) algorithms are well recognised in yielding the most accurate spectrum of narrow band signals in additive noise. These algorithms are based on the Pisarenko's method of spectral estimation [90, 91] and involve the computation of the minimum eigenvalue and the corresponding eigenvector of the data covariance matrix. The EED algorithms offer tolerance to the correlated and noisy signals (unlike the maximum likelihood and the maximum entropy techniques which can lead to severe bias and loss of resolution due to the correlated signals and noisy environment).

Applications of the EED algorithms include such fields as radar, sonar and communications whenever there is a need to detect and remove (or enhance) narrow band signals which are either buried in broad band noise or are interfering with the broad band desired signal.

Recently, several authors have addressed various techniques for adaptive implementation of Pisarenko's method [65-76, 92-95] for estimating the frequency of the sinusoids buried in broad band noise and estimating the directional spectra of narrow band signals received by spatially distributed arrays of sensors.

In this section the application of the method of ELS to the above problems is discussed and simulation results are presented to illustrate its performance. The fundamentals of Pisarenko's method are not discussed here, although detailed explanations are given in [90, 91]. Reference [65] presents an extension of Pisarenko's frequency estimation method to the estimating of the directional spectra of narrow band signals received by an array of sensors.

5.4.1 Adaptive harmonic retrieval by method of exact linear search

Referring to Figure 2.8, the augmented cost function for the quadratically constrained adaptive filter is given by,

$$J(W,\gamma) = W^t R W - \gamma(W^t W - 1) \tag{5.7}$$

where,

$R = E(XX^t)$ L by L data covariance matrix

X = L-vector of the tap voltages (data vector)

W = L-weight vector

γ = Lagrange multiplier

Adaptive minimization of (5.7) by the method of ELS involves the quadratically constrained adaptive algorithm developed in section 3.4.

In this case the computational requirement of algorithm 3.91 can be significantly reduced by assuming stationary data and noting that the eigenstructure of R is invariant to any scalar multiplication of R. As a result R can be estimated by,

$$R_K = \sum_{i=1}^{K} X_i X_i^t \tag{5.8}$$

Furthermore, using the shifting property of the data vector X_K and symmetrical structure of R_K, R_K can be updated with only L multiplications per algorithm iteration. Estimating R_K by (5.8) results in the increase of its norm as K increases but this has no effect on its eigenstructure. In addition, the scalar $m_K = W_K^t R W_K$ in algorithm 3.91 may be computed either as $m_K = W_K^t R_K W_K$ or $m_K = \sum y_K^2$, where y_K is the filter output.

In the following examples the weight vector is initialized to

$W_1^t=(1, 0, 0, 0 \ldots\ldots)$ and the size of the filter is chosen so that, $L > 2n + 1$, where n is the number of the distinct sinusoids. The algorithm is run for a finite number of iterations (eg. 100, 500) and the AR-spectrum is computed as,

$$S_{AR} = \frac{1}{|W_K^t D(f)|^2} \tag{5.9}$$

where,

$$D^t(f) = (1, e^{-j2\pi f}, e^{-j4\pi f}, \ldots\ldots e^{-jL2\pi f}) \tag{5.10}$$

Example 9

In this example the signal was a sinusoid of normalized frequency 0.2 in additive Gaussian white noise. The signal to noise ratio, SNR, was 10dB and the filter size L = 3. Figure 5.38 shows the spectral estimates for K = 100 and K = 500. The broken and the solid curves represent the spectral estimates for K = 100 and K = 500 respectively.

Example 10

The same as example 9 but the signal contained two sinusoids of equal power and normalized frequencies of 0.18 and 0.38. The filter size and the SNR were L = 5, 10dB respectively. Figure 5.39 shows the spectral estimates for K = 100 and 500 (broken and solid curves).

Example 11

The same as example 10 but SNR was 0dB. Figure 5.40 shows the spectral estimates for K = 100 and 500 (broken and solid curves).

These results clearly verify the rapid initial speed of convergence of the algorithm both at high and low SNR conditions. Further simulations revealed that the algorithm was able to peak up the spectral peaks only after 20 to 50 iterations under various SNR conditions, while a good spectral precision was achieved about 100 iterations.

5.4.2 Adaptive directional spectral estimation by method of exact linear search

Figure 2.9 shows the constrained narrow band array system. The augmented cost function for this system is given by,

$$J(W,\gamma) = W^t R W^+ - \gamma\,(W^t W^+ - 1) \tag{5.11}$$

where,

$R = E(X^+X^t)$ L by L data covariance matrix (Hermitian)

X = L-vector of the tap voltages (data vector)

γ = Lagrange multiplier

$.^+$ = complex conjugate

Application of the method of ELS to this minimization problem results in a complex version of the algorithm 3.91 which is outlined below,

$$g_K = m_K W_K - R W_K$$

$$a_K = g_K^t g_K^+ / g_K^t R g_K^+$$

$$\tilde{W}_{K+1} = W_K + a_K g_K \qquad (5.12)$$

$$W_{K+1} = \tilde{W}_{K+1} / |\tilde{W}_{K+1}^t \tilde{W}_{K+1}^+|^{\frac{1}{2}}$$

where, $m_K = W_K^t R W_K^+$. Assuming stationary data and noting that the eigenstructure of R is invariant to any scalar multiplication of R, R can be estimated by

$$R_K = \sum_{i=1}^{K} X_i^+ X_i^t \qquad (5.13)$$

The growing norm of R_K has no effect on its eigenstructure. Also, the scalar m_K may be computed either as $m_K = W_K^t R_K W_K^+$ or $m_K = \sum y_K y_K^+$, where y_K is the filter output. Note that R is Hermitian and positive definite, thus all its eigenvalues are real.

A detailed theoretical analysis of the eigenfilter spectral analysis is presented in [65]. However, to appreciate the simulation examples presented in this section, a brief explanation will be useful.

Assuming n narrow band incident signals, then the covariance matrix has n eigenvalues corresponding to n sources. Also, there exists L-n eigenvalues corresponding to the power of the additive noise (background and sensors noise). It can be shown that L-n eigenvectors corresponding to the minimum eigenvalue are orthogonal to all source direction vectors. As a result one can compute the eigenvector of R corresponding to its minimum eigenvalue. The zeros of the function

$$S(\theta) = D^t(\theta)W^* \tag{5.14}$$

give the source directions. Where W^* is the eigenvector corresponding to the minimum eigenvalue and

$$D^t(\theta) = (1, e^{\frac{j\pi d}{l}\sin\theta}, \ldots\ldots e^{\frac{j2\pi(L-1)d}{l}\sin\theta}) \tag{5.15}$$

is the direction vector which describes the phase shift at each sensor due to an incident signal. d = sensor spacing in wavelength, θ is the bearing of the narrow band incident signal and l is the wavelength.

A better indication of source directions is given by the eigenvector spectrum [65],

$$S(\theta) = \frac{1}{|D^t(\theta)W^*|^2} \tag{5.16}$$

This spectrum has strong peaks in the directions of sources due to orthogonality of $D(\theta)$ and W^* at θ = bearing of incident signals.

In the following examples the weight vector is initialized to $W_1^t=(1, 0, 0, \dots)$ and the size of the filter is chosen greater than the number of incident signals. The algorithm is run for a finite number of iterations (eg. 150, 200) and the directional spectra (DS) is computed by (5.16).

Example 12

In this example the incident field contained three narrow band signals , in additive Gaussian white noise, with bearing 10, 45, and 60 degrees at SNR = 15, 20, 15 dB respectively. The signals incident at 45 and 60 degrees were correlated with a mutual correlation coefficient 0.72. The noise per sensor was 0dB (all the noise terms were statistically independent). The narrow band signals were generated by passing three Gaussian white noise through three band pass filters.

Figure 5.41 shows the DS-estimates for K = 150 and 200 (broken and solid curves respectively).

Example 13

The same as example 12 but in order to examine the effect of a low SNR condition, the signal to noise ratios were set to 10, 15, 10 dB respectively.

Figure 5.42 shows the DS-estimates for K = 150 and 200 (broken and solid curves respectively). These results demonstrate the ability of the algorithm to estimate the spatial spectra, offering a good precision after a short number of algorithm iterations.

It must be noted that the peaks heights bear no relationship to signal power. Estimates of the source powers can be obtained (if required) by a least squares fit to the array data, based on the apriori knowledge of the source directions [66].

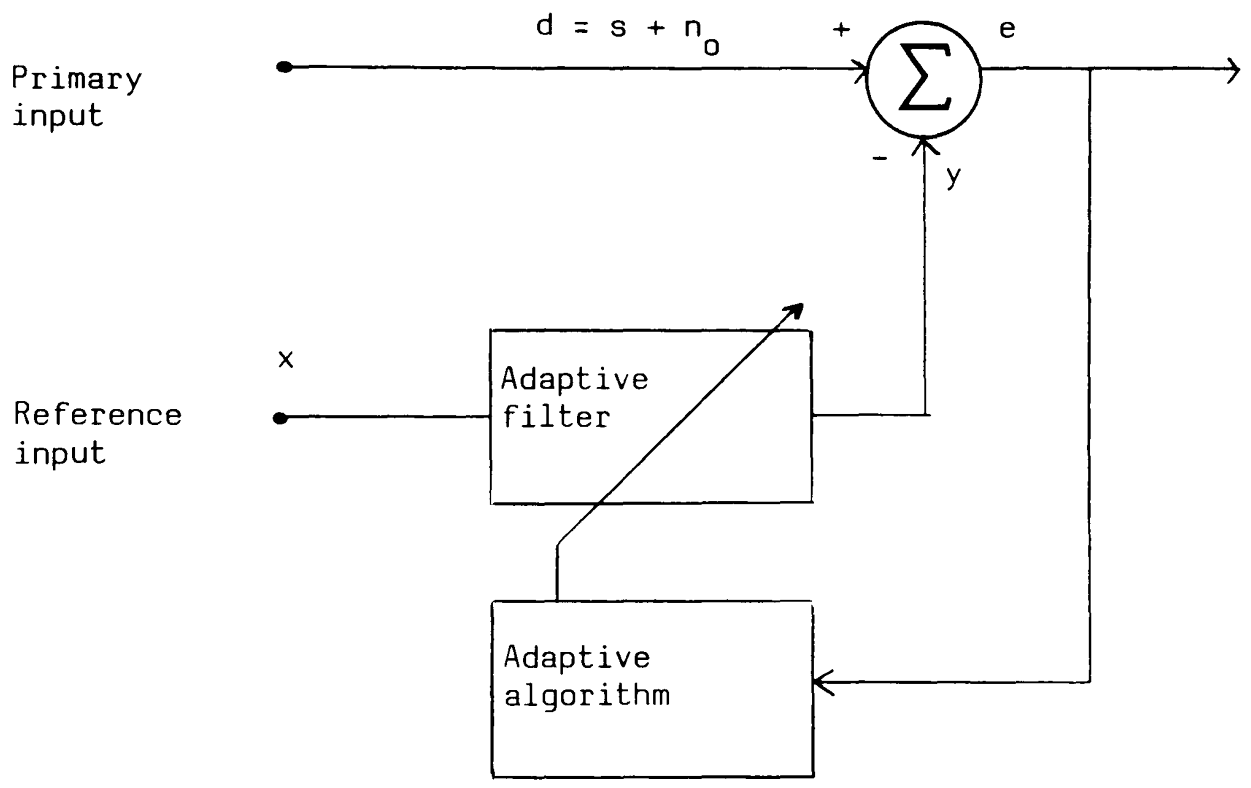


Figure 5.1 : Adaptive noise cancelling technique.

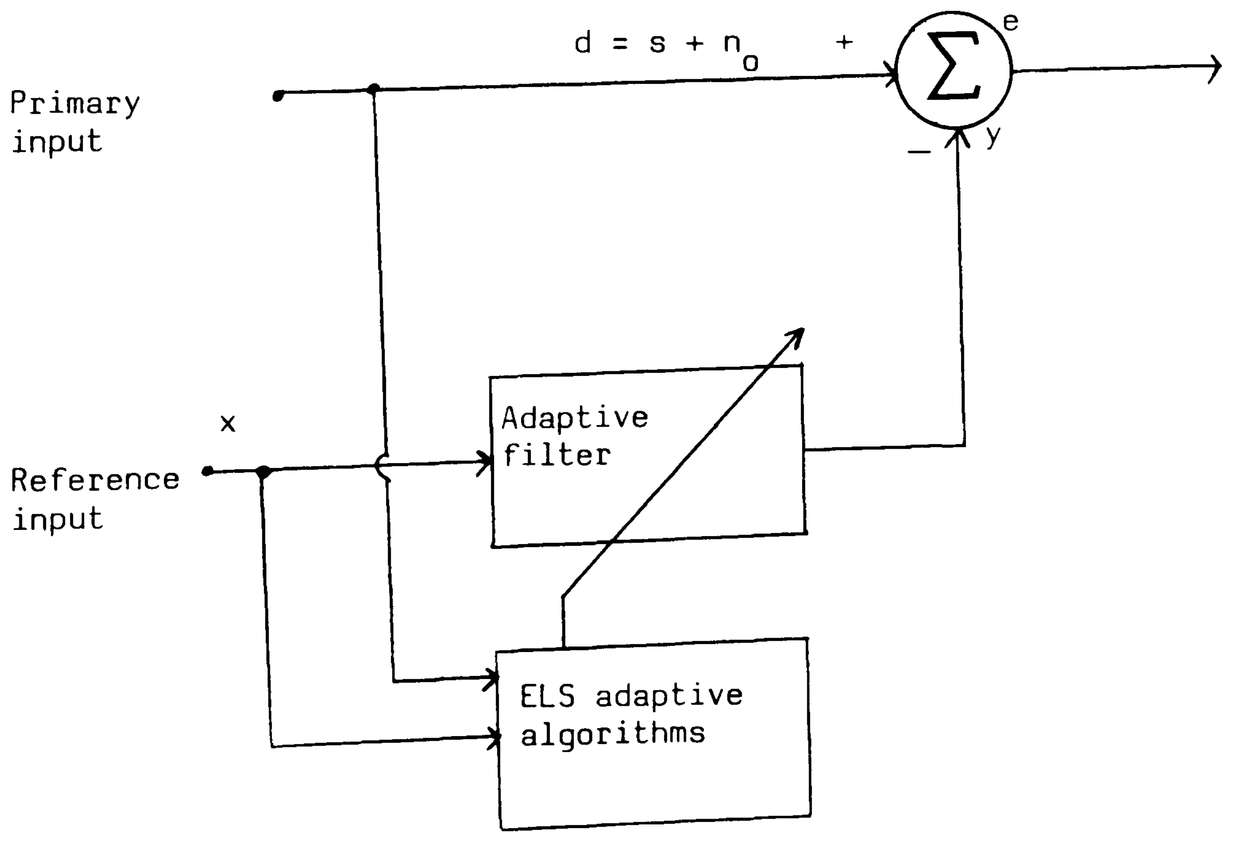


Figure 5.2 : Adaptive noise cancelling using ELS algorithms (3.14, 3.16, 3.18).

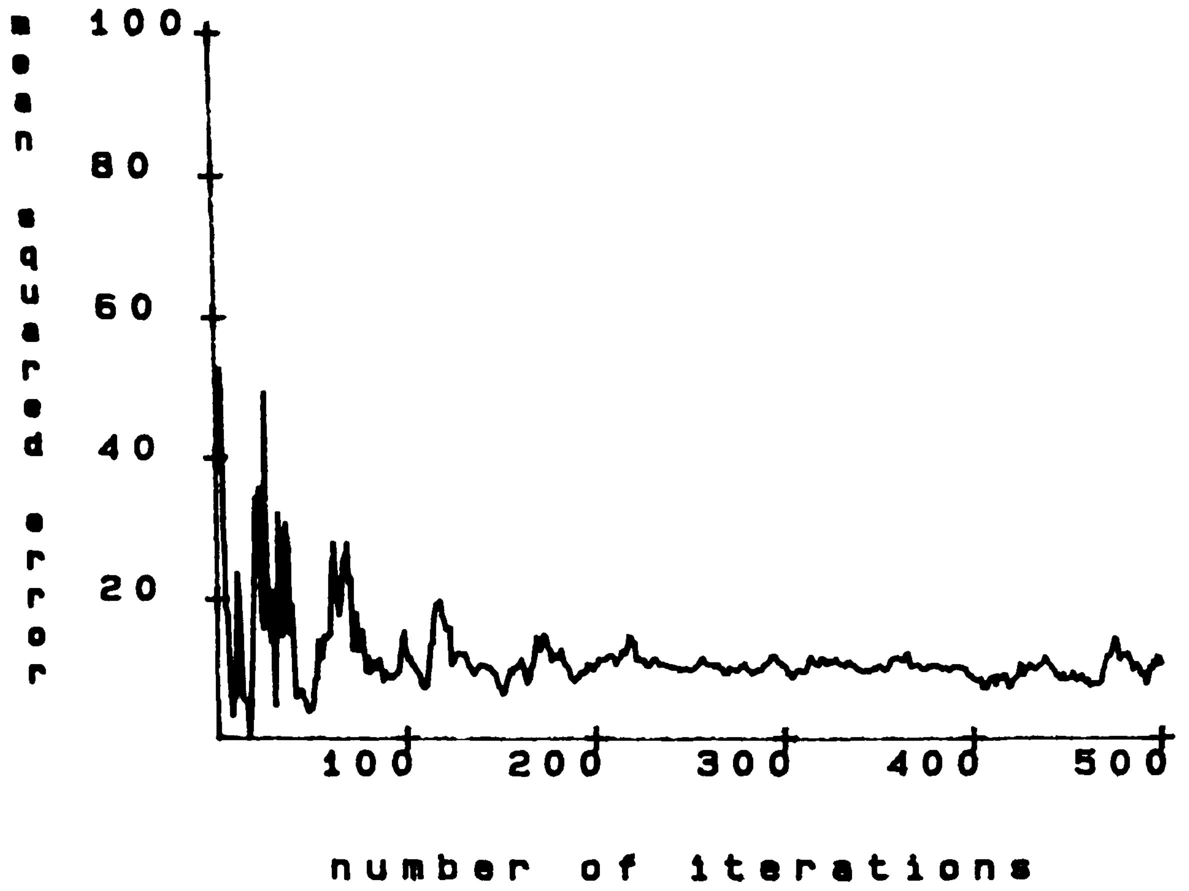


Figure 5.3 : Learning curves, averaged over 20 individual runs. ELS algorithm. EVR = 20, L = 10.

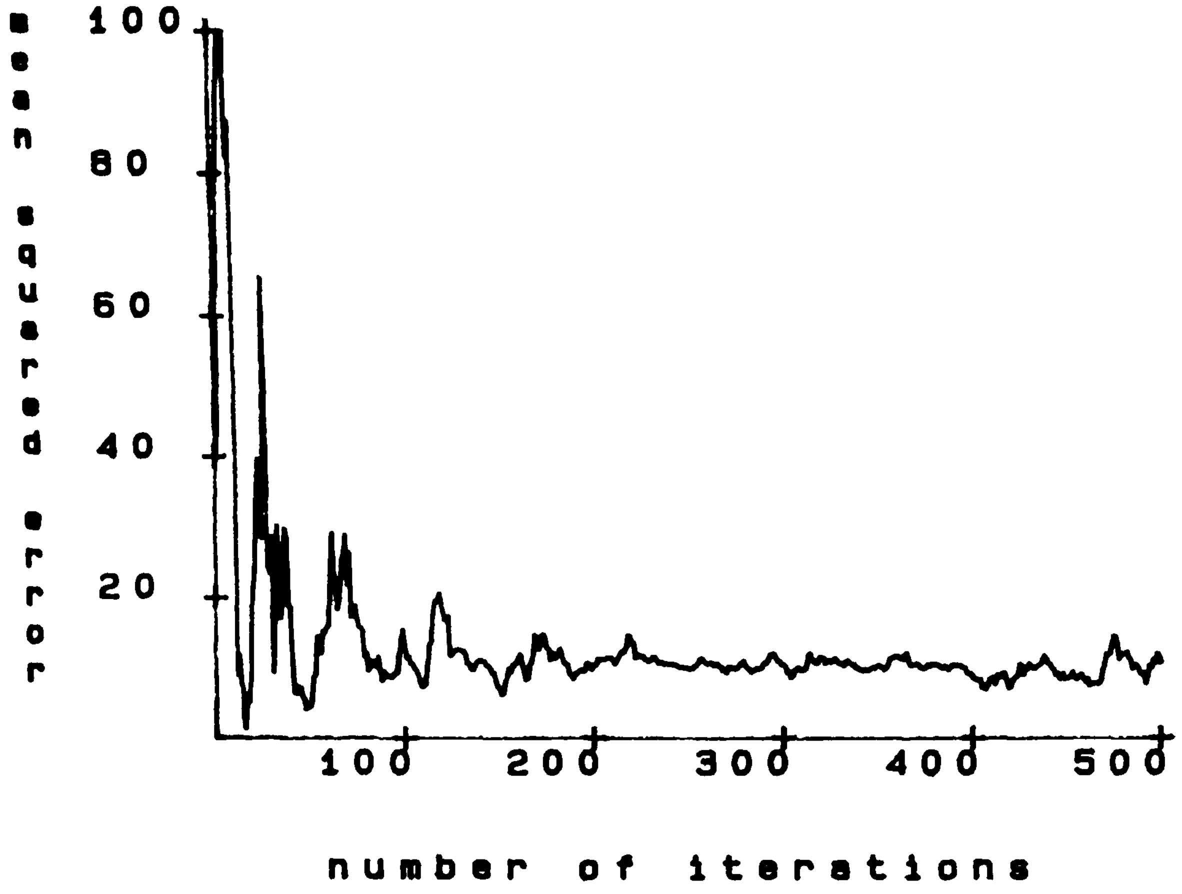


Figure 5.4 : Learning curves, averaged over 20
individual runs. RLS algorithm.
EVR = 20, L = 10.

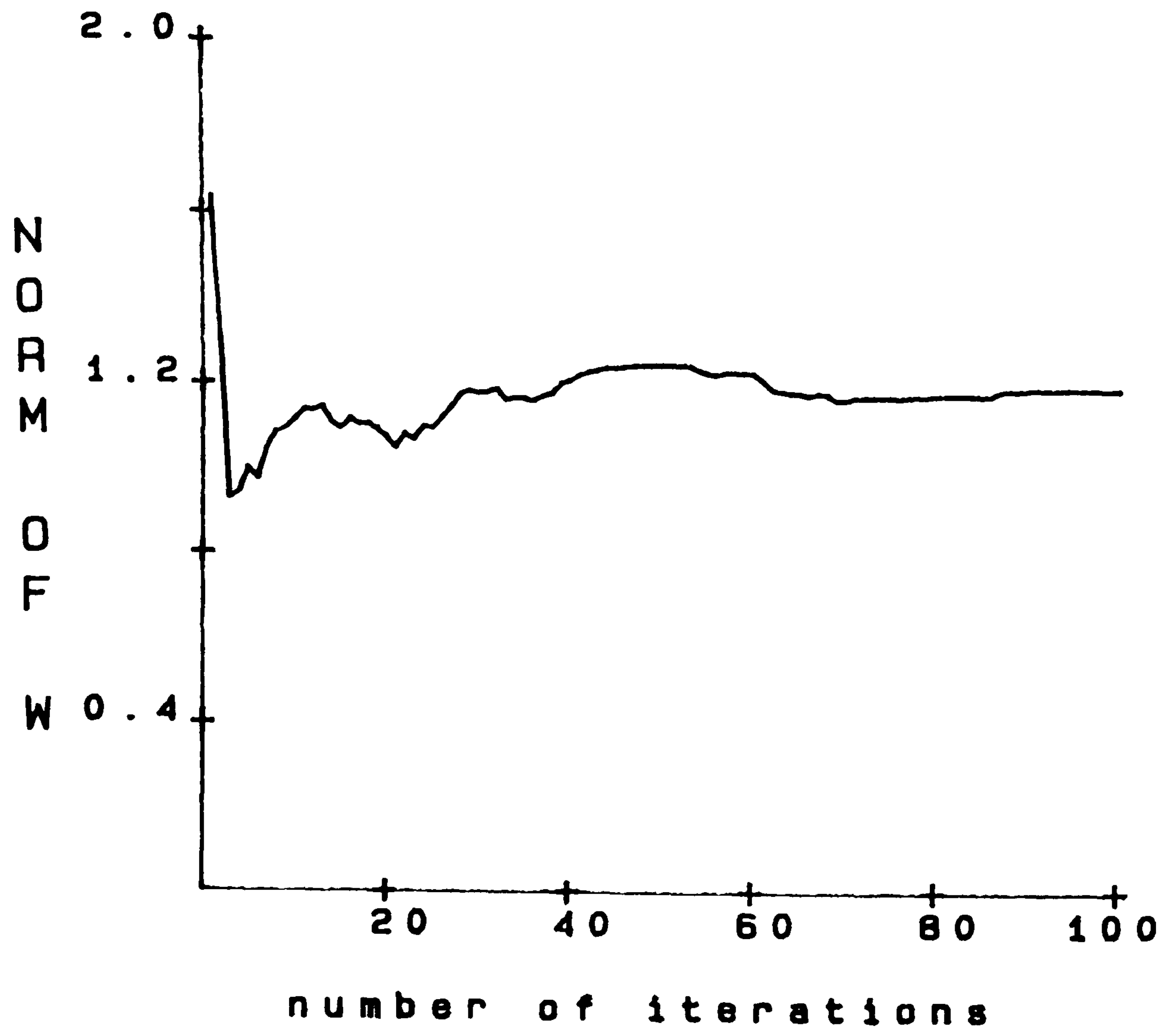


Figure 5.5 : Convergence of $|W_K|$. ELS algorithm.
EVR = 20, L = 10. $|W^*| = 1.2$.

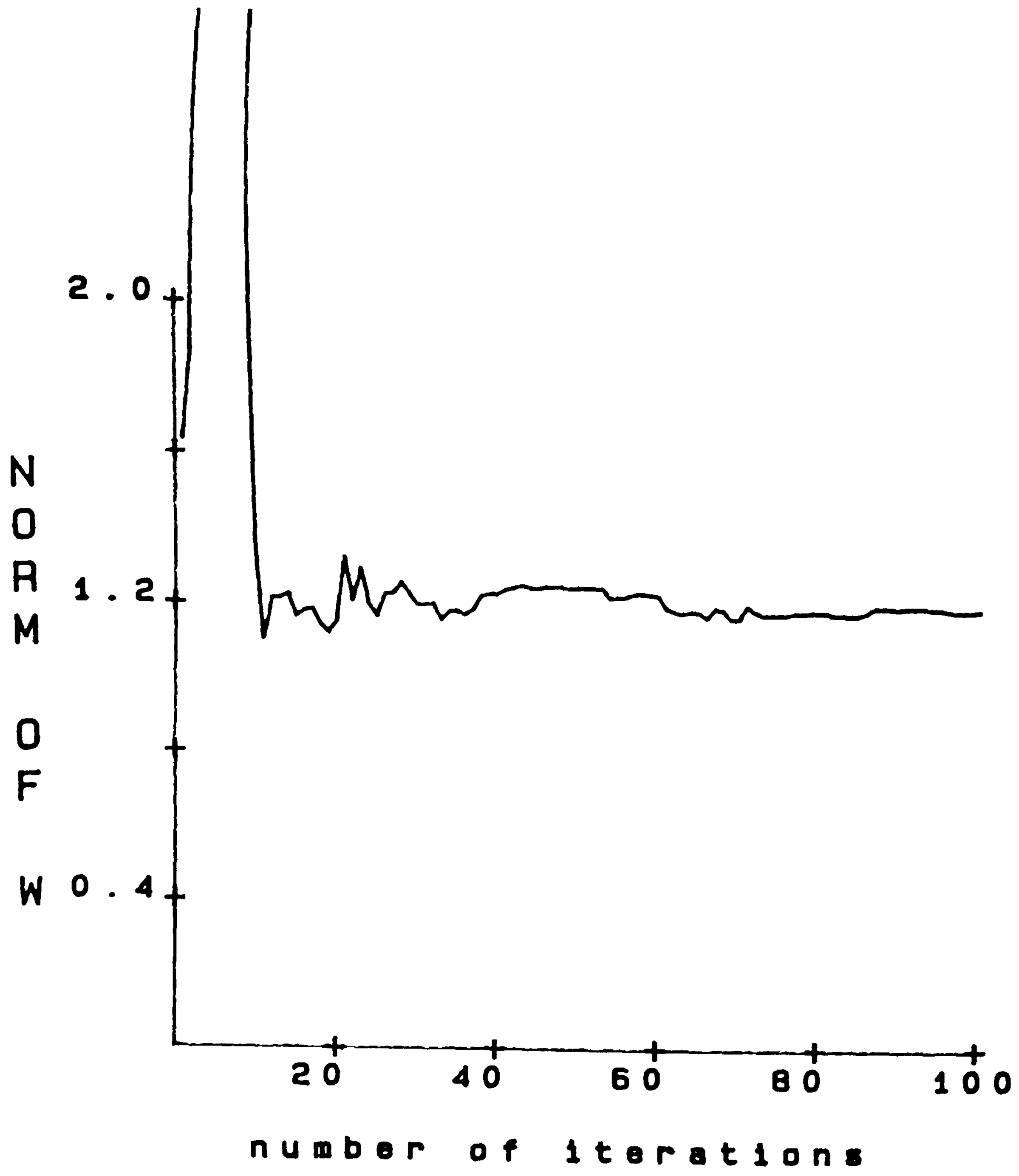


Figure 5.6 : Convergence of $|W_K|$. RLS algorithm. EVR = 20, L = 10. $|W^*| = 1.2$.

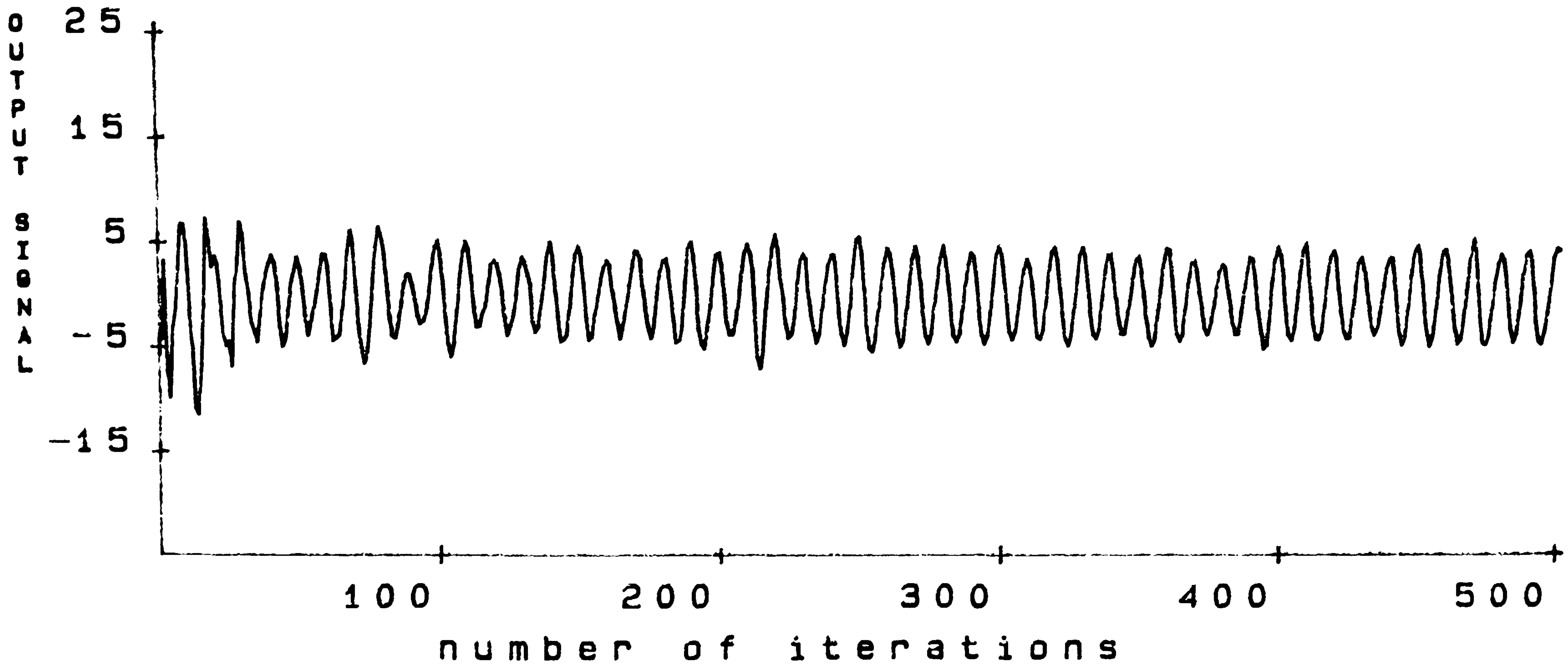


Figure 5.7 : Noise canceller output. ELS algorithm, L = 10, EVR = 20.

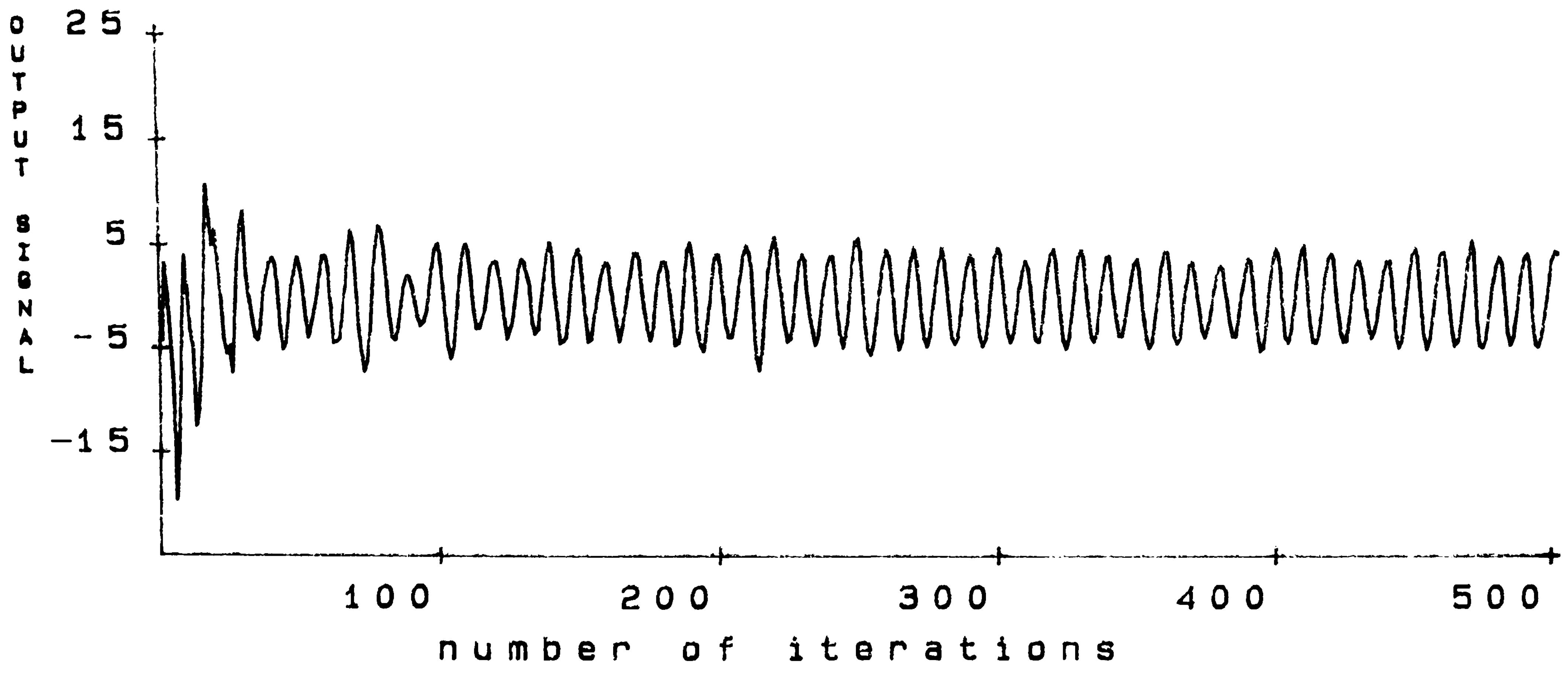


Figure 5.8 : Noise canceller output. RLS algorithm, L = 10, EVR = 20.

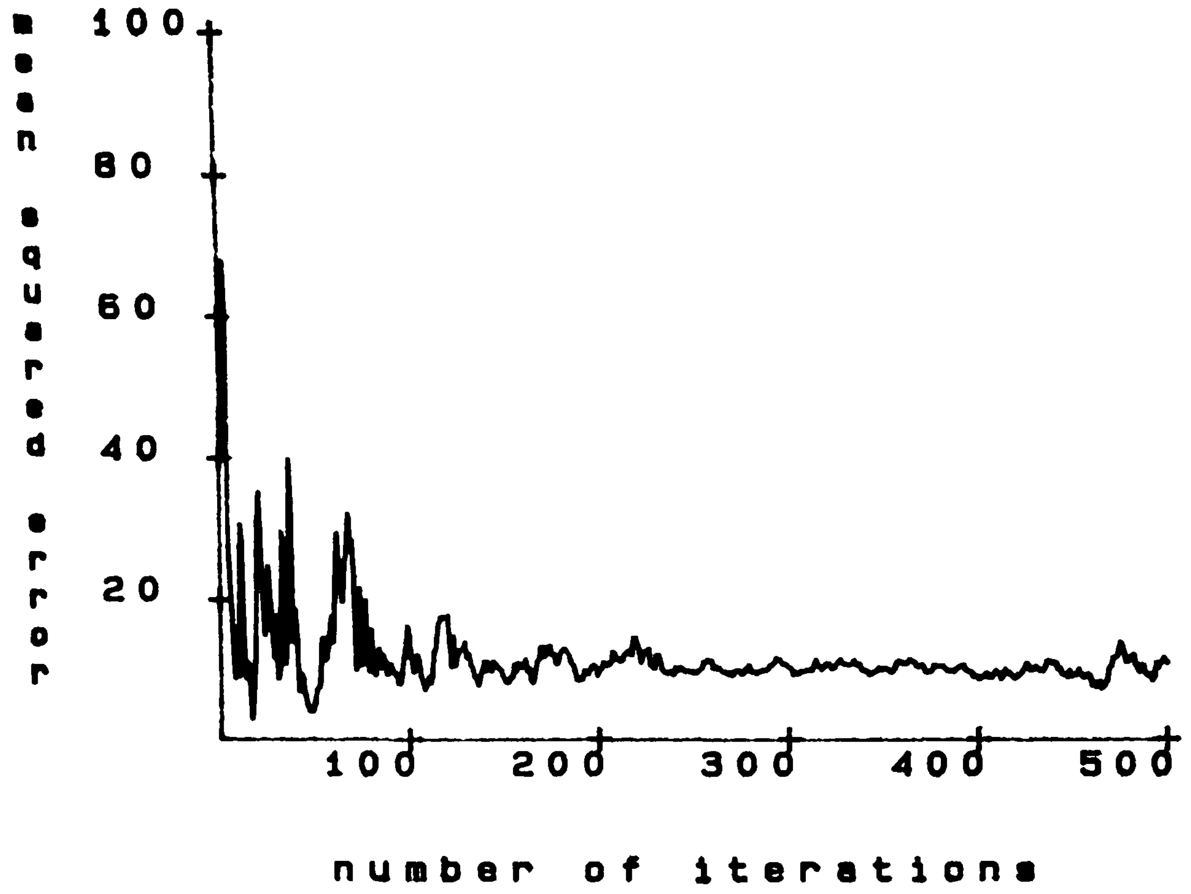


Figure 5.9 : Learning curves, averaged over 20 individual runs. ELS algorithm. EVR = 402, L = 10.

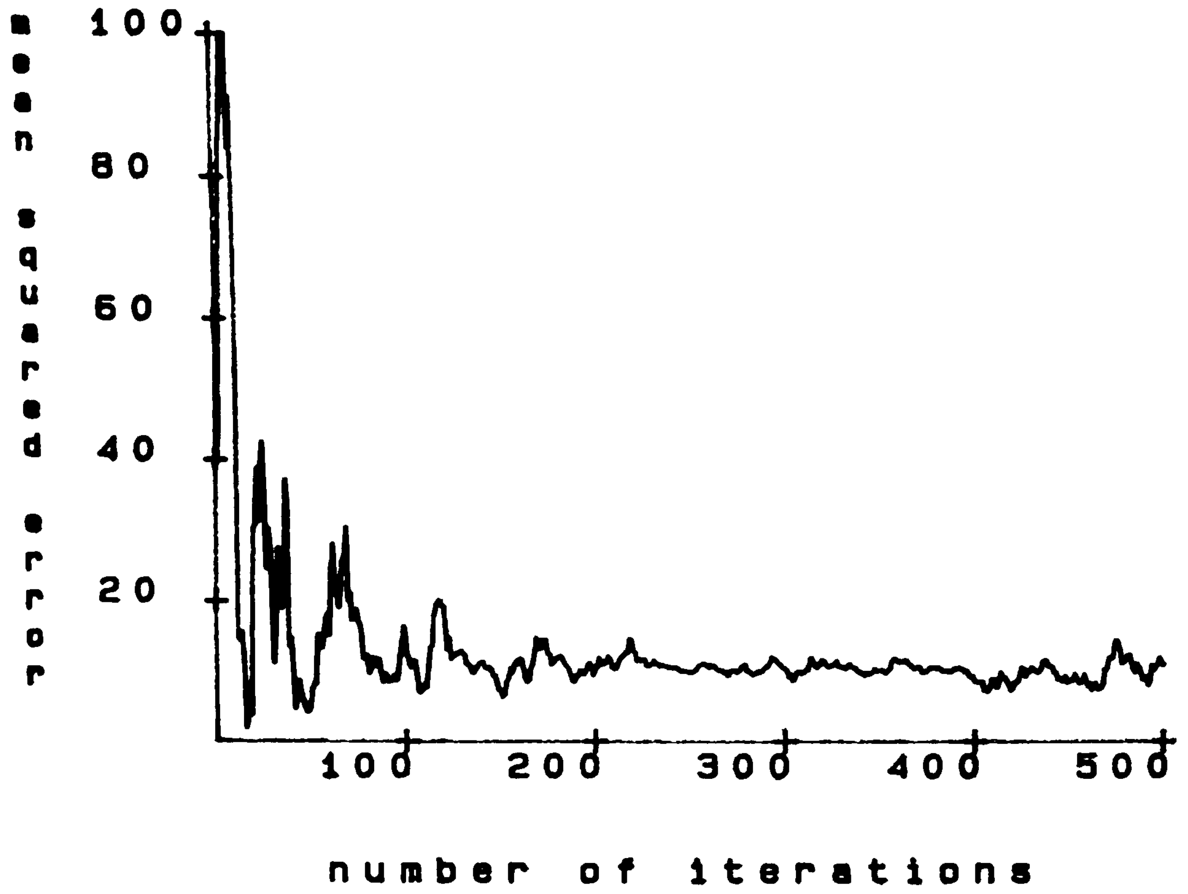


Figure 5.10 : Learning curves, averaged over 20 individual runs. RLS algorithm. EVR = 402, L = 10.

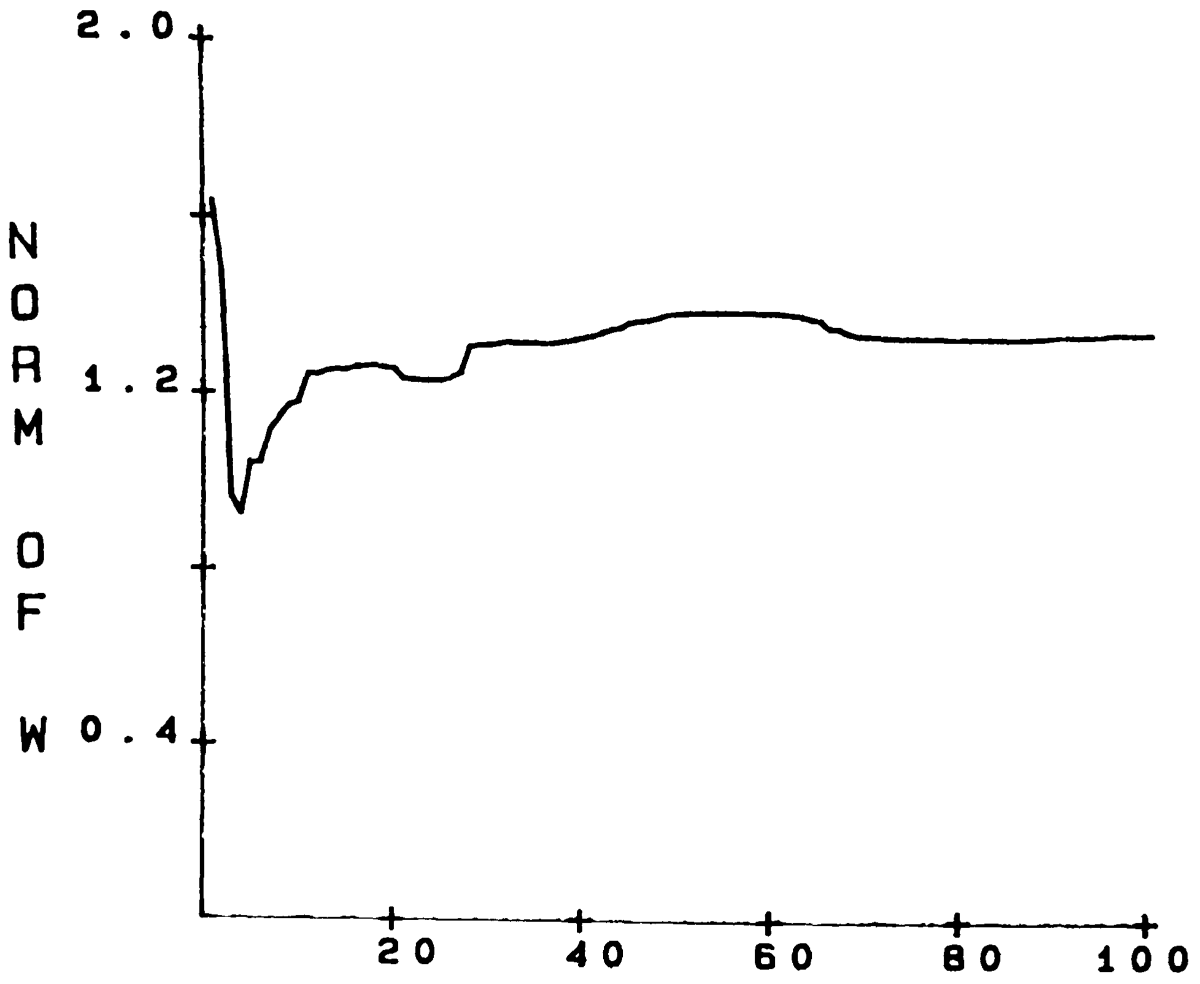


Figure 5.11 : Convergence of $|W_K|$. ELS algorithm.
EVR = 402, L = 10. $|W^*| = 1.38$.

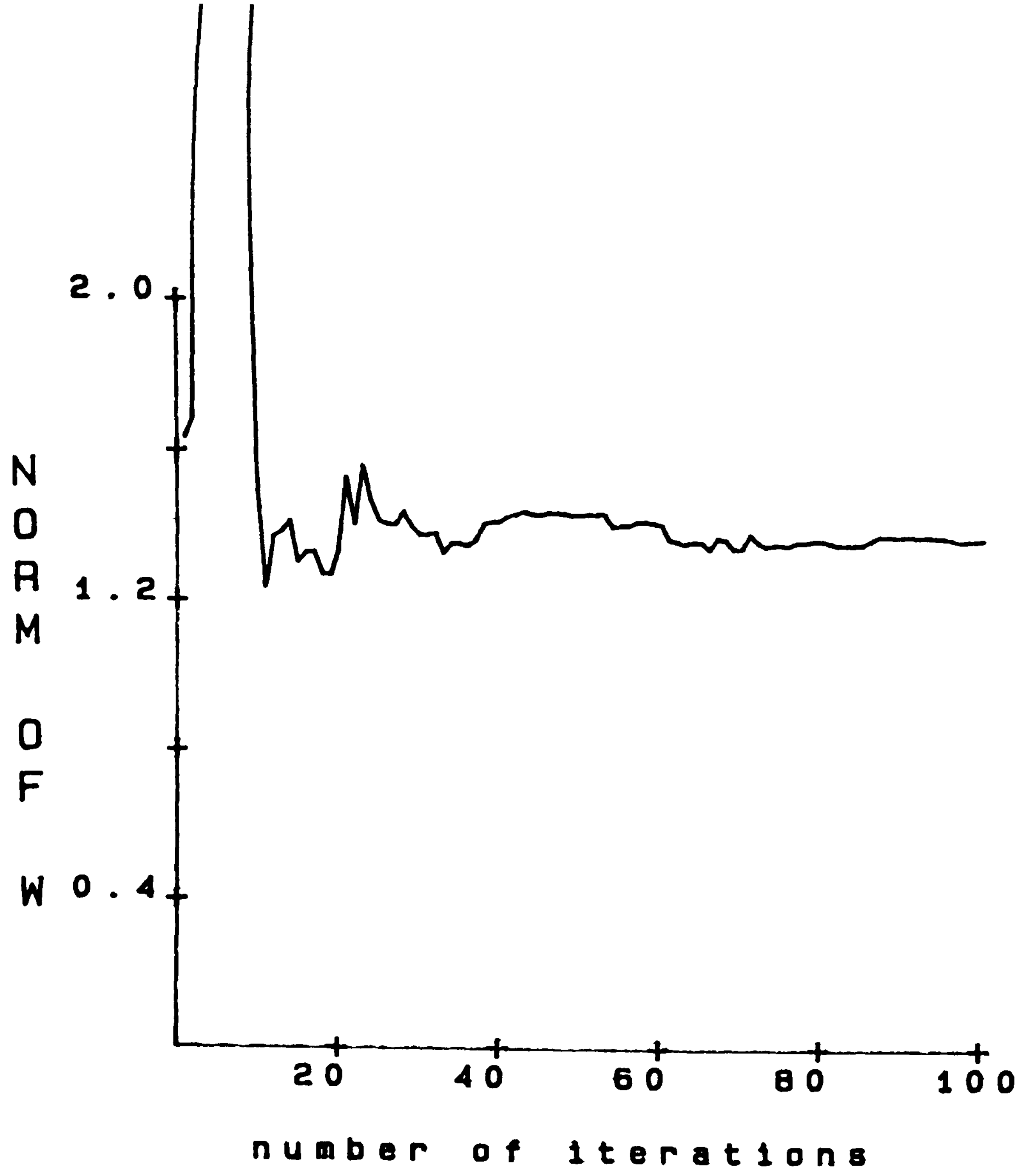


Figure 5.12 : Convergence of $|W_K|$. RLS algorithm. EVR = 402, L = 10. $|W^*| = 1.38$.

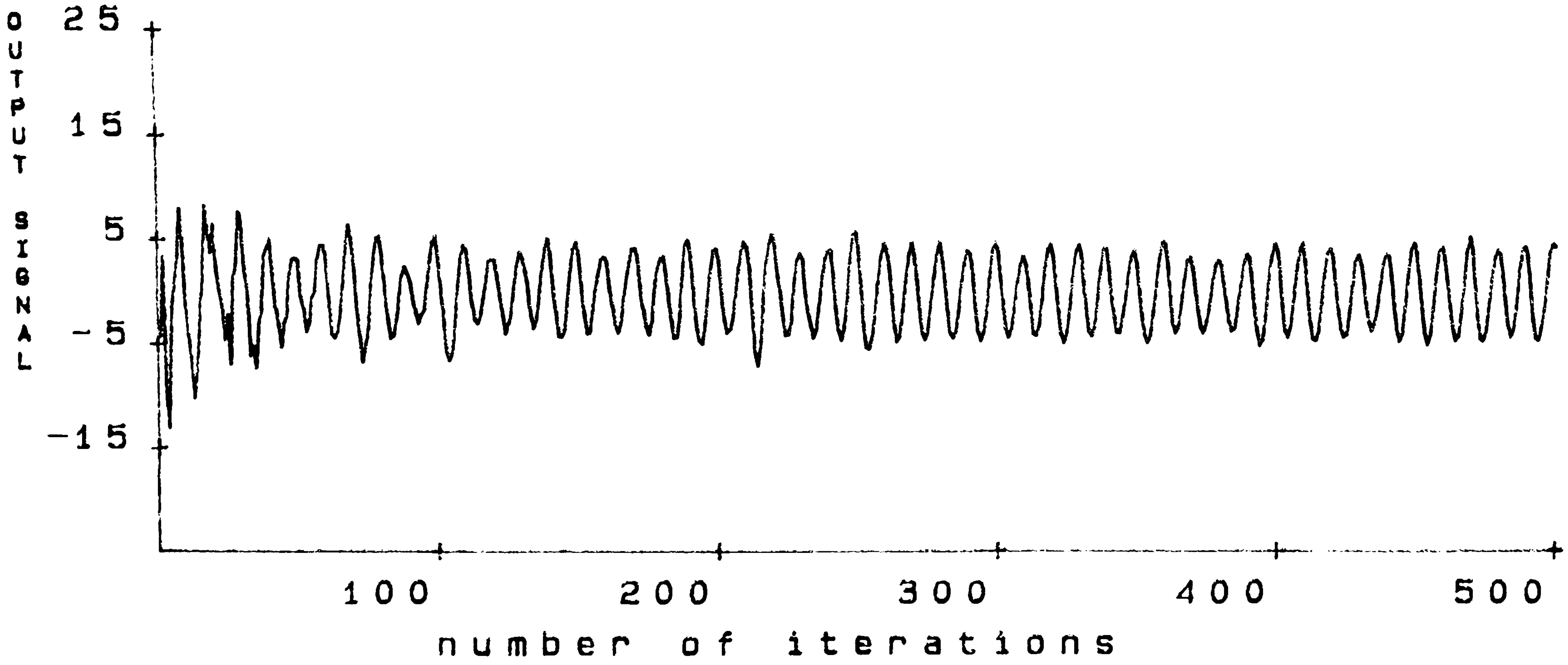


Figure 5.13 : Noise canceller output. ELS algorithm, L = 10, EVR = 402.

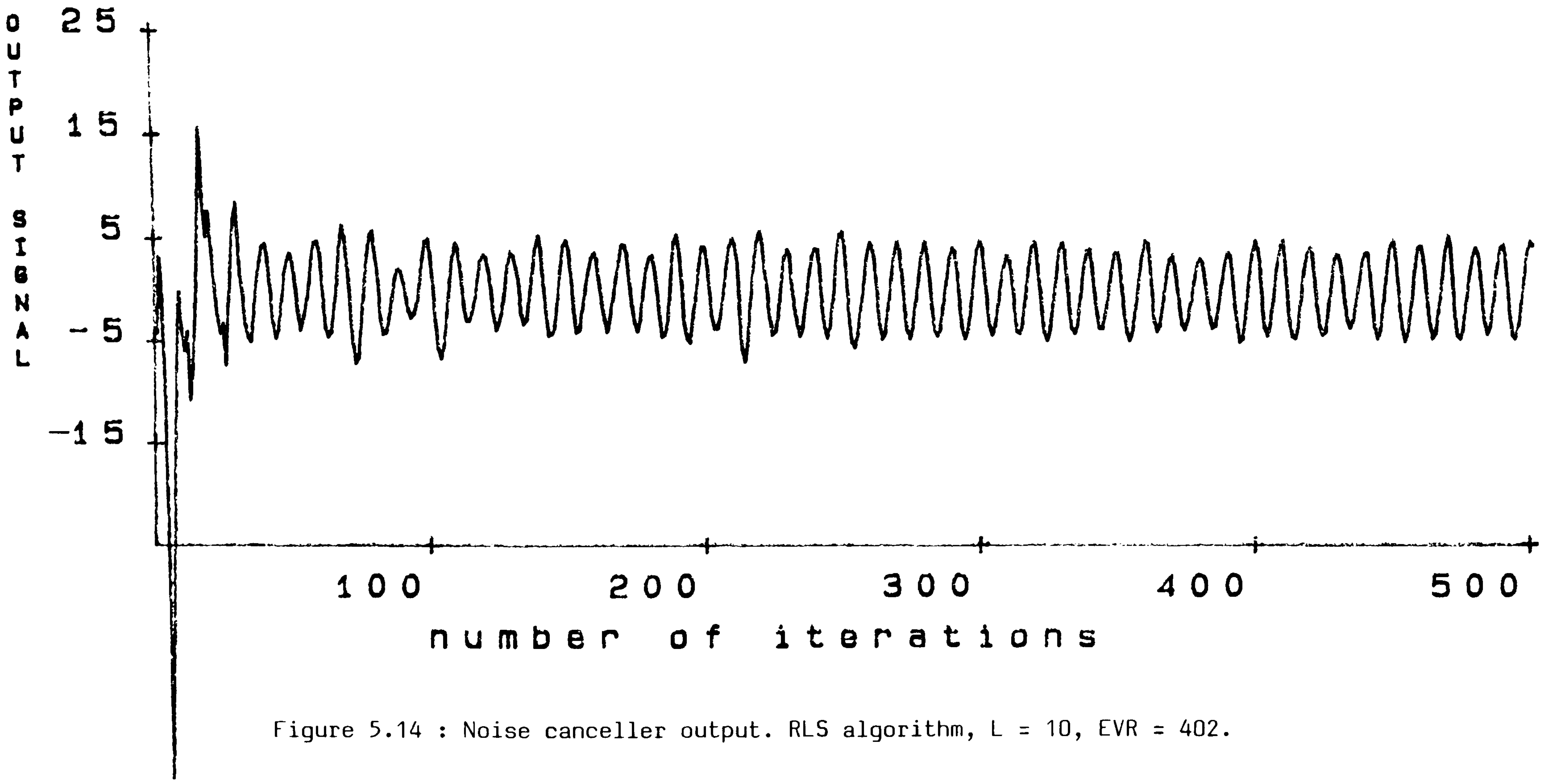


Figure 5.14 : Noise canceller output. RLS algorithm, L = 10, EVR = 402.

	$K > 1$	$K > 200$	$K > 350$
α	0.99	0.99	0.99
SIGNAL POWER	2.0	40.0	10.0
SIGNAL FREQ.	0.2	0.2	0.2
NOISE POWER	10.0	20.0	10.0
u_1 , u_2	0.9, 0.05	0.6, 0.35	0.45,0.025
EVR	402	207	8.0
$\|W^*\|$	1.1	1.28	1.36

Table 5.a : Values of the various parameters over 500 iterations

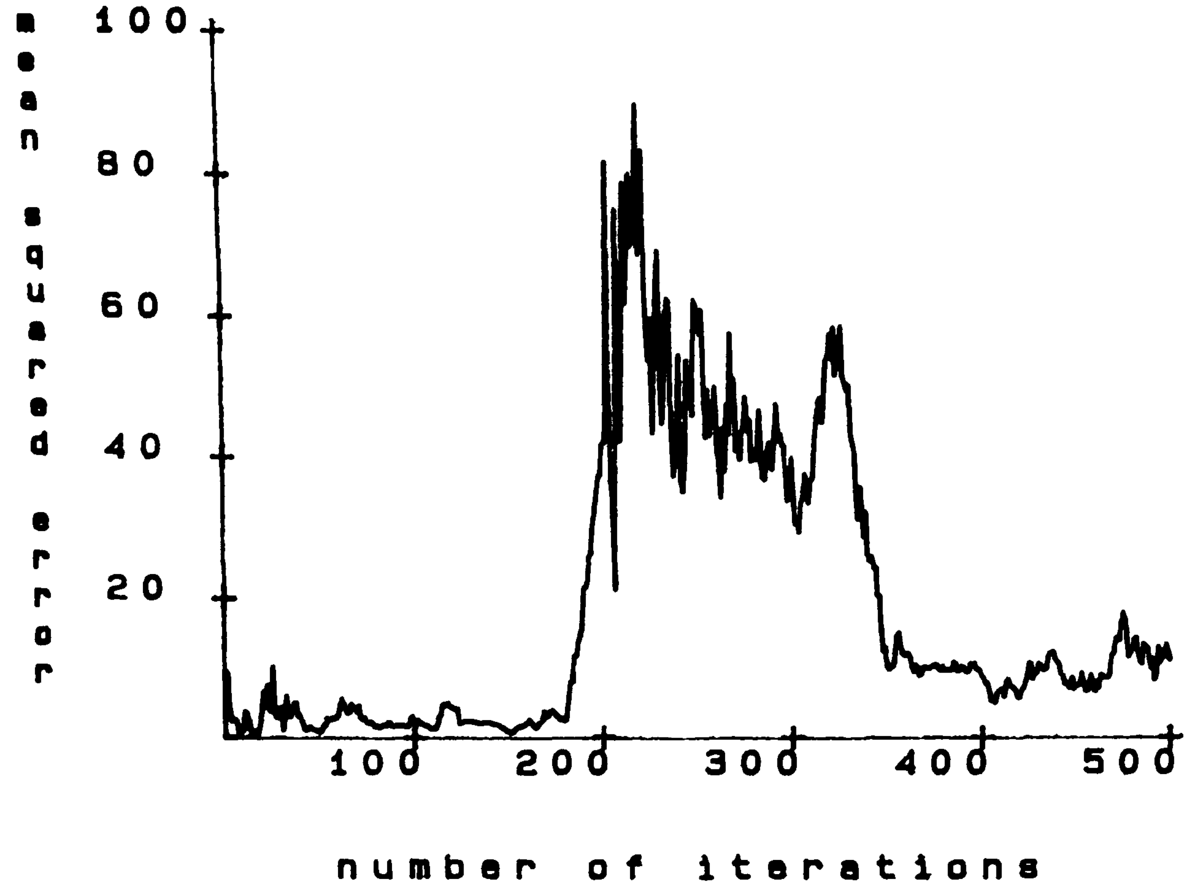


Figure 5.15 : Learning curves averaged over 20 individual runs. ELS algorithm.

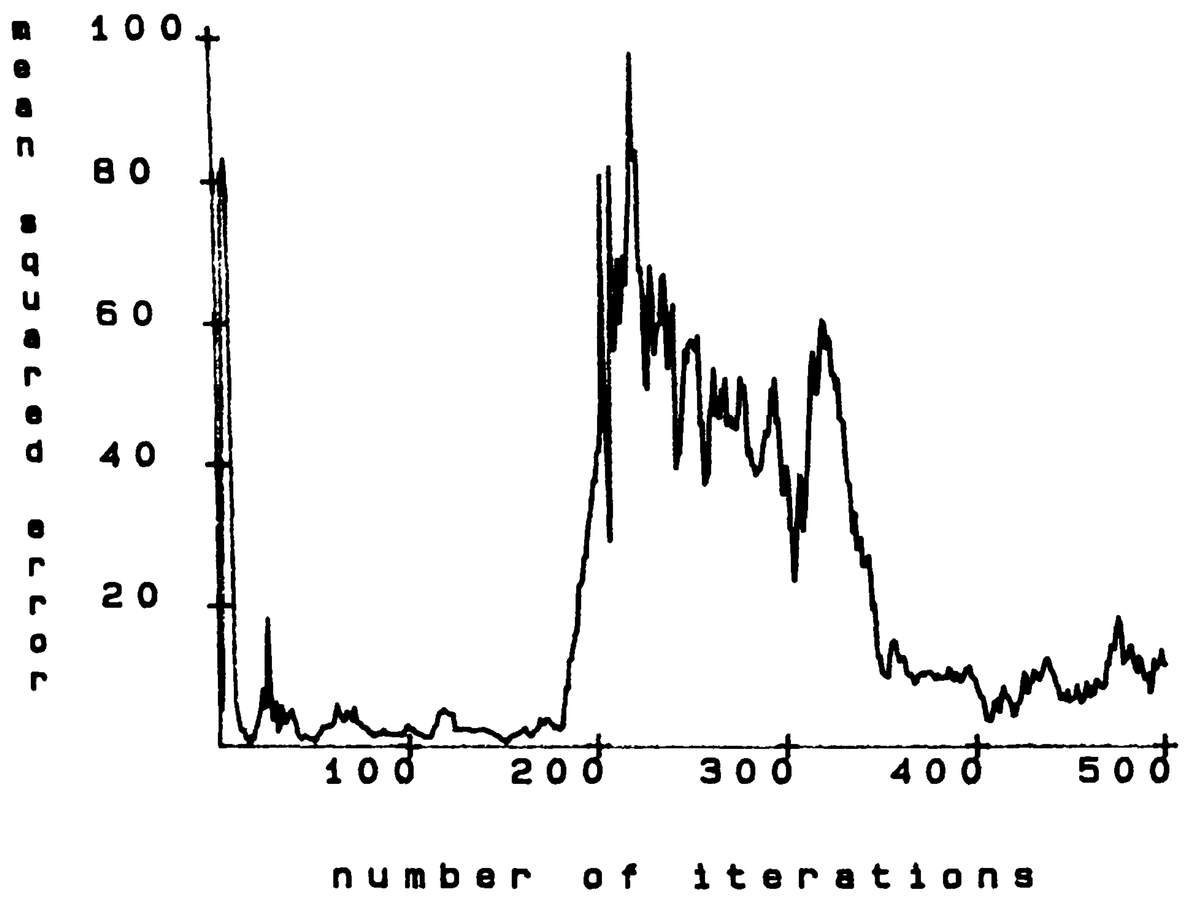


Figure 5.16 : Learning curves averaged over 20 individual runs. RLS algorithm.

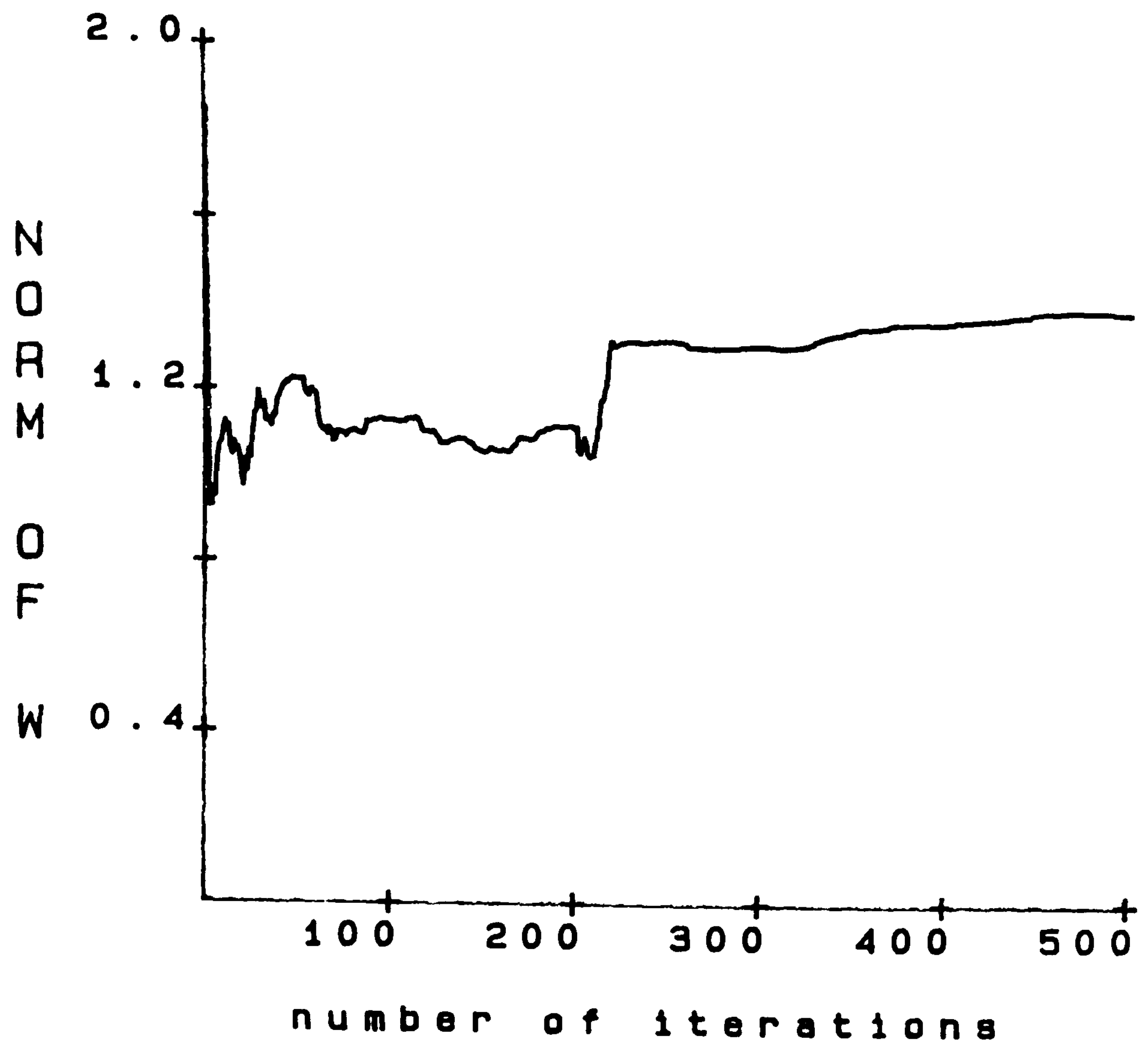


Figure 5.17 : Convergence of $|W_K|$. ELS algorithm.

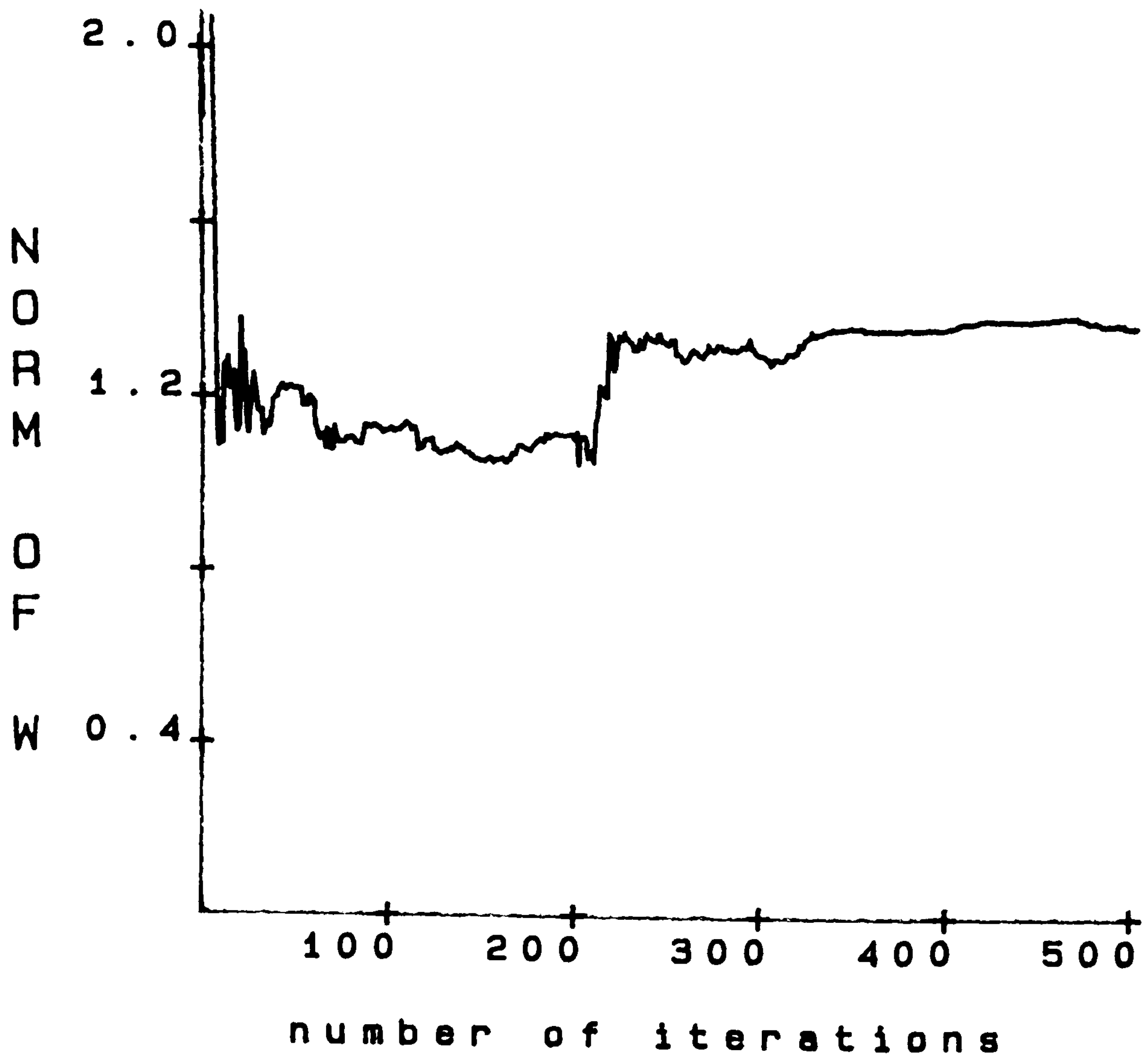


Figure 5.18 : Convergence of $|W_K|$. RLS algorithm.

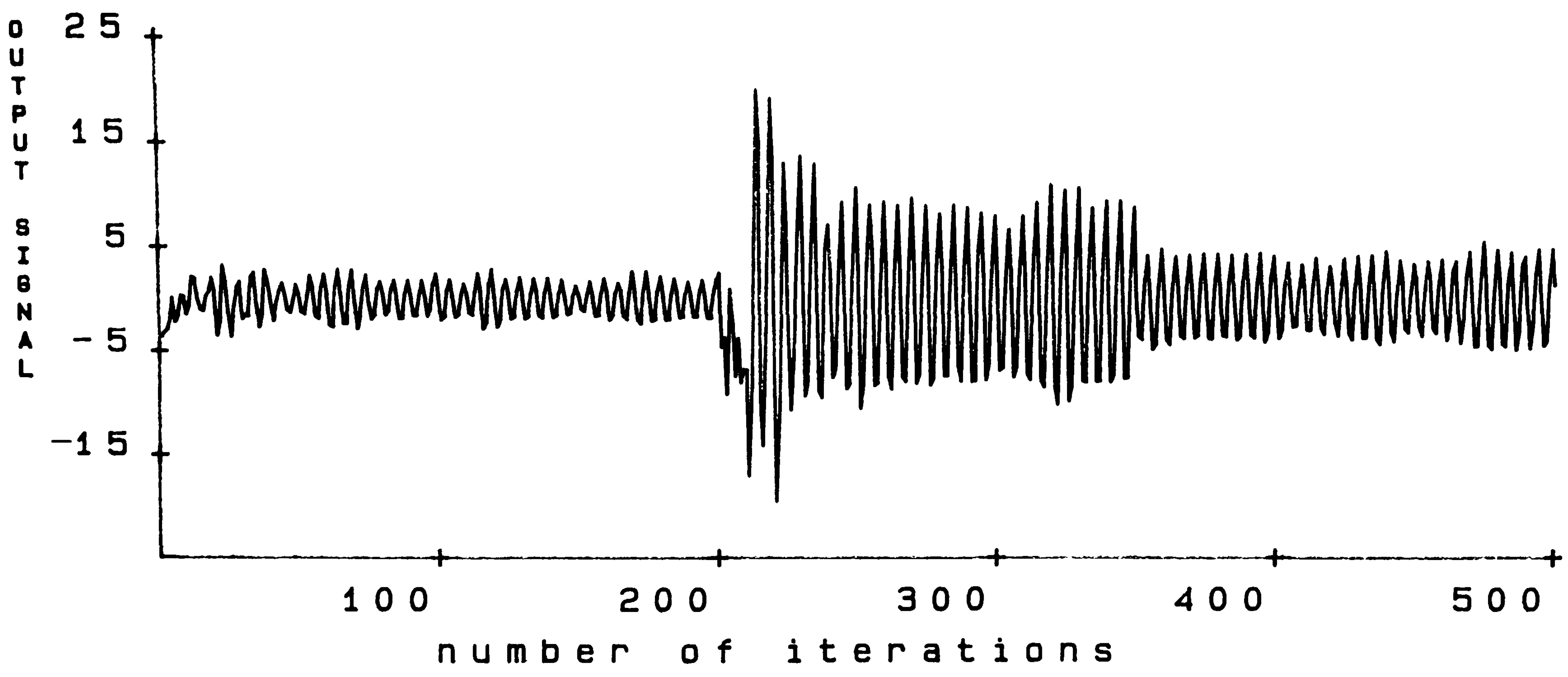


Figure 5.19 : Noise canceller output. ELS algorithm, L = 10.

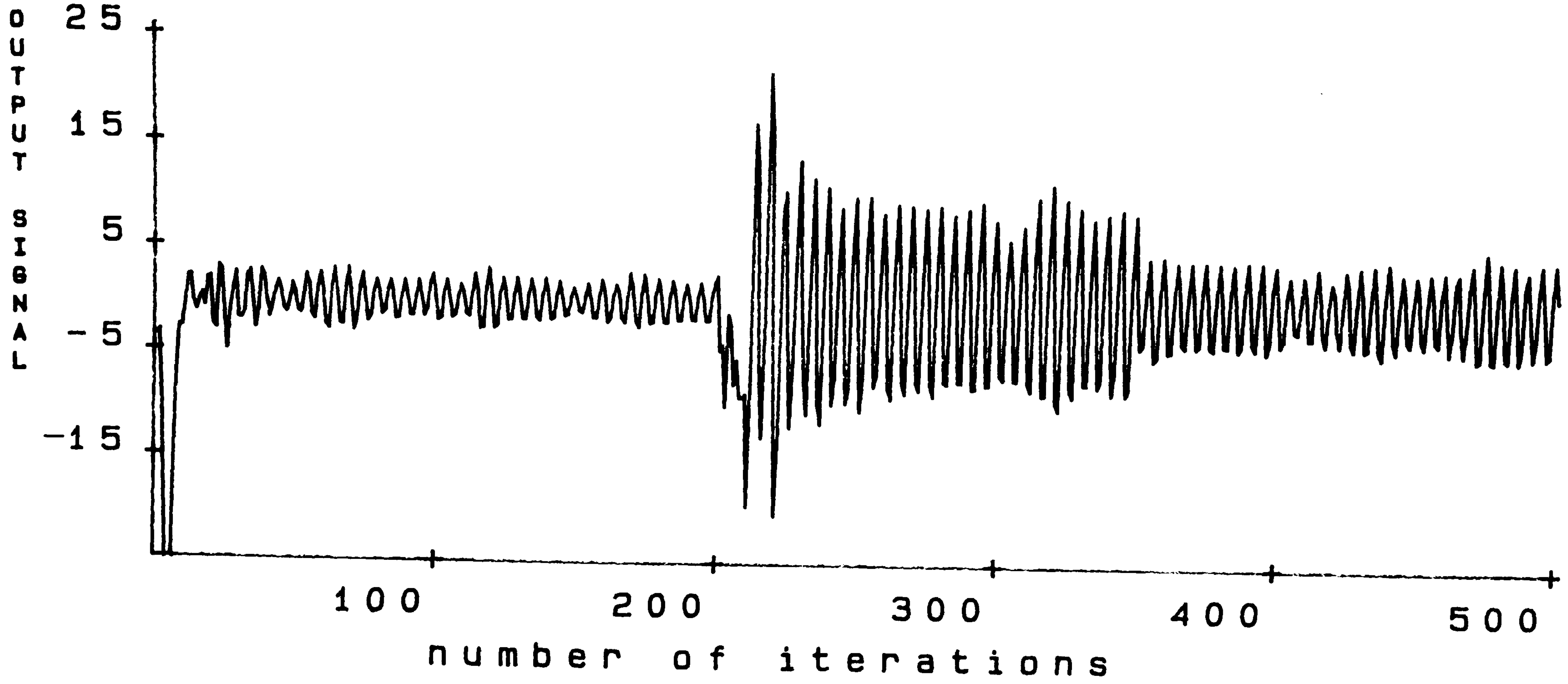


Figure 5.20 : Noise canceller output. RLS algorithm, L = 10.

	$K > 1$	$K > 200$	$K > 300$	$K > 400$
α	0.99	0.99	0.99	0.99
SIGNAL POWER	2.0	20.0	60.0	20.0
SIGNAL FREQ.	0.2	0.1	0.05	0.3
NOISE POWER	10.0	10.0	10.0	10.0
u_1 , u_2	0.6, 0.05	0.7, 0.05	0.8, 0.05	0.9, 0.05
EVR	20	37	87	402
$\|W^*\|$	1.2	1.24	1.28	1.25

Table 5.b : Values of the various parameters over 500 iterations

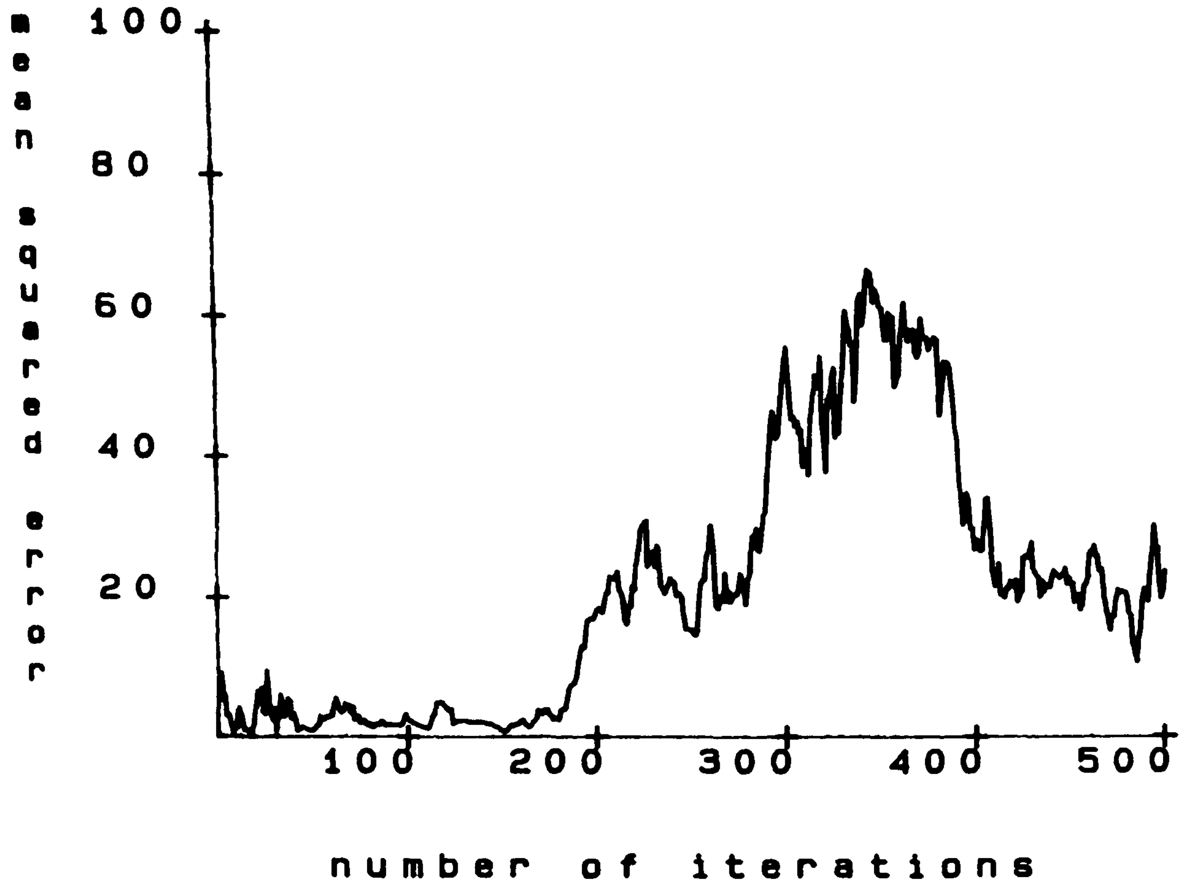


Figure 5.21 : Learning curve, averaged over 20 individual runs. ELS algorithm.

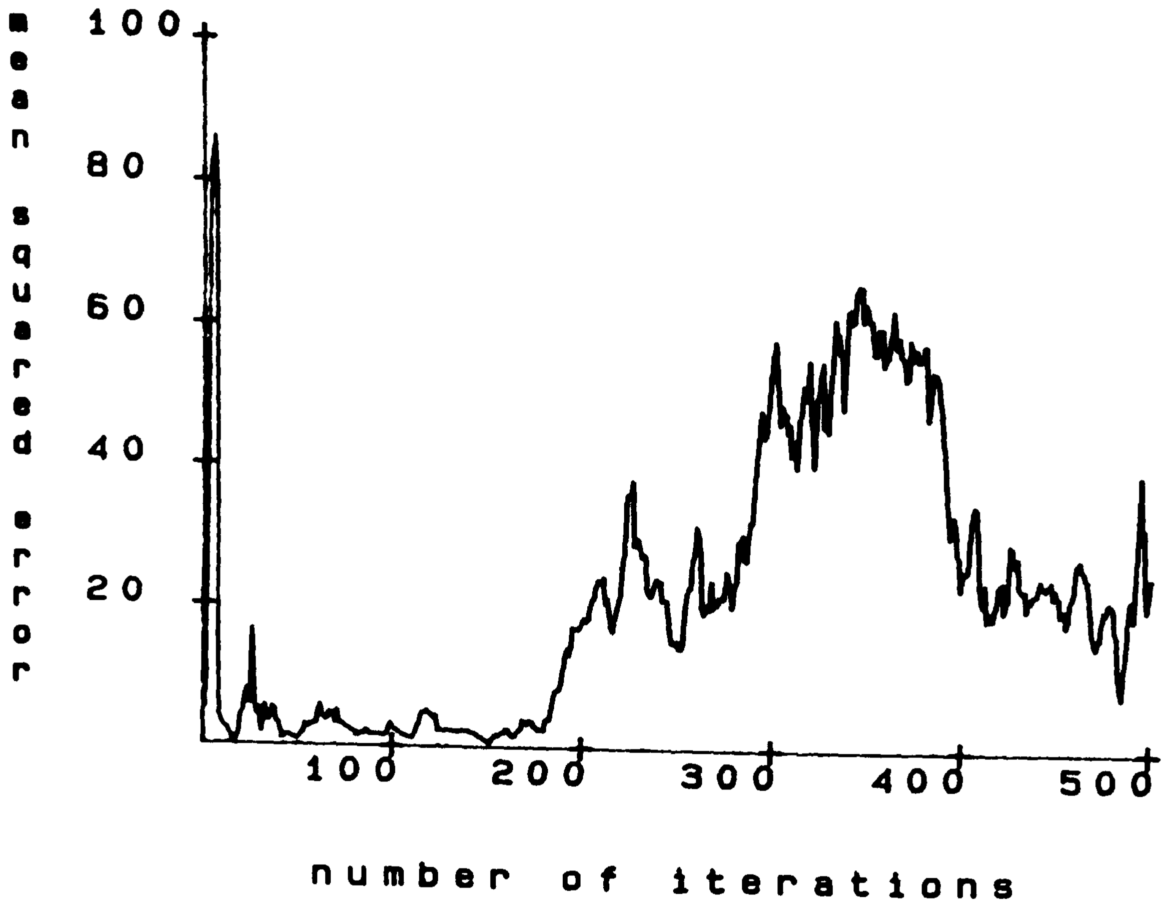


Figure 5.22 : Learning curve, averaged over 20 individual runs. RLS algorithm.

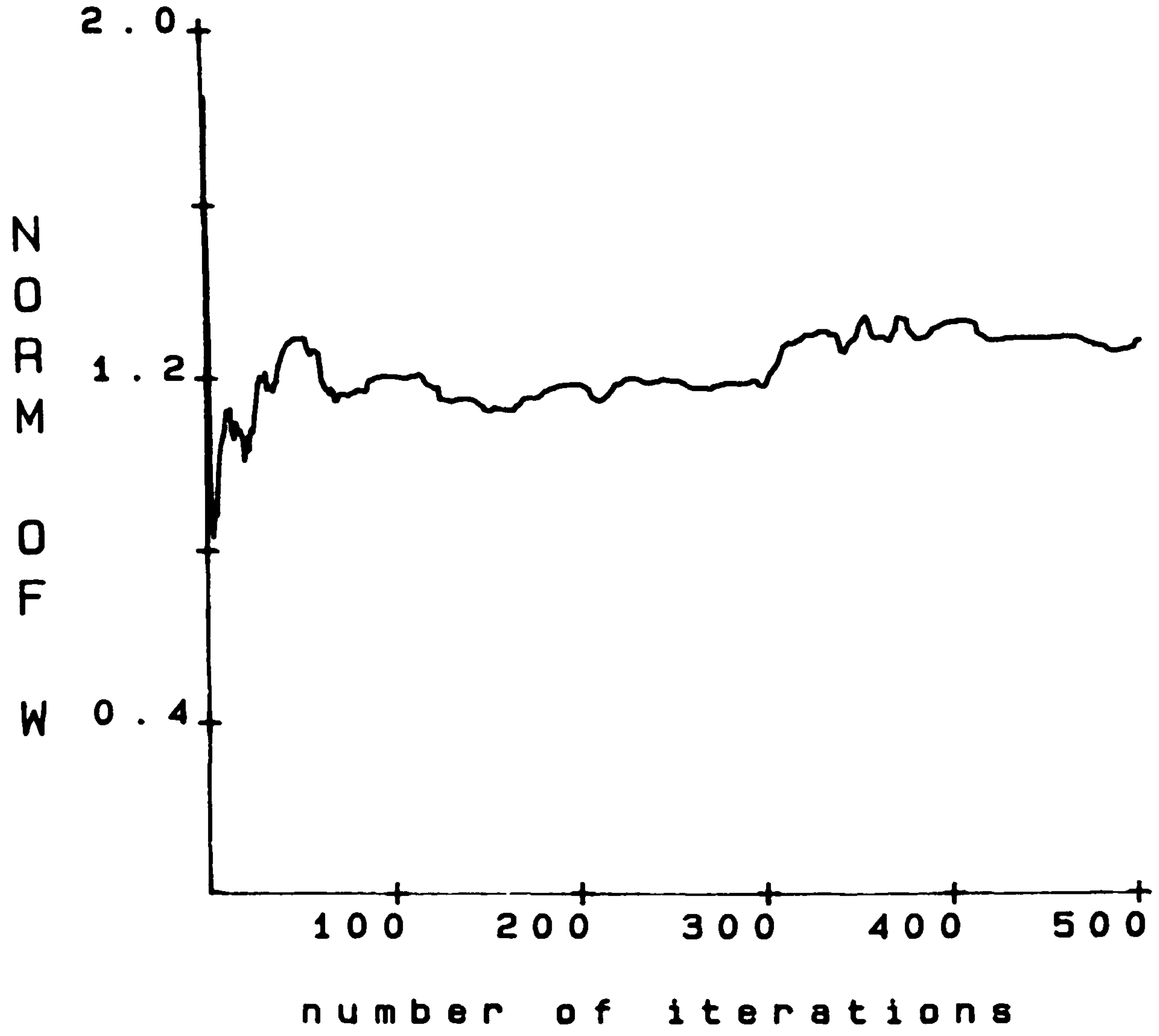


Figure 5.23 : Convergence of $|W_K|$. ELS algorithm.

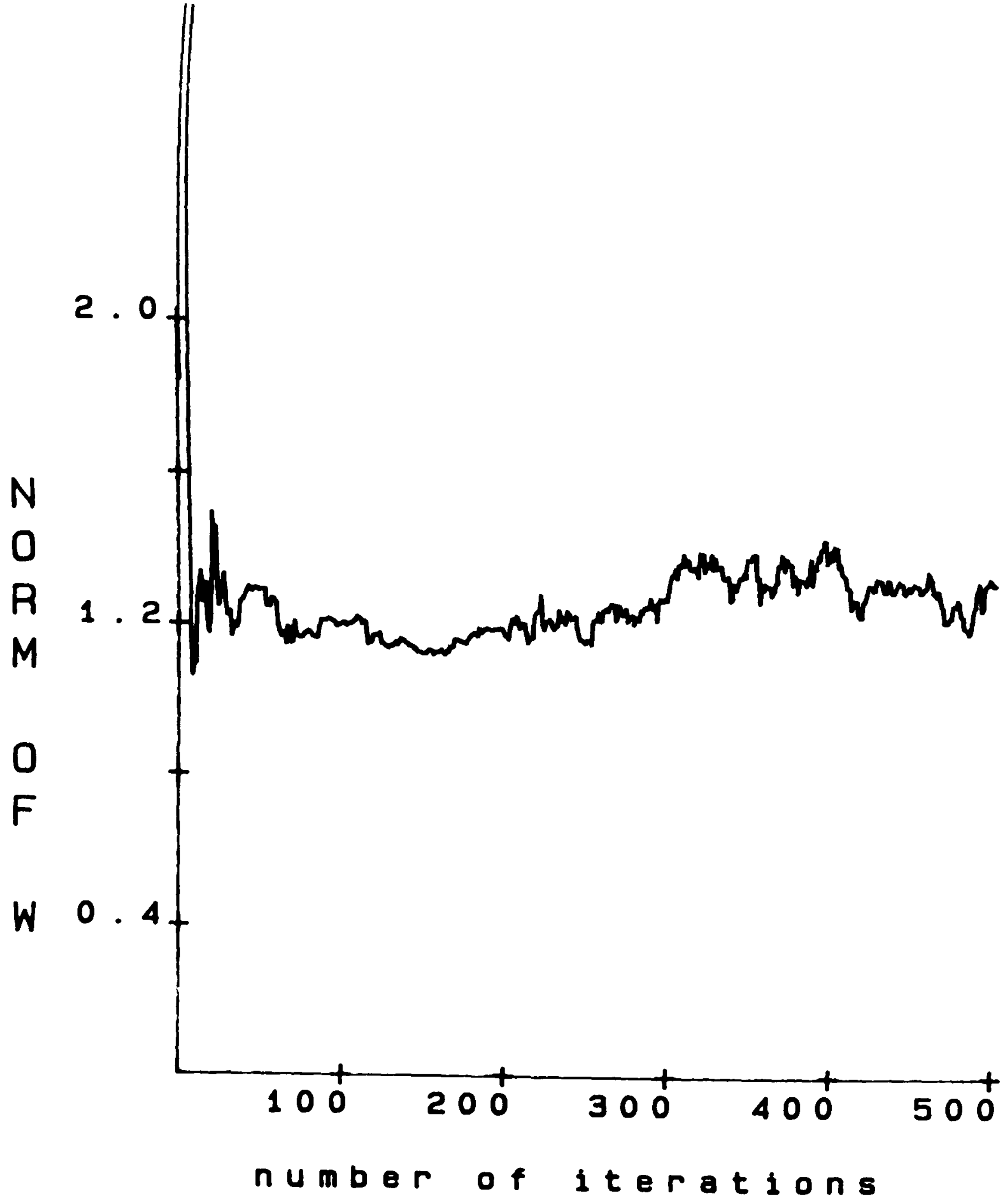


Figure 5.24 : Convergence of $|W_K|$. RLS algorithm.

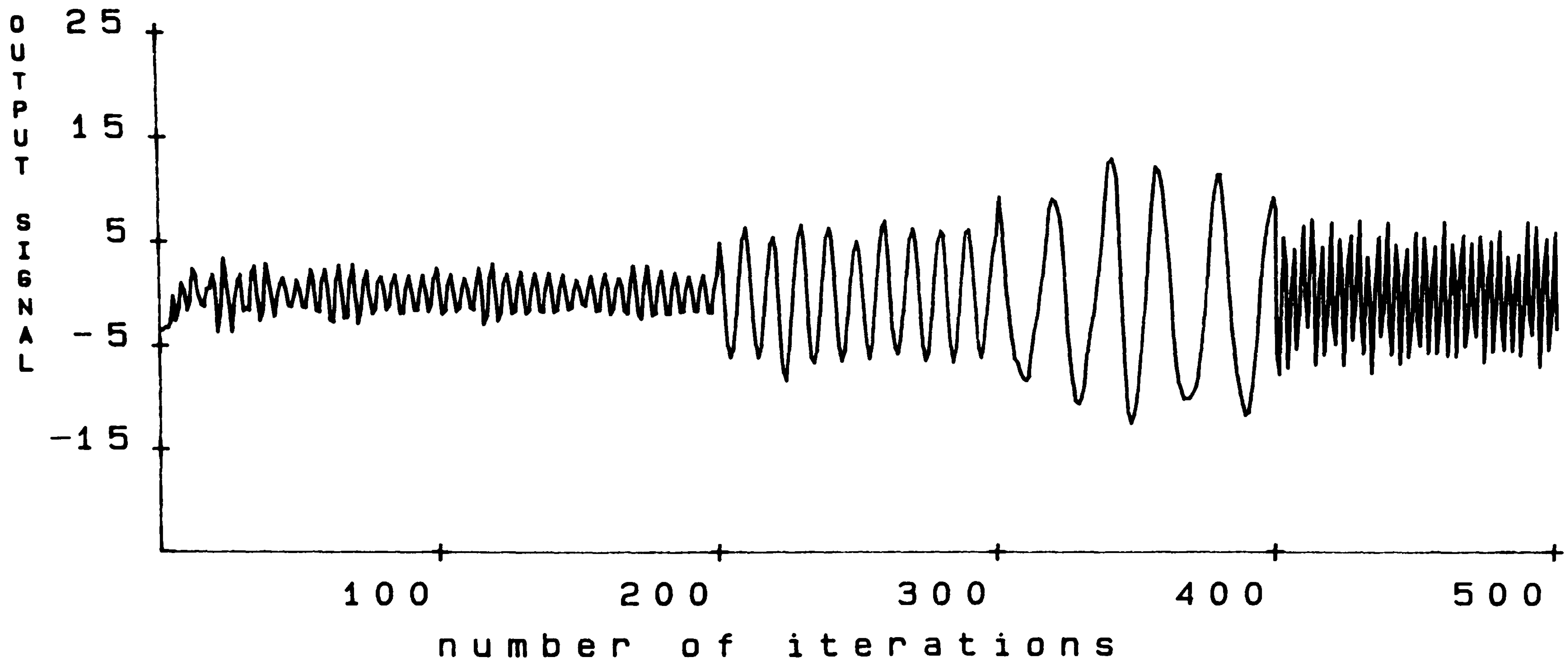


Figure 5.25 : Noise canceller output. ELS algorithm, L = 10.

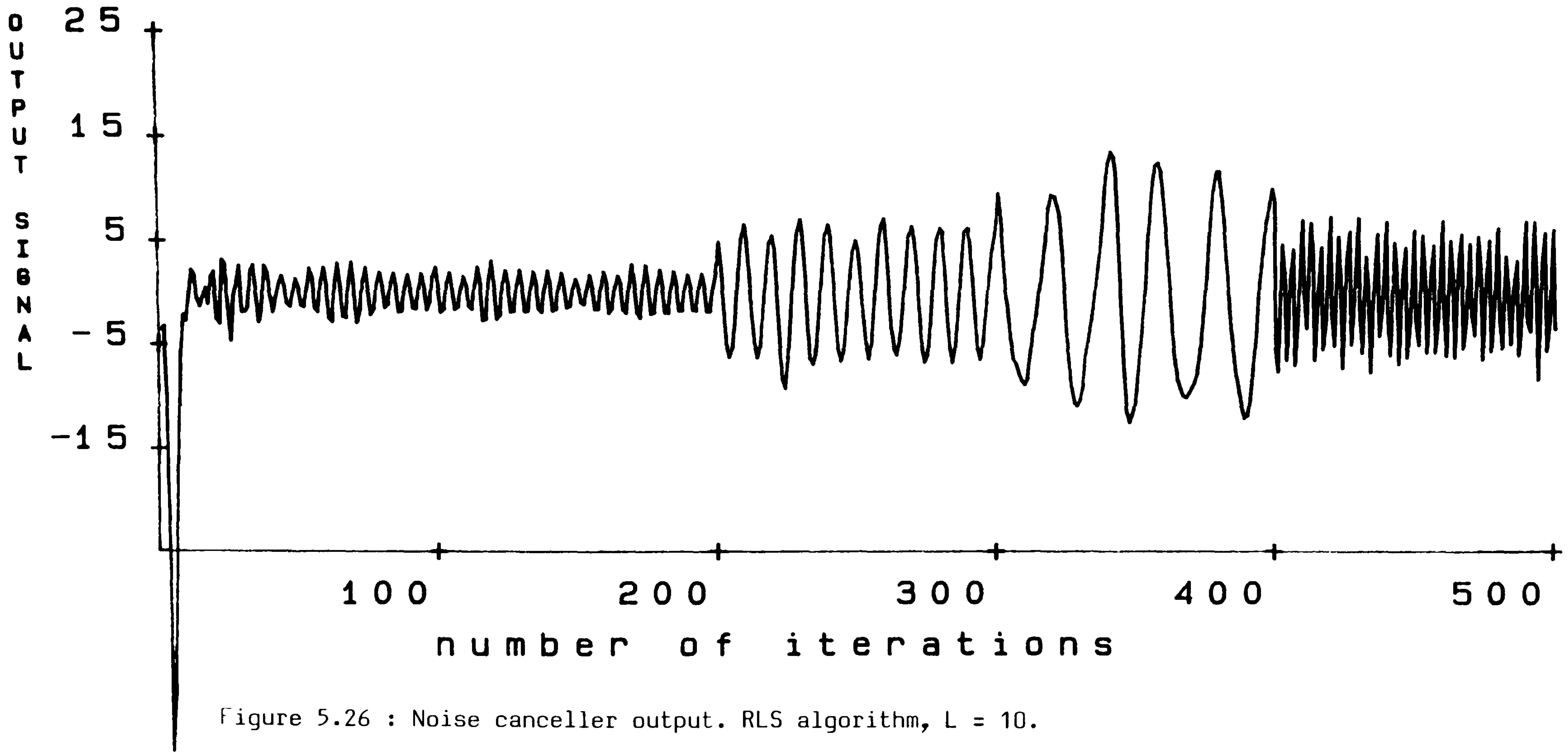


Figure 5.26 : Noise canceller output. RLS algorithm, L = 10.

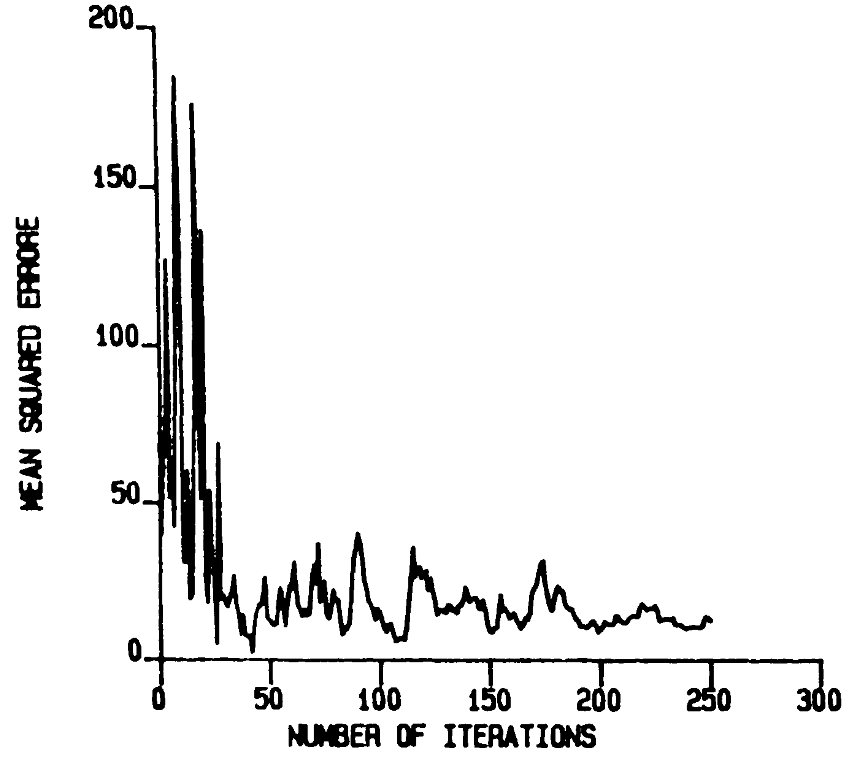


Figure 5.27 : Learning curves, averaged over 20 individual runs. ELS algorithm, Lattice filter implementation.

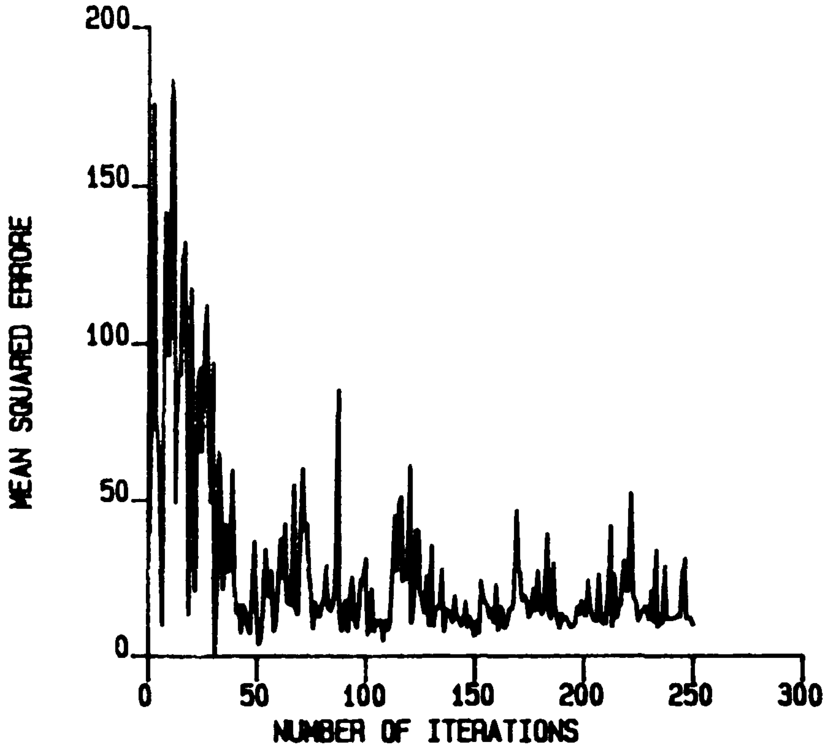


Figure 5.28 : Learning curves, averaged over 20 individual runs. Conventional Lattice gradient algorithm.

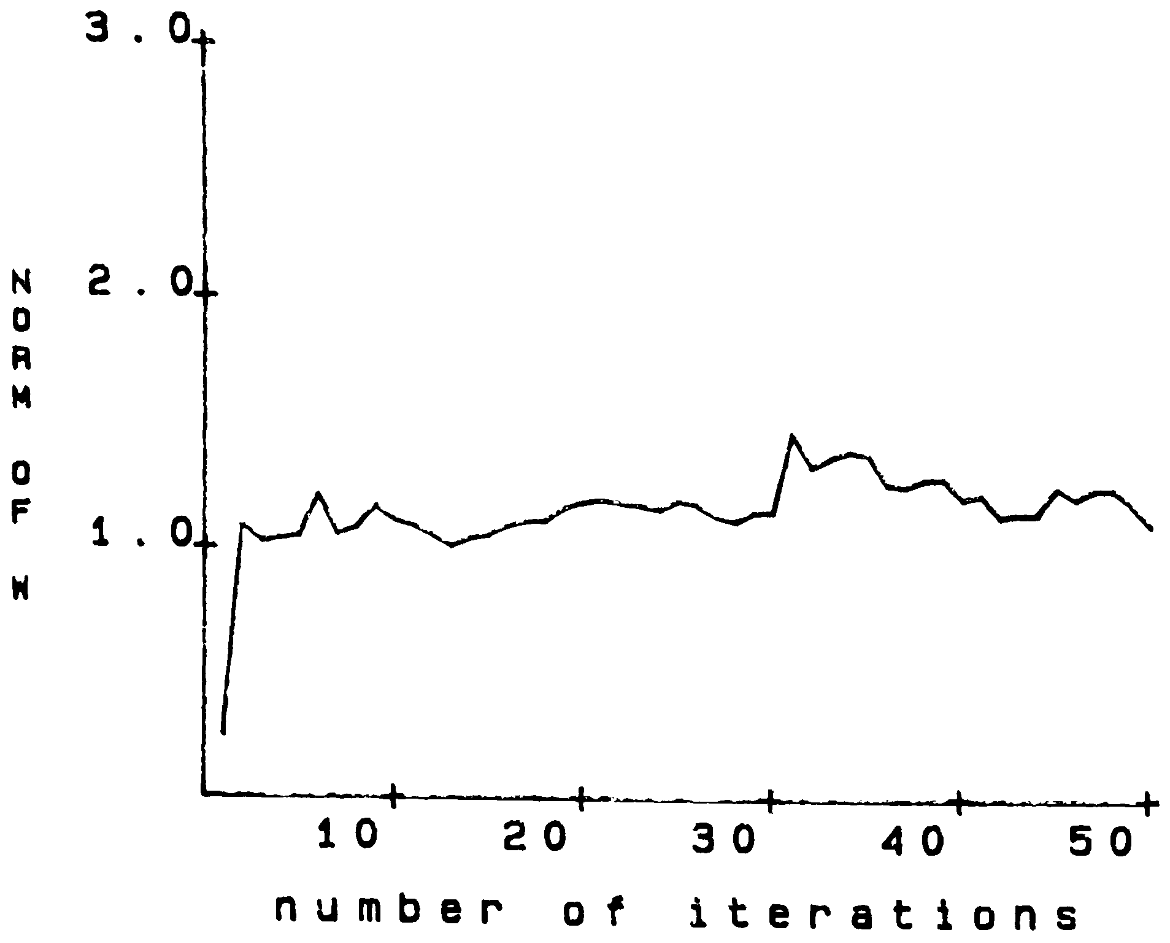


Figure 5.29 : Convergence of $|W_K|$. BIELS algorithm.
L = 5, EVR = 12.

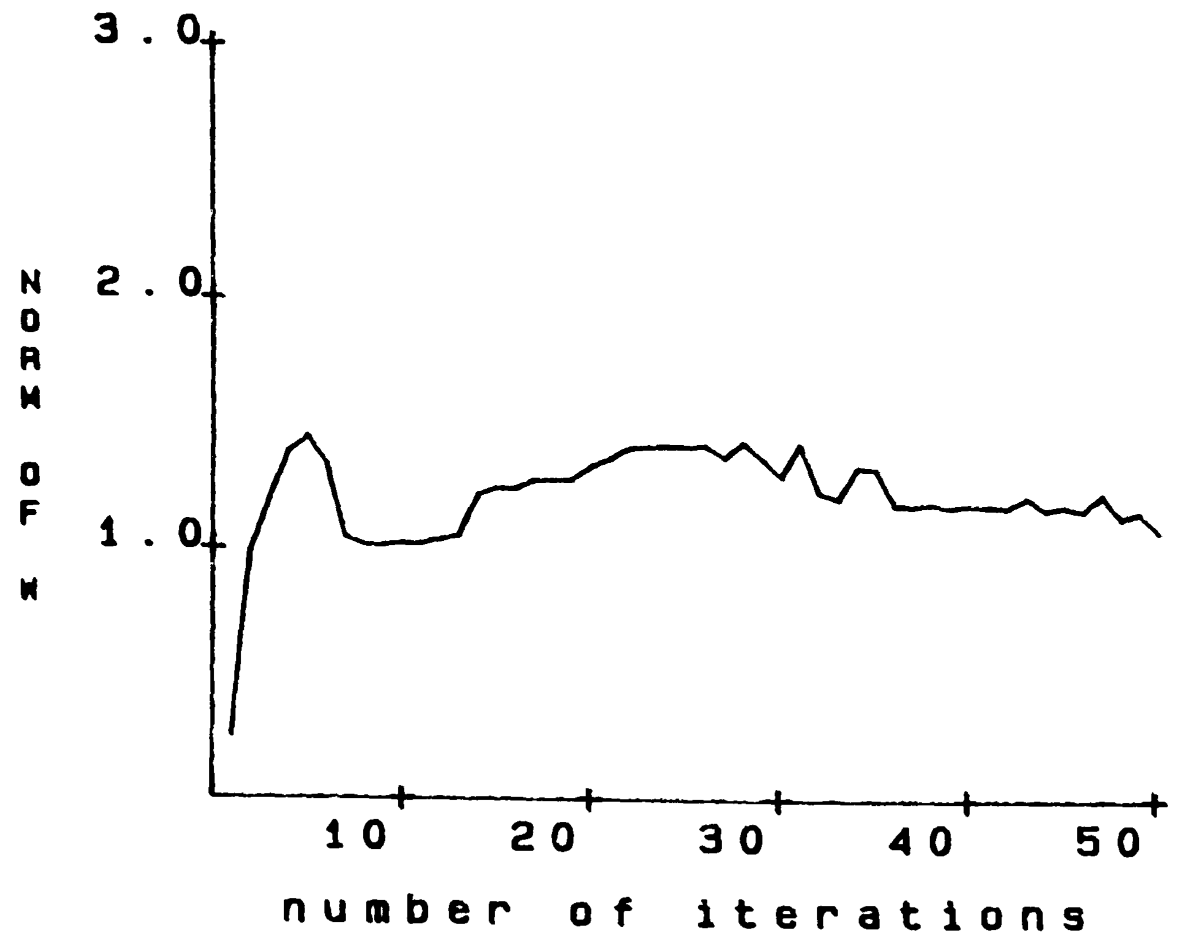


Figure 5.30 : Convergence of $|W_K|$. BICGM algorithm.
L = 5, EVR = 12.

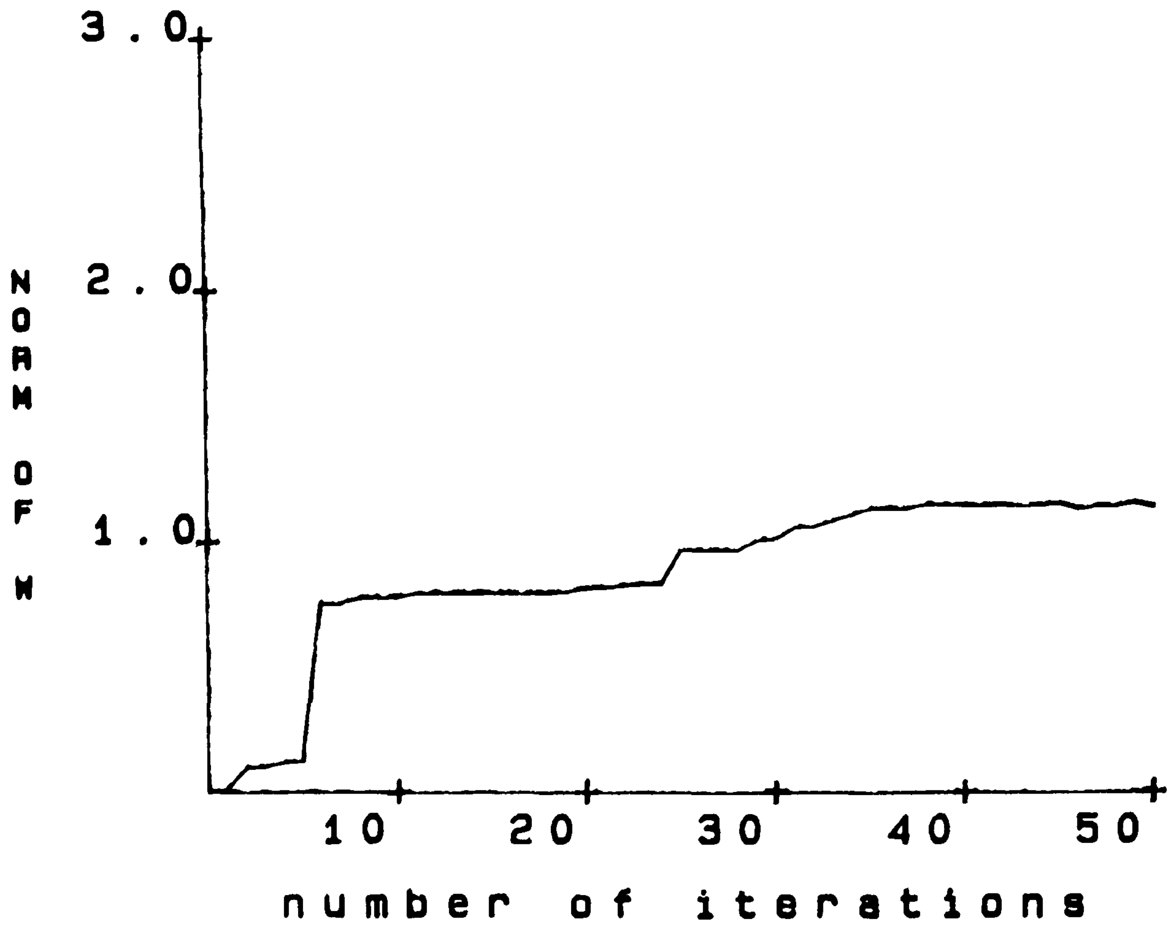


Figure 5.31 : Convergence of $|W_K|$. BIELS algorithm.
L = 5, EVR = 365.

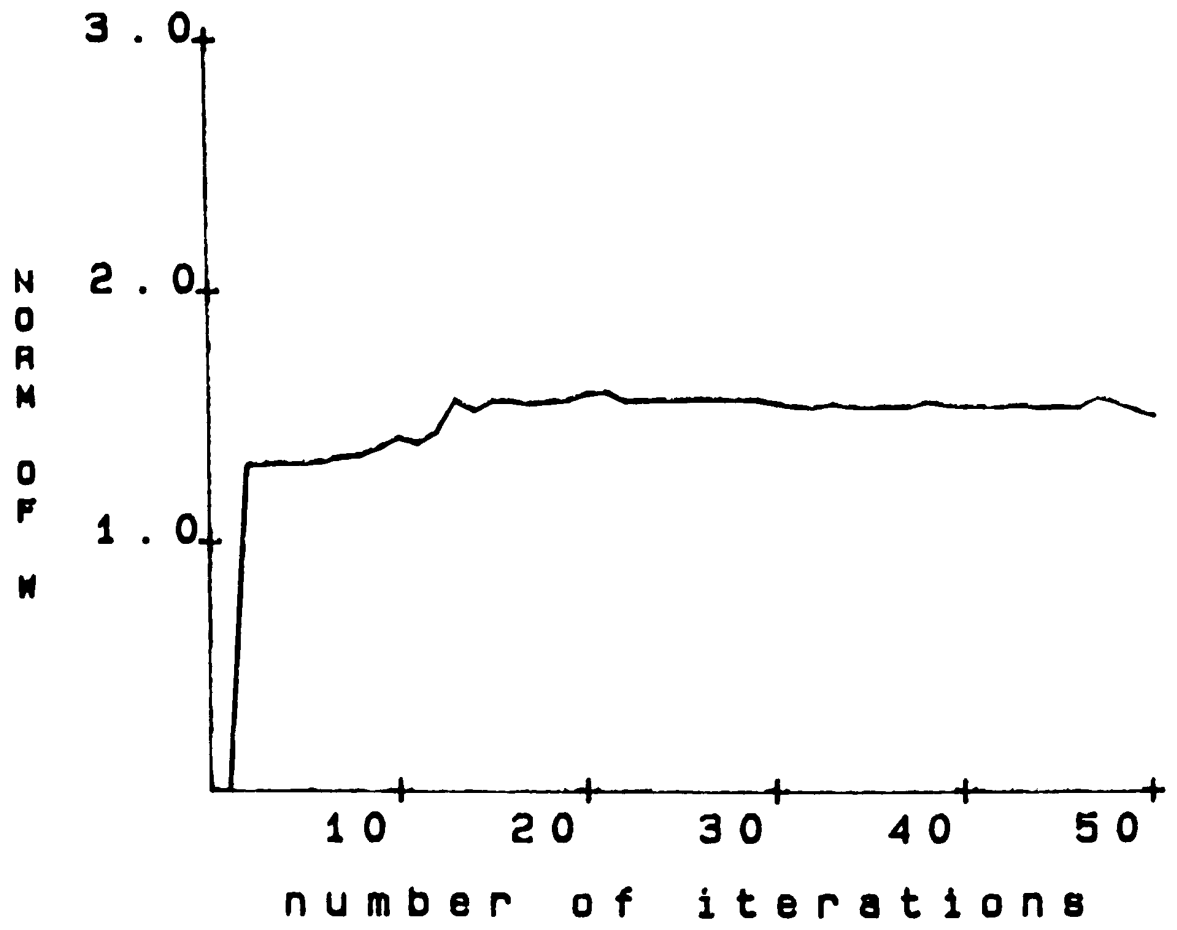


Figure 5.32 : Convergence of $|W_K|$. BICGM algorithm.
L = 5, EVR = 365.

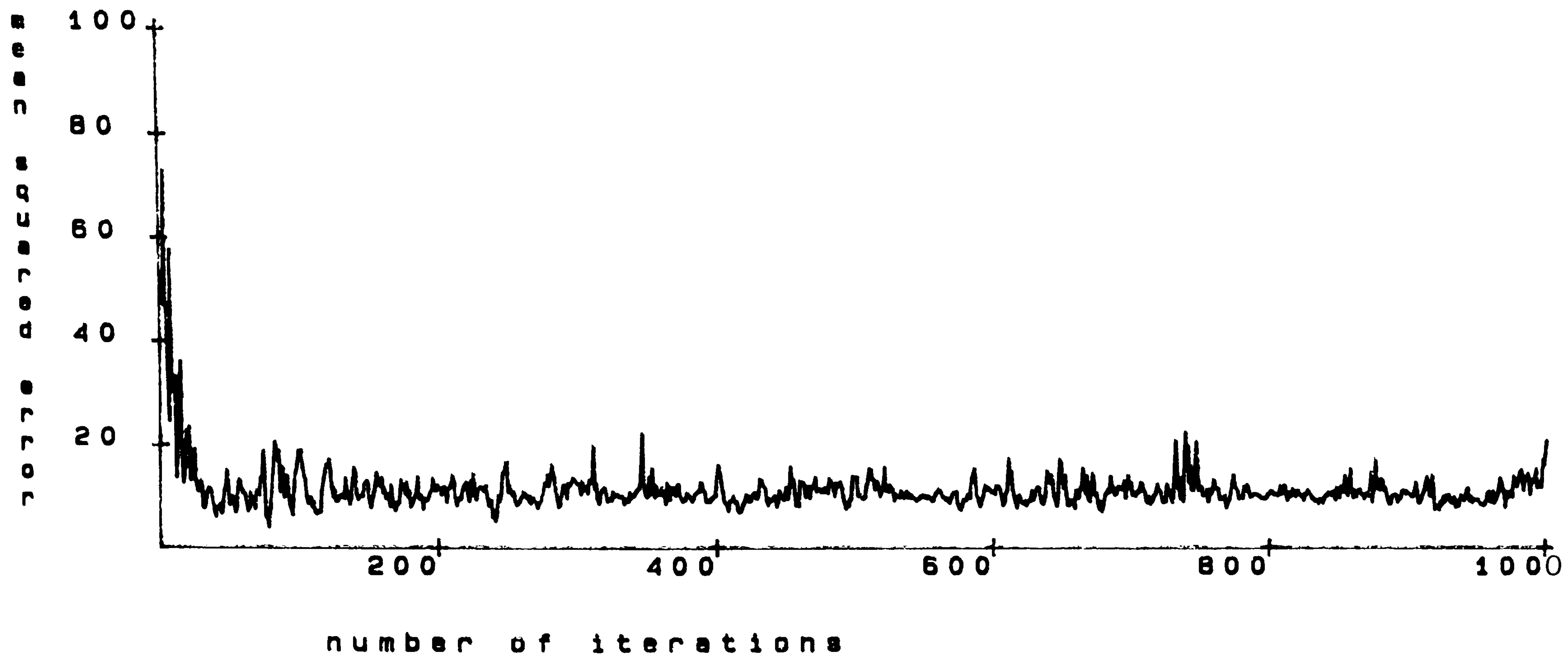


Figure 5.33 : Learning curve, averaged over 20 individual runs. Effect of finite precision implementation on ELS algorithm.

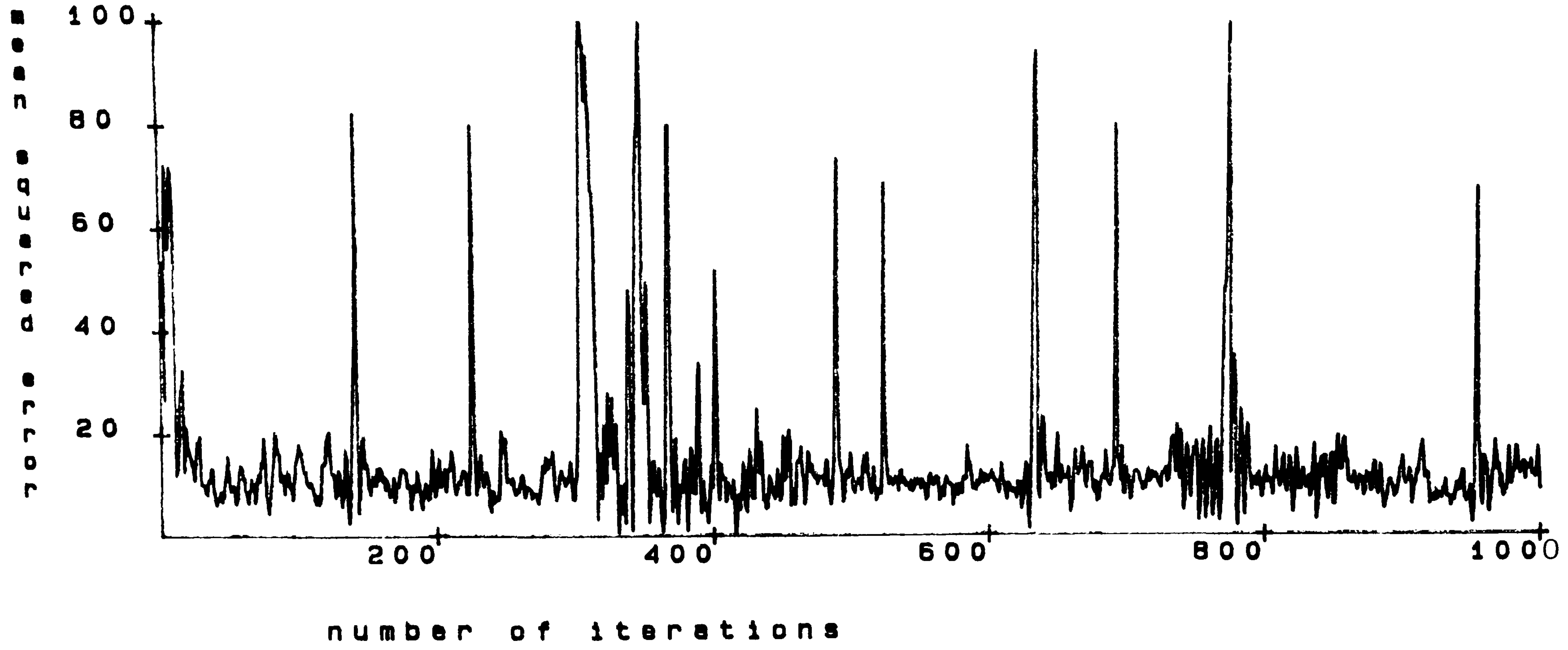


Figure 5.34 : Learning curve, averaged over 20 individual runs. Effect of finite precision implementation on RLS algorithm.

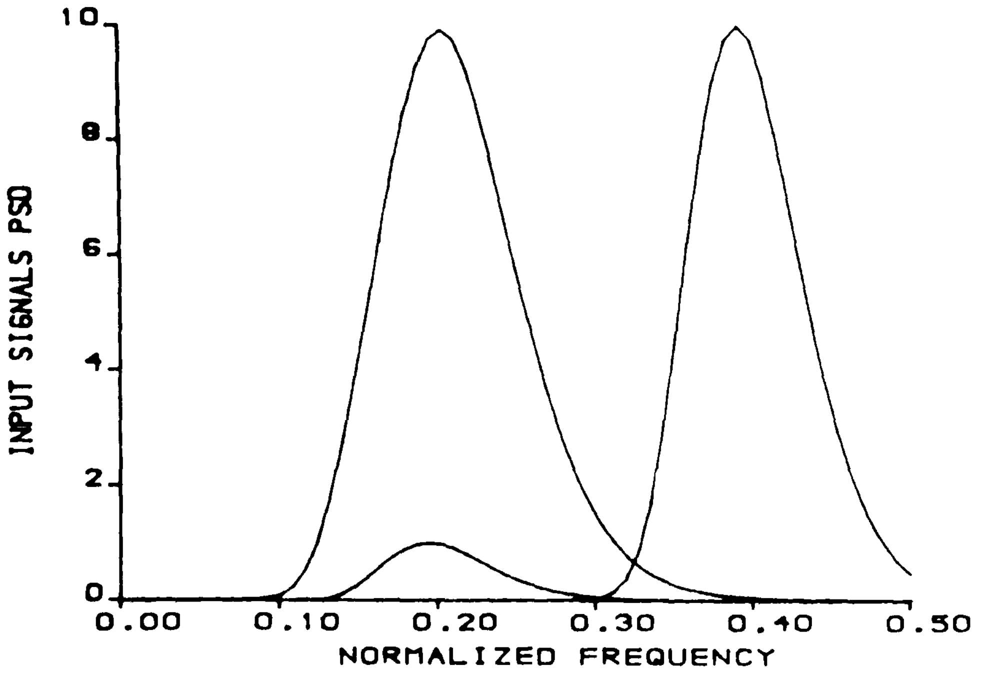


Figure 5.35 : Power spectrum of the incident signals, 128 data samples.

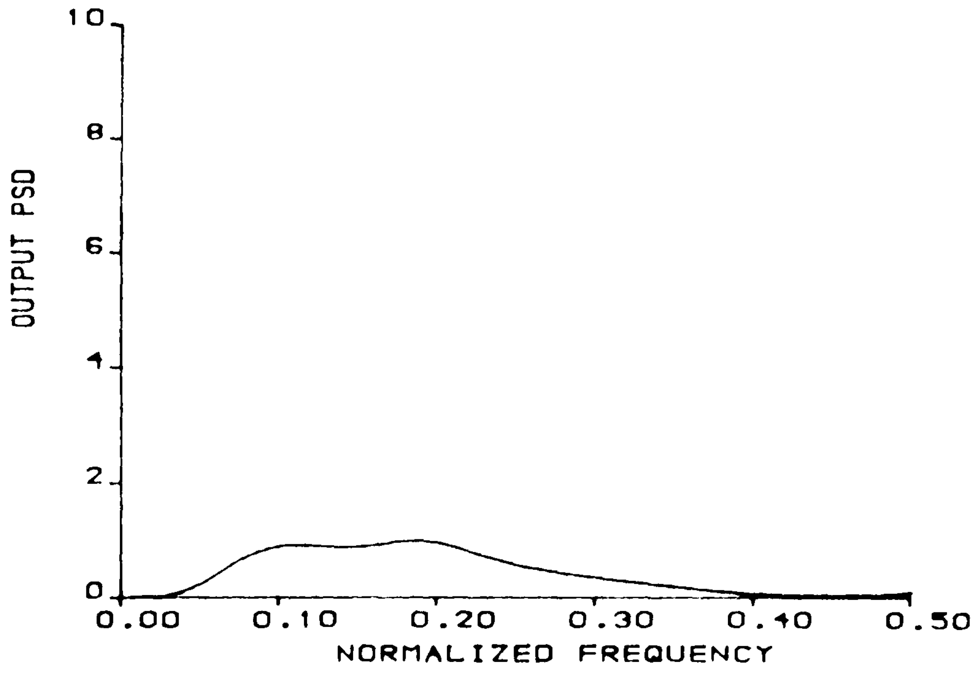


Figure 5.36 : Power spectrum of the array output sequence, 128 data samples ie. 128 iterations. LMS algorithm.

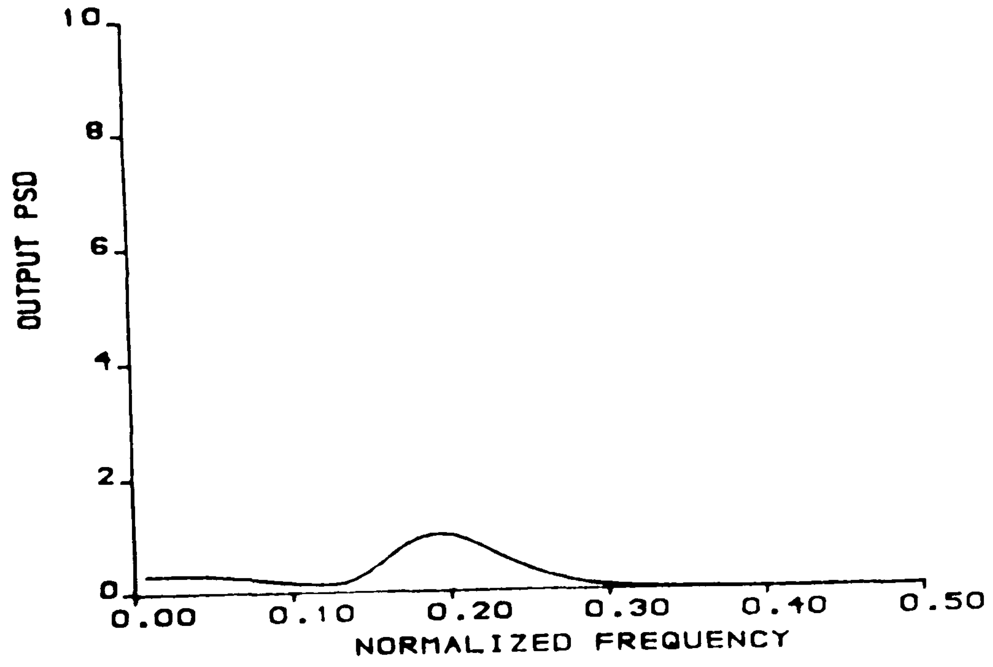


Figure 5.37 : Power spectrum of the array output sequence, 128 data samples ie. 128 iterations. ELS algorithm.

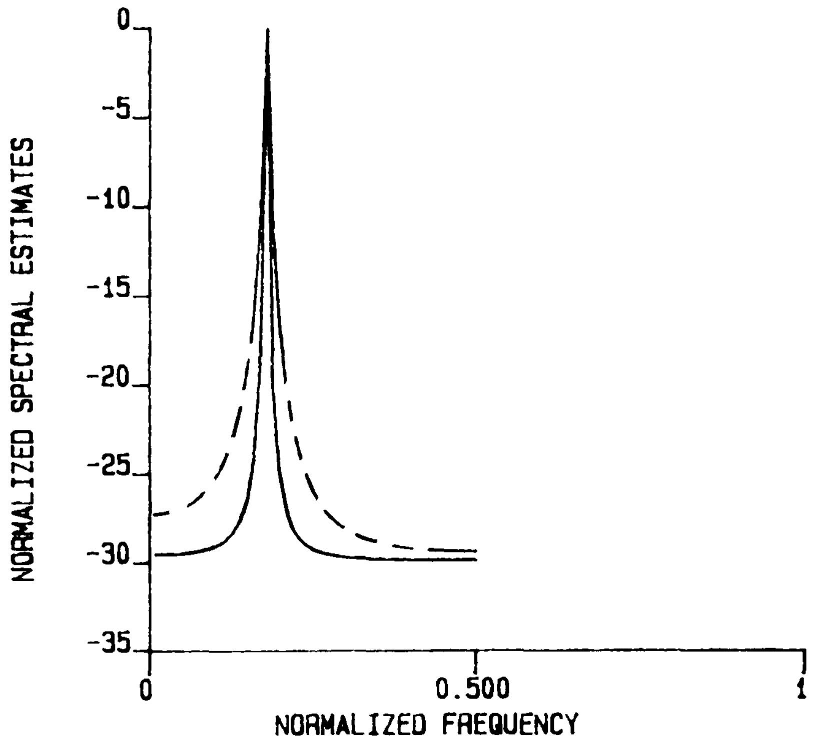


Figure 5.38 : Spectral estimates, f = 0.2. Broken curve K = 100, Solid curve K = 500. L = 3, SNR = 10dB.

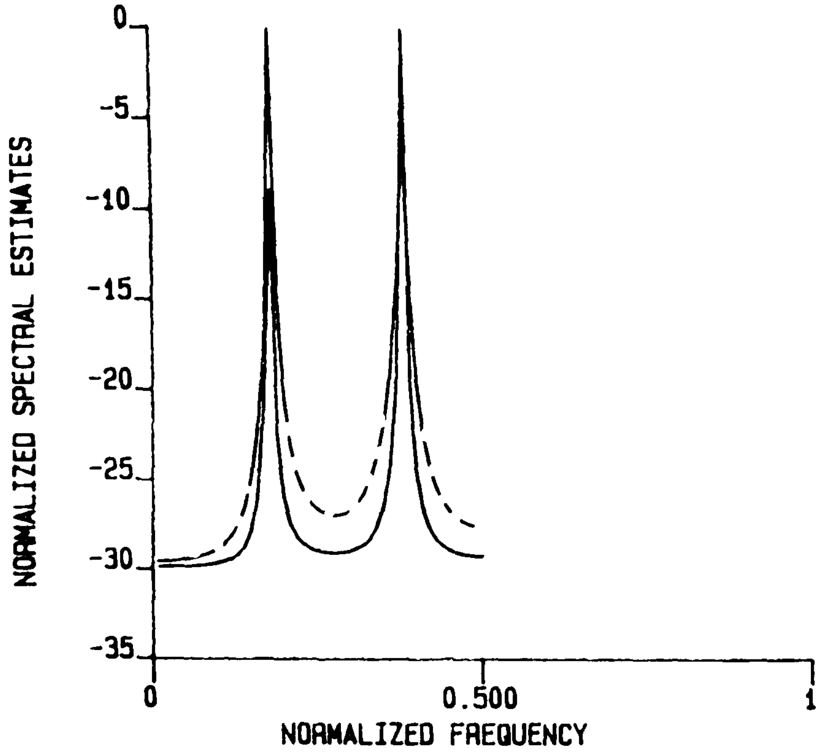


Figure 5.40 : Spectral estimates, $f_1 = 0.18$, $f_2 = 0.38$. Broken curve K = 100, Solid curve K = 500. L = 5, SNR = 0.0dB.

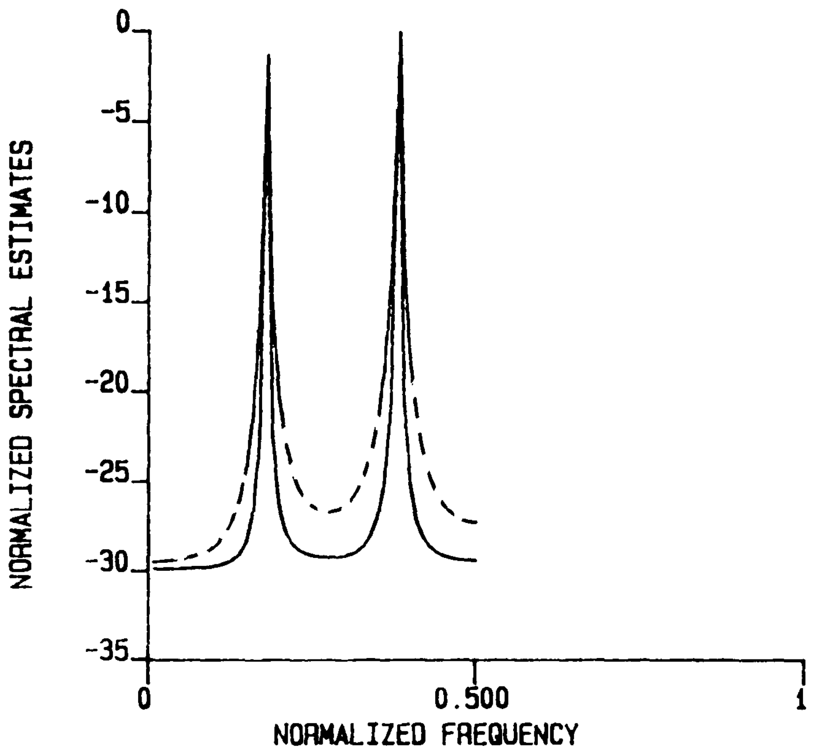


Figure 5.39 : Spectral estimates, $f_1 = 0.18$, $f_2 = 0.38$. Broken curve K = 100, Solid curve K = 500. L = 5, SNR = 10dB.

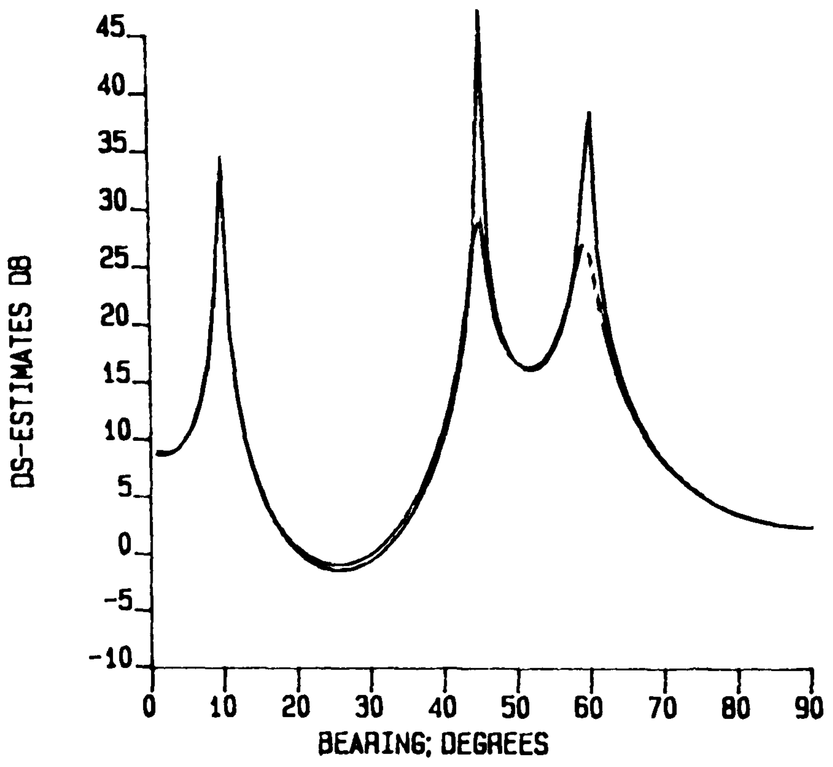


Figure 5.41 : DS-estimates.
Broken curve K = 150,
Solid curve K = 200.
SNR = 15, 20, 15dB.
Bearing = 10, 45, 60
degrees.

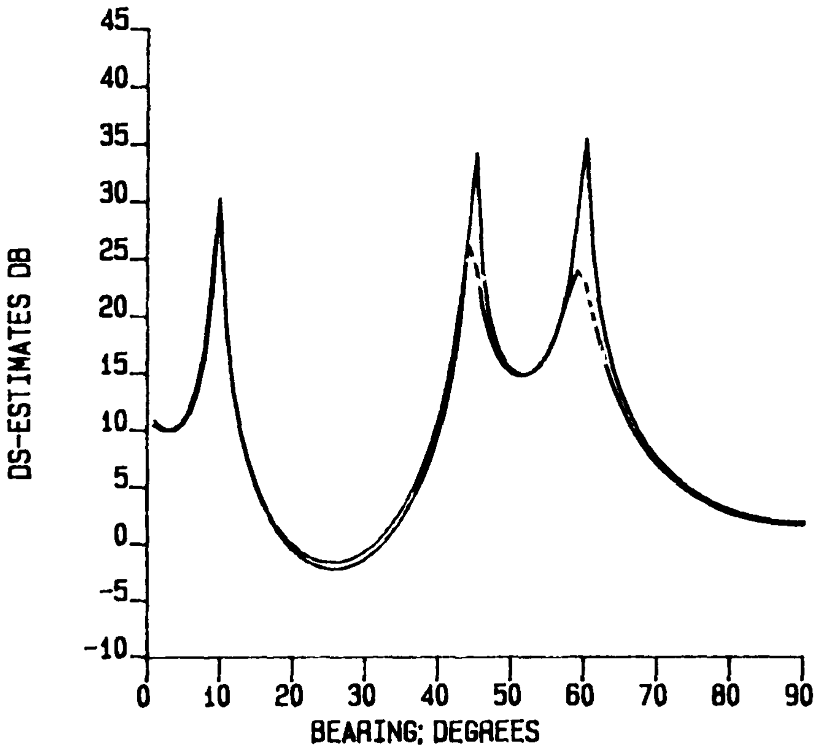


Figure 5.42 : DS-estimates.
Broken curve K = 150,
Solid curve K = 200.
SNR = 10, 15, 10dB.
Bearing = 10, 45, 60
degrees.

CHAPTER 6

CONCLUSIONS

The major objective of this thesis was to develop rapidly converging adaptive algorithms based on the accelerated gradient techniques for various applications of adaptive signal processing. The main emphasis was on the simplest AG technique, method of steepest descent with ELS. Because of its simplicity and its rapid initial convergence speed, the ELS technique appears to be a suitable approach to "relatively well conditioned problems".

The convergence analysis of the algorithm indicates its slow progress, in the case of the large condition numbers, to locate the minimum point, but reveals no information about what happens in the earlier iterations. It is illustrated in various optimization text books eg. [98], that the initial convergence speed of the modified Newton technique and ELS technique is comparable and both techniques achieve the same degree of accuracy within the first few iterations though the final convergence rate of ELS algorithm is severely affected by eigenvalue spread. This is the main disadvantage of the ELS technique.

Based on the above discussion, it is clear that the convergence speed of the ELS adaptive algorithms will deteriorate for extremely large condition numbers (eg. in the order of thousands). In this case faster convergence can be obtained by using RLS algorithms or preferably least squares lattice algorithms with the same computational requirement.

On the positive side, ELS adaptive algorithms appear to be effective minimization techniques for "relatively well conditioned" problems. Simulation results presented in chapter 5 clearly demonstrated the close performance of the standard RLS and ELS adaptive algorithms for eigenvalue ratios of up to several hundreds (which can be regarded as a high figure). Nevertheless, with a computational load equal to that of RLS algorithms and a convergence speed affected by eigenvalue spread of the data covariance matrix, ELS adaptive algorithms can offer nothing but a better numerical stability over standard RLS techniques. This was verified by simulation results presented in chapter 5 which were obtained by considering some effects of the finite precision implementation.

This property of ELS adaptive algorithms is worth investigating in detail both analytically and experimentally.

The ELS algorithm can be modified by using the concepts of the conjugacy and linear independence. The result will be a conjugate gradients algorithm. A wide variety of CG algorithms are available and can be implemented adaptively. But one sample update implementation of these algorithms involves two cycles of computations. The first cycle involves the computation of the first and the second order statistics for each new data sample and the second cycle is to compute the least squares solution at that iteration. This type of implementation is a drawback for on line signal processing due to increased hardware complexity associated with the implementation and the attendant increased computing burden.

However, adopting block data adaptation and a parallel processing technique, various AG algorithms can be implemented efficiently for processing of stationary data. A typical example is the BICGM algorithm presented in chapter 3. With a computational requirement in the order of L and its consistent convergence properties, the algorithm is well suited for processing of long length, statistically stationary data.

In general, to take full benefit of the convergence properties of CGM algorithms in adaptive signal processing, block data adaptation and parallel processing techniques must be employed.

On the other hand, computationally efficient AG techniques can be developed for adaptive processing of short length stationary (or non-stationary) data by combining the ELS technique with a CGM as is explained below.

Recall that, the ELS technique offers rapid initial convergence speed and very slow convergence speed in the neighbourhood of the minimum point if the eigenvalue spread of the data covariance matrix is very high. Also, considering that the CGMs are L-step convergent, a property that can be preserved by choosing an appropriate initial estimate of the weight vector. Then the following unconstrained (though the idea can be extended to the constrained problems) adaptive algorithm can be proposed.

I. Initialize W_1

II. For $K = 1, 2, \ldots\ldots$ up to a finite number N update the weight vector by means of an ELS technique.

III. For $K = N$, W_K corresponds to a good estimate of the weight vector. Employ a CGM to compute the least square solution W_K^*.

If N is large enough, then the estimates of the sample covariance matrix and sample cross correlation vector will be unbiased and in this case $W_K^* \simeq W^*$. In practice unbiased estimates of the covariance matrix and cross correlation vector can be obtained after 2L to 3L iterations by using estimators (3.13) and (3.15b).

The above algorithm will be suitable for adaptive joint processing purposes where only a short length data is available.

In conclusion, the main contribution of this thesis was to develop various adaptive algorithms based on AG techniques suitable for "on line signal processing". The application of the proposed algorithms to the fields of adaptive noise cancelling, adaptive array processing and adaptive spectral estimation was discussed and extensive simulation results were presented to verify the theoretical findings of the thesis.

REFERENCES

REFERENCES

1. Rudin, H.J. Automatic Equalization Using Transversal Filters, IEEE Spectrum, Vol. 2, No. 1, January 1967.

2. Weinstein, S.B. Echo Cancellation in the Telephone Network, IEEE Communications Society Magazine, Vol. 15, No. 1, January 1977.

3. Qureshi, S.U.H. Adaptive Equalization, IEEE Communications Society Magazine, Vol. 21, No. 2, March 1982.

4. Sworder, D. Optimal Adaptive Control Systems, Academic Press, 1966.

5. Monzingo, R.A. and Miller, T.W. Introduction to Adaptive Arrays, John Wiley, New York, 1980.

6. Goodwin, G.C. and Sin, S.K. Adaptive Filtering Prediction and Control, Prentice-Hall, Englewood Cliffs, NJ, 1984.

7. Cowan, C.F.N. and Grant, P.M. Adaptive Filters, Prentice Hall, Englewood Cliffs, NJ, 1985.

8. Widrow, B. and Stearns, S.D. Adaptive Signal Processing, Prentice-Hall, Englewood Cliffs, NJ, 1984.

9. Honig, M.L. and Messerschmitt, D.G. Adaptive Filters - Structures, Algorithms and Applications, Kluwer Academic Publication, USA, 1984.

10. Widrow, B. and McCool, J.M. A Comparison of Adaptive Algorithms Based on the Methods of Steepest Descent and Random Search, IEEE Trans., Vol. AP-24, No. 5, Sept. 1976, pp.615-637.

11. Gaurpe, D. Identification of Systems, Van Nostrand Rienhold, London, 1972.

12. Bennett, R.J. Spatial Time Series, Pion Ltd., 1979.

13. Gill, P.E., Murray, W. and Wright, M.H. Practical Optimization, Academic Press, London and New York, 1981.

14. Hestenes, M. Conjugate Direction Methods in Optimization, Springer-Verlag, New York, 1980.

15. Hestenes, M. Optimization Theory, John Wiley, New York, 1975.

16. Gardener, W.A. Learning Characteristics of Stochastic-Gradient-Descent Algorithms; A General Study, Analysis and Critique Signal Processing, North Holland, 1984.

17. Farden, D.C. Tracking Properties of Adaptive Signal Processing Algorithms, IEEE Trans., Vol. ASSP-29, No. 3, June 1981.

18. Farden, D.C. Stochastic Approximation with Correlated Data, IEEE Trans., Vol. IT-27, No. 1, January 1981.

19. Macchi, O. and Eweda, E. Second Order Convergence Analysis of Stochastic Adaptive Linear Filtering, IEEE Trans., Aut-Cont., Vol. AC-28, No. 1, January 1983.

20. Daniel, T.P. Adaptive Estimation with Mutually Correlated Training Sequences, IEEE Trans., Sys-Sci and Cybernetics, Vol. SSC-6, No. 1, January 1970.

21. Mazo, J.E. On the Independent Theory of Equalizer Convergence, Bell System Technical Journal, Vol. 58, No. 5, May-June 1979.

22. Gersho, A. Adaptive Equalization of Highly Dispersive Channels for Data Transmission, Bell System Technical Journal, January 1969.

23. Ungerboeck, G. Theory on the Speed of Convergence in Adaptive Equalizers for Digital Communication, IBM Journal of Research and Development, Vol. 16, No. 6, November 1972.

24. Widrow, B. et al. Stationary and Non-Stationary Learning Characteristics of the LMS Adaptive Filter, Proceedings IEEE, Vol. 64, No. 8, August 1976.

25. Hsia, T.C. Convergence Analysis of LMS and NLMS Adaptive Algorithms, ICASSP 83, Boston, 15A.2, pp.667-670.

26. Claasen, T.A.C.M.and Mecklenbräuker, W.G. Comparison of the Convergence of Two Algorithms for Adaptive FIR Digital Filters, IEEE Trans., Vol. ASSP-29, No. 3, June 1981.

27. Horowitz, L.L. and Senne, K.D. Performance Advantage of Complex LMS for Controlling Narrow-Band Adaptive Arrays, IEEE Trans., Vol. ASSP-29, No. 3, June 1981.

28. Falconer, D. and Ljung, L. Application of Fast Kalman Estimation to Adaptive Equalization, IEEE Trans., Vol. COM-26, No. 10, October 1978.

29. Godard, D. Channel Equalization Using a Kalman Filter for Fast Data Transmission, IBM Journal of Research and Development, Vol. 18, No. 3, May 1974.

30. Gitlin, R.D. and Magee, F.R. Self-Orthogonalizing Adaptive Equalizer Algorithms, IEEE Trans., Vol. COM-25, No. 7, July 1977.

31. Mueller, K.H. A New, Fast-Converging Mean Square Algorithm for Adaptive Equalizers with Partial Response Signalling, Bell System Technical Journal, January 1975.

32. Qureshi, S.U.H. Fast Start-Up Equalization with Periodic Training Sequences, IEEE Trans., Vol. IT-23, No. 5, September 1977.

33. Chang, R.W. A New Equalizer Structure for Fast Start-Up Digital Communications, Bell System Technical Journal, Vol. 50, No. 6, July-August 1971.

34. Friedlander, B. Lattice Filters for Adaptive Processing, Proceedings IEEE, Vol. 70, No. 8, August 1982.

35. Makhoul, J. A Class of All-Zero Lattice Digital Filters: Properties and Applications, IEEE Trans., ASSP, Vol. 26, No. 4, August 1978.

36. Messerschmitt, D.G. A Class of Generalized Lattice Filters, IEEE Trans., ASSP, Vol. 28, No. 2, April 1980.

37. Itakura, F. and Saito, S. Digital Filtering Techniques for Speech Analysis and Synthesis, Proceedings 7th International Conference on Acoustics, Budapest, Paper 25-C1, 1971, pp. 261-264.

38. Burg, J. Maximum Entropy Spectral Analysis, 37th Annual International Meeting Society of Exploratory Geophysicists, Oklahoma City, OK, 1967.

39. Gibson,C.J. and Haykin,S. Learning Characteristics of Adaptive Lattice Filtering Algorithms, IEEE Trans., ASSP, Vol. 28, No. 6, December 1980.

40. Sohie, G.R.L. and Sibul, L.H. Stochastic Convergence Properties of the Adaptive Gradient Lattice, IEEE Trans., ASSP, Vol. 32, No. 1, February 1984.

41. Satorius, E.H. and Alexander, S.T. Channel Equalization Using Adaptive Lattice Algorithms, IEEE Trans., Vol. COM-27, No. 6, June 1979.

42. Grant, P.M. and Rutter, M.J. Application of Gradient Adaptive Lattice Filters to Channel Equalization, Proceedings of IEE, Vol. 131, Pt.F, No. 5, August 1984.

43. Rutter, M.J. et al. Timed Gradient Adaptive Lattice Equalizers, Proceedings of IEE, Vol. 132, Pt.F, No. 3, June 1985.

44. Shapard,J.M., Edelblube, D. and Kinnison, G. Adaptive Matrix Inversion, Naval Under Sea R & D Centre, NUC-TN-528, May 1971.

45. Cioffi, J.M. and Kailath, T. Fast, Fixed-Order, LS Algorithms for Adaptive Filtering, ICASSP 83, Boston, 15A.5, pp. 679-682.

46. Falconer,D.D. and Ljung,L. Application of Fast Kalman Estimation to Adaptive Equalization, IEEE Trans., Vol. COM-26, No. 10, October 1978.

47. Ljung, L. et al. Fast Calculation of Gain Matrices for Recursive Estimation Schemes, International Journal of Control, Vol. 27, No. 1, January 1978.

48. Levinson, N. The Wiener RMS (Root-Mean-Square) Error Criterion in Filter Design and Prediction, Journal of Mathematics and Physics, Vol. 25, 1947, pp. 261-278.

49. Morf, M. and Lee, D.T. Recursive Least Squares Ladder Forms for Fast Parameter Tracking, Proceedings of IEEE Conference Decision and Control, San Diego, CA, USA, January 1979, pp. 1362-1367.

50. Satorius, E.H. and Pack, J.D. Application of Least Squares Lattice Algorithms to Adaptive Equalization, IEEE Trans., Vol. COM-29, 1981, pp. 136-142.

51. Ljung, L., Morf,M. and Falconer,D. Fast Calculations of Gain Matrices for Recursive Estimation Schemes, International Journal of Control, Vol. 27, No. 1, 1978.

52. Durrani, T.S. and Murukutla, N.L.M. Recursive Algorithms for Adaptive Lattices, Electronic Letters, Vol. 15, No. 25, 6th December 1979.

53. Durrani, T.S. and Murukutla, N.L.M. Convergence of Adaptive Lattice Filters, Electronic Letters, Vol. 15, No. 25, 6th December 1979.

54. Lee, D.T.L., Friedlander, B. and Morf, M. Recursive Ladder Algorithms for ARMA Modelling, IEEE Trans., Vol. AC-27, No. 8, August 1982.

55. Lee, D.T.L. and Morf, M. Recursive Square Root Estimation Algorithms, ICASSP 1980, Denver, CO., USA, pp. 1005-1017.

56. Morf, M. Ladder Forms in Estimation and System Identification, Conference Circuit and Systems and Computers, Monteray, CA., November 1977, pp. 424-429.

57. Morf, M. Recursive Ladder Least Squares Forms for Far Parameter Tracking, IEEE Conference Proceedings Decision and Control, San Diego, CA., 1979, pp. 1362-1367.

58. Owsley, N.L. A Recent Trend in Adaptive Spatial Processing for Sensor Arrays: Constrained Adaptation, in J.W.R. Griffiths et al (eds), Signal Processing, Academic, 1973.

59. Evans, R.J. and Ahmed, K.M. Robust Adaptive Array Antenna, Journal Acoustics Society of America, Vol. 71, No. 2, February 1982.

60. Frost,O.L. An Adaptive Algorithm for Linearly Constrained Adaptive Array Processing, Proceedings IEEE, August 1972.

61. Gardener, W.A. and Brown, W.A. Rapidly Converging Computationally Efficient Algorithms for Adaptive Sensor Arrays, Signal and Image Processing Laboratory, Dept. of Electrical and Computer Engineering, University of California, Davis, CA, (to be published).

62. Klemm, R. Adaptive Clutter Suppression for Airborn Phase Array Radar, IEE Proceedings, Vol. 130, Pt. F, February 1983.

63. Griffiths, J.W.R. Adaptive Array Processing, A Tutorial, IEE Proceedings, Vol. 130, Pts. F and H, No. 1, February 1983,

64. Al Khatib, H. and Compton Jr, R.T. A Gain Optimizing Algorithm for Adaptive Arrays, IEEE Trans., Vol. AP-26, 1978, pp. 275-282.

65. Durrani, T.S. and Sharman, K.C. Eigenfilter Approaches to Adaptive Array Processing, IEE Proceedings, Vol. 130, Pts. F and H, No. 1, February 1983.

66. Durrani, T.S. and Sharman, K.C. Directional Spectra and Interference Removal Techniques in Adaptive Arrays, Proceedings of the Institute of Acoustics, Spectral Analysis and its Use in Under Water Acoustics, Imperial College, London, 29-30 April 1982.

67. Sharman, K.C. and Durrani, T.S. A Comparative Study of Modern Eigenstructure Methods for Learning Estimation - A New High Performance Approach, IEEE Conference on Decision and Control, Greece, December 1986.

68. Thompson, P.A. An Adaptive Spectral Analysis Technique for Unbiased Frequency Estimation in the Presence of White Noise, Proceedings 13th Asilomar Conference Circuit and Systems and Computers, Pacific Grove, CA, November 1979, pp. 529-533.

69. Larimore, M.G. Adaptation Convergence of Spectral Estimation Based on Pisarenko's Harmonic Retrieval, IEEE Trans., Vol. ASSP-31, No. 4, August 1983.

70. Reddy, V.U. et al. Least Square Type Algorithm for Adaptive Implementation of Pisarenko's Harmonic Retrieval Method, IEEE Trans., Vol. ASSP-30, No. 3, June 1982.

71. Yaminysharif, M. and Durrani, T.S. Adaptive Signal Processing Using a Modified Gradient Estimation Technique, IEE Colloquium on Adaptive Filters, London, No. 1985/76, September 1985.

72. Yaminysharif, M. and Durrani, T.S. Adaptive Signal Processing Using a Modified Gradient Estimation Technique, ICASSP 1986, Tokyo, Vol. 4, April 1986.

73. Chen, H. et al. Adaptive Spectral Analysis by the Conjugate Gradient Method, ICASSP 85, Florida, USA, 1985.

74. Chen, H. et al. Adaptive Spectral Estimation by the Conjugate Gradient Method, IEEE Trans., Vol. ASSP-34, No. 2, April 1986.

75. Hen, Y. and Chou, P.K. Effective Adaptive Pisarenko Spectrum Estimate, ICASSP 86, Tokyo, April 1986.

76. Fuhrman, D.R. and Liu, B. Rotational Search Methods for Adaptive Pisarenko Harmonic Retrieval, IEEE Trans., Vol. ASSP 34, No. 6, December 1986.

77. Peters, G. and Wilkinson, J.H. Inverse Iteration, Ill-Conditioned Equations and Newton's Method, SIAM Revue, Vol. 21, 1979, pp. 339-360.

78. Ferrara, E.R. and Widrow, B. The Time-Sequenced Adaptive Filter, IEEE Trans., Vol. ASSP-29, No. 3, June 1981.

79. Clark, G.A., Mitra, S.K. and Parker, S.R. Block Implementation of Adaptive Digital Filters, IEEE Trans., Vol. ASSP-29, No. 3, June 1981.

80. Kim, J.K. and Davisson, L.D. Adaptive Linear Estimation for Stationary M-Dependent Processes, IEEE Trans., Vol. IT-21, No. 1, January 1975.

81. Kailath, T. Linear Systems, Prentice-Hall, Englewood Cliffs, 1980.

82. Hestenes, M.R. and Stiefel, E. Methods of Conjugate Gradients for Solving Linear Systems, Journal of Research of the National Bureau of Standards, Vol. 49, No. 6, December 1952.

83. Fletcher, R. and Reeves, C.M. Function Minimization by Conjugate Gradients, Computer Journal, Vol. 7, No. 2, July 1964.

84. Powell,J.D. An Iterative Method for Finding Stationary Values of a Function of Several Variables, Computer Journal, Vol. 5, No. 2, July 1962.

85. Luenberger, D.G. Optimization by Vector Space Methods, Wiley, New York, 1969.

86. Satorius, E., Lanxch, S., Lee, S. and Griffiths, D.O. Fixed Point Implementation of Adaptive Digital Filters, Proceedings ICASSP 83, Boston, MA, April 1983.

87. Samson, C. and Reddy, V. Fixed Point Error Analysis of the Normalized Ladder Algorithm, Proceedings ICASSP 82, Paris, France, May 1982.

88. Ling, F. and Proakis, J. Numerical Accuracy and Stability; Two Problems of Adaptive Estimation Algorithms Caused by Round-Off Error, Proceedings ICASSP 84, San Diego, CA, March 1984.

89. Widrow, B. et al. Adaptive Noise Cancelling: Principles and Applications, Proceedings of the IEEE, Vol. 63, No. 12, December 1975.

90. Pisarenko, V.F. The Retrieval of Harmonics from a Covariance Function, Geophys. J Roy, ASTR, Soc, 1973.

91. Kay, S.M. and Marple Jr., S.L. Spectrum Analysis - A Modern Perspective, Proceedings of the IEEE, Vol. 69, No. 11, November 1981.

92. Durrani, T.S. and Sharman, K.C. Extraction of an Eigenvector- Oriented "Spectrum" from the MESA Coefficients, IEEE Trans., Vol. ASSP-30, No. 4, August 1982.

93. Sharman, K.C. and Durrani, T.S. Spatial Lattice Filter for High-Resolution Spectral Analysis of Array Data, IEE Proc., Vol. 130, Pt. F, No. 3, April 1983.

94. Hayes, M.H. and Clements, M.A. An Efficient Algorithm for Computing Pisarenko's Harmonic Decomposition Using Levinson's Recursion, IEEE Trans., Vol. ASSP-34, No. 3, June 1986.

95. Martinelli, G. and Orlandi, G. Convergence Improvement in the Adaptive Implementation of Pisarenko's Method, IEEE Trans., Vol. ASSP-35, No. 4, April 1987.

96. Ardalan, S.H. Floating-Point Error Analysis of Recursive Least-Squares and Least-Mean-Squares Adaptive Filters, IEEE Trans., Vol. CAS-33, No.12, December 1986.

97. Ardalan, S.H. and Alexander, S.T. Finite Wordlength Analysis of the Recursive Least Squares Algorithm, Proceedings Eighteenth Annual Asilomar Conference, Circuits, Syst. Comput., Monterey, CA, November 1984.

98. Scales, L.E. Introduction to Non-linear Optimization, Macmillan, London, 1985.

APPENDICES

APPENDIX 1

Recursive Least Squares Algorithm (RLS)

Consider the weighted least squares optimization problem,

$$\text{minimize} \sum_{i=1}^{K} \alpha^{K-i} (d_i - W_{K+1}^t X_i)^2 \tag{1.1}$$

The solution to the problem (1.1) is,

$$W_{K+1} = R_K^{-1} P_K \tag{1.2}$$

where,

$$R_K = \sum_{i=1}^{K} \alpha^{K-i} X_i X_i^t \tag{1.3}$$

$$P_K = \sum_{i=1}^{K} \alpha^{K-i} d_i X_i \tag{1.4}$$

It follows directly from (1.3) and (1.4) that R_K and P_K obey the recursions,

$$R_K = \alpha R_{K-1} + X_K X_K^t \tag{1.5}$$

$$P_K = \alpha P_{K-1} + d_K X_K \tag{1.6}$$

Application of matrix inversion lemma to (1.5) yields [81],

$$R_K^{-1} = \frac{1}{\alpha}\left[R_{K-1}^{-1} - \frac{R_{K-1}^{-1} X_K X_K^t R_{K-1}^{-1}}{\alpha + X_K^t R_{K-1}^{-1} X_K} \right] \qquad (1.7)$$

From (1.2), we have

$$R_K W_{K+1} = P_K \qquad (1.8)$$

Substitute (1.8) into (1.6) to obtain,

$$R_K W_{K+1} = \alpha R_{K-1} W_K + d_K X_K \qquad (1.9)$$

Substituting (1.5) into (1.9) yields,

$$R_K W_{K+1} = (R_K - X_K X_K^t)\, W_K + d_K X_K \qquad (1.10)$$

Multiplication of both sides of (1.10) by R_K^{-1} results in,

$$W_{K+1} = W_K - R_K^{-1}\, X_K X_K^t\, W_K + R_K^{-1}\, d_K X_K \qquad (1.11)$$

since $R_K^{-1}\, R_K = I$. Using relations $y_K = W_K^t X_K$ and $e_K = d_K - y_K$ in (1.11) yields,

$$W_{K+1} = W_K + R_K^{-1}\, X_K e_K \qquad (1.12)$$

which is the RLS algorithm.

In order to ensure that R_K^{-1}, given by (1.7), be a positive definite matrix for $i < L$, R_K^{-1} is initialized as

$$R_1^{-1} = \delta I \tag{1.13}$$

where, δ is a positive quantity, eg. 100, and I is the identity matrix.

The exponential weighting α^{K-i} is included to enable the adaptive estimators to track the data non-stationarities. Choosing $\alpha = 1$ corresponds to equally weighted data samples. In this case the adaptive estimators for R_K and P_K are expressed by,

$$R_K = \frac{1}{K}\sum_{i=1}^{K} X_i X_i^t \tag{1.14}$$

$$P_K = \frac{1}{K}\sum_{i=1}^{K} d_i X_i \tag{1.15}$$

Note that the estimators (1.3) and (1.4) result in biased estimates of R and P. These estimators can be modified to yield unbiased estimates by scaling them in accord with the formulae,

$$R_K = \frac{1-\alpha}{1-\alpha^i} \sum_{i=1}^{K} \alpha^{K-i} X_i X_i^t \tag{1.16}$$

$$P_K = \frac{1-\alpha}{1-\alpha^i} \sum_{i=1}^{K} \alpha^{K-i} d_i X_i \tag{1.17}$$

APPENDIX 2

The Lattice Gradient Algorithm

Consider the cost function for stage, i+1, with respect to both FD and BD residual energies,

$$J^{i+1} = E[(b^i_{K-1})^2] + E[(f^i_K)^2] - 4k^{i+1} E[(b^i_{K-1})(f^i_K)]$$

$$+ (k^{i+1})^2 \left[E[(b^i_{K-1})^2] + E[(b^i_{K-1})^2]\right] \qquad (2.1)$$

Taking the instantaneous gradient of (2.1) with respect to k^{i+1}_K yields,

$$\nabla J^{i+1}_K = -2f^i_K b^i_{K-1} + k^{i+1}_K [(f^i_K)^2 + (b^i_{K-1})^2] \qquad (2.2)$$

Rearranging (2.2) we obtain,

$$\nabla J^{i+1}_K = -f^i_K(2b^i_{K-1} - k^{i+1}_K f^i_K) + k^{i+1}_K (b^i_{K-1})^2$$

$$= -f_K(b^i_{K-1} + (b^i_{K-1} - k^{i+1}_K f^i_K)) + k^{i+1}_K (b^i_{K-1})^2 \qquad (2.3)$$

Using the equation for BD residuals in (2.3) results in

$$\nabla J^{i+1}_K = -f^i_K b^i_{K-1} - f^i_K b^{i+1}_K + k^{i+1}_K (b^i_{K-1})^2 \qquad (2.4)$$

Rearrange (2.4) to obtain,

$$\nabla J_K^{i+1} = -b_{K-1}^{i} \, (f_K^{i} - k_K^{i+1} \, b_{K-1}^{i}) - f_K^{i} \, b_K^{i+1} \qquad (2.5)$$

Using the equation for FD residuals in (2.5) yields the simplified equation for the instantaneous gradient for (i+1) -th stage,

$$\nabla J_K^{i+1} = -(b_{K-1}^{i} \, f_K^{i+1} + f_K^{i} \, b_K^{i+1}) \qquad (2.6)$$

To prove that the step size parameter a^{i+1} is inversely proportional to the sum of the FD and BD prediction error power, take the gradient of J^{i+1} with respect to a^{i+1} and equate it to zero which results in

$$a^{i+1} = \frac{1}{E[(b_{K-1}^{i})^2] + E[(f_K^{i})^2]} = \frac{1}{p^i} \qquad (2.7)$$

Because p^i is not known apriori, its estimate $\tilde{p}_K^i$ is used which is calculated recursively as data becomes available. A suitable estimator is the one given by (2.74).

APPENDIX 3

Convergence of the Method of Steepest Descent with Exact Linear Search

Assume that R is a symmetric positive definite matrix with M and m being its maximum and minimum eigenvalues. It follows that for any vector $V \neq 0$ (an eigenvector of R) the following inequalities are satisfied [14, pp. 11-45].

$$m|V|^2 \leqslant V^tRV \leqslant M|V|^2 \tag{3.1}$$

$$\frac{|V|^2}{M} \leqslant V^tR^{-1}V \leqslant \frac{|V|^2}{m} \tag{3.2}$$

$$m \leqslant \frac{V^tRV}{|V|^2} \leqslant M \tag{3.3}$$

$$\frac{1}{M} \leqslant \frac{|V|^2}{V^tRV} \leqslant \frac{1}{m} \tag{3.4}$$

$$1 \leqslant \frac{V^tRV}{|V|^2}\,\frac{V^tR^{-1}V}{|V|^2} \leqslant \frac{(M+m)^2}{4mM} \tag{3.5}$$

Consider the quadratic function,

$$J(W) = W^t RW - P^t W + c \qquad (3.6)$$

where, R is a symmetric positive definite matrix. The function J(W) has a unique minimum which is the solution of

$$\nabla J(W) = RW - P = 0 \qquad (3.7)$$

which is accordingly given by

$$W^* = R^{-1}P \qquad (3.8)$$

or

$$RW^* = P \qquad (3.9)$$

Define the residual vector (or error vector) of J(W) at W by,

$$g = - \nabla J(W) = P - RW \qquad (3.10)$$

Substitute (3.9) into (3.10), then

$$g = R(W^* - W) \qquad (3.11)$$

Defining the vector q by

$$q = W^* - W = R^{-1}g \qquad (3.12)$$

and using the Taylor expansion formula yields,

$$J(W+q) = J(W) - 2g^t q + q^t Rq \qquad (3.13)$$

It follows,

$$J(W) - J(W+W^*-W) = (W-W^*)^t R(W-W^*) \qquad (3.14)$$

or

$$J(W) - J(W^*) = g^t R^{-1} g \qquad (3.15)$$

Referring to the inequalities (3.1) and (3.2), we have,

$$m|W - W^*|^2 \leqslant J(W) - J(W^*) \leqslant M|W - W^*|^2 \qquad (3.16)$$

$$\frac{|g|^2}{M} \leqslant J(W) - J(W^*) \leqslant \frac{|g|^2}{m} \qquad (3.17)$$

where, M and m are the maximum and the minimum eigenvalues of R.

Now, with g=q, the function J(W+ag) can be expressed as,

$$J(W+ag) = J(W) - 2ag^t g + a^2 g^t Rg \qquad (3.18)$$

Equating the gradient of (3.18) with respect to a to zero yields,

$$a = g^t g/g^t Rg \qquad (3.19)$$

Substitute for a from (3.19) into (3.18) to obtain,

$$J(W) - J(W + ag) = \frac{|g|^4}{g^t R g} \tag{3.20}$$

Combining (3.20) with (3.15) yields,

$$J(W) - J(W + ag) \leqslant \frac{|g|^4}{(g^t R g)\ (g^t R^{-1} g)} (J(W) - J(W^*)) \tag{3.21}$$

Rearranging (3.21) we have,

$$J(W + ag) - J(W^*) \leqslant l\,(J(W) - J(W^*)) \tag{3.22}$$

where,

$$l = 1 - \frac{|g|^4}{(g^t R g)\ (g^t R^{-1} g)} \tag{3.23}$$

Substitute from (3.5) into (3.23) to obtain,

$$l = 1 - \frac{4Mm}{(M+m)} \tag{3.24}$$

$$l = \frac{(M - m)^2}{(M + m)^2} \tag{3.25}$$

With regard to the above analysis, it can be stated that after each ELS the value of the quadratic function is closer to its optimal value by a factor l . The above analysis can be used to express the value of the quadratic function after the (K+1)-th iteration in terms of its initial value and its optimum value. So that,

$$J(W_{K+1}) - J(W^*) \leqslant l^K \, (J(W_1) - J(W^*)) \qquad (3.26)$$

where, $J(W_1)$ is the initial value of the quadratic function. Since $l < 1$ it follows that $J(W_K) \dashrightarrow J(W^*)$ linearly with constant l . Also, $J(W_K)$ is a quadratic function of W_K. Therefore, $W_K \dashrightarrow W^*$ linearly with a constant $l^{\frac{1}{2}}$.

APPENDIX 4

CONJUGATE GRADIENTS METHOD

Conjugacy and Linear Independence

A set of vectors q_K are said to be mutually conjugate with respect to a positive definite matrix R if and only if R-orthogonality condition,

$$q_K^t R q_j = 0 \qquad K \neq j \tag{4.1}$$

is satisfied. For K = j, $q_K^t R q_j$ is strictly positive if $q_K \neq 0$ for all K.

A set of conjugate vectors are linearly independent and there can be no more than L vectors mutually conjugate with respect to any L by L matrix R. This useful property of mutually conjugate search vectors implies that any point in a L-dimensional space can be reached by a suitable linear combination of at most L mutually conjugate search vectors.

Quadratic Termination

Consider the equation,

$$W_{K+1} = W_K + a_K q_K \tag{4.2}$$

where, a_K is the exact step to the minimum along q_K.

Define

$$\delta W_K = W_{K+1} - W_K = a_K q_K \qquad (4.3)$$

$$\delta g_K = g_{K+1} - g_K \qquad (4.4)$$

where, g_K is the gradient of the quadratic function with respect to W_K. It follows,

$$R\,\delta W_K = \delta g_K \qquad (4.5)$$

The gradient g_K is related to g_{K+1} by,

$$g_K = g_{j+1} + \sum_{i=j+1}^{K-1} \delta g_i \qquad j = 0, K-1 \qquad (4.6)$$

Multiply (4.6) by q_j^t to obtain,

$$q_j^t g_K = q_j^t g_{j+1} + \sum_{i=j+1}^{K-1} q_j^t \,\delta g_i \qquad j = 0, K-1 \qquad (4.7)$$

Note that ELS implies that,

$$\frac{\delta}{\delta a_k} J(W_K + a_K q_k) = 0$$

$$g(W_K + a_K q_k) \frac{\delta}{\delta a_k} (W_K + a_k q_K) = 0$$

$$g_{K+1}^t \, q_K = 0 \qquad (4.8)$$

As a result of (4.8), the first term on the right hand side of (4.7) is zero. Using (4.5) and (4.3) in (4.7) yields,

$$q_j^t g_K = \sum_{i=j+1}^{K-1} q_j^t R\, \delta w_i$$

$$= \sum_{i=j+1}^{K-1} a_i q_j^t R q_i$$

$$= 0 \quad , \qquad j = 0,\ K-1 \tag{4.9}$$

The equality (4.9) follows from the conjugacy property (4.1).

Set $K = L$, then

$$q_j^t g_L = 0 \ , \quad j = 0 \ , \ L - 1 \tag{4.10}$$

If the gradient vector $g_L \neq 0$, then it must be orthogonal to all q_j , $j = 0$, $K - 1$. But q_j , $j = 0$, $K - 1$ are linearly independent and consequently (4.10) implies that there exists $L + 1$ linearly independent vectors in an L dimensional space. But this is not possible and (4.10) proves that $g_L = 0$, which is proof of quadratic termination due to ELS and linearly independent direction vectors.

The Construction of Conjugate Search Vectors

A set of mutually conjugate directions can be obtained by taking q_0 as the steepest descent direction and computing each subsequent direction as a linear combination of g_K and the previous search directions ie.

$$q_K = + g_K + \sum_{j=0}^{K-1} b_{Kj} q_j \tag{4.11}$$

As a result of (4.11), g_K is a linear combination of q_0 , q_1 , ... q_K , and

$$g_K^t g_i = 0 \qquad i < K \tag{4.12}$$

Also, q_K can be constructed to be conjugate to q_0 , ... q_{K-1} as follows. Multiply (4.11) by $q_i^t R$ and use the conjugacy condition (4.1) and the relationship

$\delta g_K = R(W_{K+1} - W_K) = a_k R q_k$, to obtain

$$q_i^t R q_K = + q_i^t R g_k + \sum_{j=0}^{K-1} b_{Kj} q_i^t R q_j$$

$$= + \frac{1}{a_K} (g_{i+1} - g_i)^t g_K + b_{Ki} q_i^t R q_i$$

for $i = 0$, ... $K-1$. (4.13)

The equation (4.12) implies that the first term on the right hand side of (4.13) vanishes for $i < K-1$. Therefore, to make q_K conjugate to q_i for $i < K-1$, simply choose b_{Ki} equal to zero, which results in one non zero coefficient $b_{K,K-1}$ (denote $b_{K,K-1} = b_{K-1}$). It is neccessary to obtain a value for b_{K-1} to ensure that q_K be conjugate to q_{K-1} . Multiply (4.11) by δg_{K-1} and apply the orthogonality condition $\delta g^t_{K-1} q_K = 0$ to obtain,

$$0 = \delta g^t_{K-1} g_K + b_{K-1} \delta g^t_{K-1} q_{K-1} \qquad (4.14)$$

or,

$$b_{K-1} = \frac{\delta g^t_{K-1} g_K}{\delta g^t_{K-1} q_{K-1}} \qquad (4.15)$$

Therefore, q_K can be written as,

$$q_K = g_K + b_{K-1} q_{K-1} \qquad (4.16)$$

Note that there are various possible choices for b_K (which can be obtained by using the orthogonality of the gradient vectors and the definition of the direction vectors) given by

$$b_{K-1} = \frac{\delta g^t_{K-1} g_K}{|g_{K-1}|^2} \qquad (4.17)$$

$$b_{K-1} = \frac{|g_K|^2}{|g_{K-1}|^2} \qquad (4.18)$$

For a detailed explanation of constructing conjugate search vectors see [13, 14].

Convergence of CGM

In theory the CGM is L-step convergent if exact arithmetic is used. However, in practice rounding errors cause the computed directions to lose conjugacy, and CGM behaves like the ELS method requiring a large number of iterations to converge. Furthermore, in the case of an ill conditioned system, CGM requires more than L iterations to converge, eg. 2L to 3L iterations.

To ensure the L-step convergence property of CGM, a restarting strategy must be used in order to preserve the conjugacy of the direction vectors q_K [13, 14].

AUTHOR'S PUBLICATIONS

1. Yaminysharif, M. and Durrani, T.S. Adaptive Signal Processing Using a Modified Gradient Estimation Technique, IEE Colloquium on Adaptive Filters, London, No. 1985/76, September 1985.

2. Yaminysharif, M. and Durrani, T.S. Adaptive Signal Processing Using a Modified Gradient Estimation Technique, ICASSP 1986, Tokyo, Vol. 4, April 1986.

ADAPTIVE SIGNAL PROCESSING USING A MODIFIED GRADIENT ESTIMATION TECHNIQUE

M Yaminysharif and T S Durrani

Abstract

In this paper an algorithm based on the gradient descent approach is proposed for adaptive signal processing. The proposed algorithm has the advantages of low computational overheads, and fast convergence properties - very close to those of the Conjugate-Gradient-Method (CGM). Besides it does not require an adhoc choice of step length and is applicable to both single channel and multichannel data. In this paper extensive computational results are presented to illustrate the algorithm's performance in terms of convergence properties, and its applications in adaptive noise cancelling, array processing, harmonic retrieval and eigen spectral analysis. The aspects of computational complexity of the proposed algorithm are discussed when used in different applications of adaptive filtering and are compared with alternative techniques.

1 Introduction

Conventional techniques for adaptive signal processing have primarily evolved from the Least-Mean-Square (LMS) algorithm or its variants, and extensive studies have been conducted on its convergence characteristics and performance properties [see Ref 1 for a recent comprehensive review]. The most attractive feature of the LMS algorithm is its simple computational requirements. On the other hand, the relatively slow convergence factor (step length) represents a handicap which has to be taken into account when considering high speed applications. Thus in applications where rapid convergence and speed are important, Recursive-Least-Squares (RLS) algorithms are gaining favour. These techniques though more complex than LMS algorithms, in general offer faster convergence.

Recently a special class of algorithms called the Conjugate Gradient (CG) algorithms have been proposed for adaptive filtering [2]. For a modern survey of these algorithms, see [3]. Although these algorithms lead to optimal solutions in the mean square sense, they involve significant computational overheads and, as such, have rarely found applications in practice, particularly for real time processing.

In this paper we propose an algorithm based on a gradient descent approach which is easy to implement for both single-channel and multi-channel data. In the next section a development of this Gradient Descent Based Algorithm (GDBA) is outlined and its convergence properties are discussed. In section (3.1) the application of CGM and GDBA to adaptive noise cancelling is discussed and the simulation results are presented to compare their relative performance. In section (3.2) two approaches, namely LMS and GDBA to the broad band adaptive array processing are discussed and their relative performance is compared by means of simulated examples. In sections (3.3) and (3.4) the application of GDBA to the adaptive implementation of Pisarenko's harmonic retrieval method and eigenfilter directional spectral estimation is discussed and simulation results are presented to illustrate its performance and convergence behaviour.

M Yaminysharif and T S Durrani are with the Department of Electronic and Electrical Engineering, University of Strathclyde, Glasgow, Scotland.

2 Problem Formulation

Three types of minimization problems frequently arising in adaptive signal processing theory are to minimise the following cost functions:

$$\text{minimize } J = \underline{W}^t R \underline{W} - 2\,\underline{P}^t \underline{W} + C \qquad (1)$$

$$\text{minimize } J = \underline{W}^t R \underline{W} \qquad (2)$$

subject to some linear constraints

$$\text{minimize } J = \underline{W}^t R \underline{W} \qquad (3)$$

subject to the norm constraints

where: $\underline{W}$ = adaptive filter weight vector, C = constant, R = data covariance matrix, $\underline{P}$ = cross correlation vector between the reference and the desired signal, and t denotes transposition.

The conventional approach to these problems is the gradient descent technique leading to some sort of LMS or constrained LMS algorithm when the gradient is replaced by its instantaneous estimate. As pointed out earlier, these algorithms involve compromises based on choice of step length vs convergence rate and, as a result, the associated problems of misadjustment and speed arise. Different techniques such as RLS and CGM can be applied to these problems which yield substantial improvement in overall performance and the speed of convergence at the cost of increasing computational complexity.

However, in order to obtain rapid convergence at the expense of a modest increase in computational complexity, we propose the following algorithm which is based on the gradient descent approach. For K = 1, 2, ... compute,

$$\begin{aligned}
R_K &= (R_{K-1}*(k-1) + \underline{X}_K \underline{X}_K^t)/K \\
\underline{P}_K &= (\underline{P}_{K-1}*(K-1) + d_K \underline{X}_K)/K \\
\underline{r}_K &= R_K \underline{W}_K - \underline{P}_K \\
a_K &= \underline{r}_K^t \underline{r}_K \,/\, \underline{r}_K^t R_K \underline{r}_K \\
\underline{W}_{K+1} &= \underline{W}_K - a_K \underline{r}_K
\end{aligned} \qquad (4)$$

where: $\underline{X}$ = vector of the tap voltages and d = the desired signal. For minimization problems (2) and (3) the algorithm is modified since $\underline{r}_K = R_K \underline{W}_K$. The attractive feature of this algorithm is the recursive estimation of its parameters in particular the convergence factor a_K which satisfies the optimality condition for an optimum step sequence [W A Gardner 1].

Although the above modified gradient algorithm requires more computations than the LMS algorithm (this is not a great concern with today's computing power cost), it is a rapidly converging algorithm which is the essential requirement for a wide range of practical applications. The performance of the algorithm has been studied when used in a range of applications of adaptive filtering. Extensive simulation studies have illustrated a substantial improvement in the speed of convergence and reduced final misadjustment over the conventional LMS algorithm. Figure 1 illustrates the variation of the convergence factor a_K of the proposed algorithm, with respect to the inverse of the largest and the smallest eigenvalues of the covariance matrix. This illustrates that a_K satisfies the optimal boundary condition for an optimum step sequence [1].

3 Application of the Proposed Algorithm

3.1 Adaptive noise cancelling (ANC):

The application of the LMS algorithm to ANC yields the very well known update formula of Widrow-Hoff. In order to apply CGM to ANC, computations are carried out in two steps. The first step is to compute the estimate of the gradient r_K for $K = 1, 2, \ldots$. The second step is to use CGM and compute the optimum solution W_K^0. The procedure is outlined below:

I. Compute the estimate of the gradient $\underline{r}_K = R_K\underline{W}_K - \underline{P}_K$.

II. Select an arbitrary initial estimate of $\underline{W}_K$.

III. Use CGM to solve $\underline{r}_K$ and obtain $\underline{W}_K^0$.

Minimization criterion using GDBA is straight forward and results in an exact recursive formulation (4). Figures 2 and 3 represent the learning curves obtained by CGM and GDBA for different values of SNR. In these simulations the primary input was a sine wave of amplitude five representing the signal, corrupted by Gaussian white noise. The reference signal was a first order autoregressive white noise. In the case of CGM the r_K was estimated by the same recursive equations as listed in (4). An arbitrary initial estimate of $W_1 = (0{\cdot}5, 0{\cdot}5, \ldots)$ was chosen for all these simulations and the adaptive filter had five taps. These results clearly show the close performance of CGM and GDBA with higher final misadjustment of CGM for low SNR conditions. Thus it is obvious that a great reduction in computational complexity can be achieved with performnace similar to CGM when using GDBA.

3.2 Adaptive array processing:

The performance of the LMS algorithm and GDBA is illustrated in Figures 4, 5 and 6. A simulation was set up with a linear array of five elements, each being processed through a filter of five tap weights which were updated according to the LMS algorithm (with convergence factor 0·00001 set arbitrary) and GDBA. The environment had three point noise sources of power 1, 10 and 10 located at 0, 30 and 45 degrees respectively and noise per sensor was set at 0 dB. The look direction was assumed to be 0 degree (normal to array). There was no correlation between the desired signal (located at 0 degree) and two strong interferences (located at 30 and 45 degrees). Figures 5 and 6 show the spectral output obtained (128 data samples) from the LMS and GDB algorithms. It is clear that GDBA has completely suppressed the two strong interferences unlike the LMS algorithm.

3.3 Adaptive implementation of Pisarenko's harmonic retrieval method:

The proposed algorithm was tested for the computation of spectra of sinusoids by implementing Pisarenko's technique as an adaptive GDBA. The result is shown in Figure 7. Note that the algorithm is able to pick up the spectral peaks after only N = 100 data points even for the very poor SNR of = 2 dB.

3.4 Eigenfilter approach to array spectral analysis:

The algorithm has been applied for estimating the spatial spectra from data received for an array of narrow band sources. The performance of the algorithm has been tested for both correlated and uncorrelated sources. Here a representative result is included for spectral estimates (Figure 8) for sources located at 10, 45 and 60 degrees respectively. Sources at 45 and 60 degrees are correlated with the mutual correlation coefficient, 0·8 the SNR for the sources are 5, 10 and 5 dB. Note that the algorithm allows a clear and unambiguous estimation of the spatial spectra even for as short a data length as N = 128.

4 Conclusions

In this paper we have presented an algorithm for on-line adaptive filtering. The main objectives of the algorithm were to increase the speed of convergence, to eliminate the requirement for an adhoc choice of convergence factors and to reduce the final misadjustment. Extensive computer simulations for different applications of adaptive filtering verified the excellent speed, accuracy and convergence properties of the proposed algorithm. However, the cost associated with these achievements is the increase in computational requirements which is modestly increased in comparison with that of the LMS algorithm, but significantly reduced when compared with those of alternative techniques such as CGM and the techniques requiring the inverse of the covariance matrix.

5 Acknowledgement

The authors wish to thank the Science and Engineering Research Council for supporting this work.

6 References

[1] GARDNER W A: "Learning characteristics of stochastic-gradient-descent algorithm: a general study, analysis and critique", Signal and Image Processing Laboratory, Department of Electrical and Computer Engineering, University of California, Davis 95616, USA.

[2] NEHORAI A and MORF M: "A new derivation for fast recursive least squares and Levinson algorithms by the conjugate direction method", Information Systems Laboratory, Electrical Engineering Department, Stanford University, Stanford, CA 94305, USA.

[3] HESTENES M: "Conjugate direction methods in optimization", Springer-Verlag, NY, 1980.

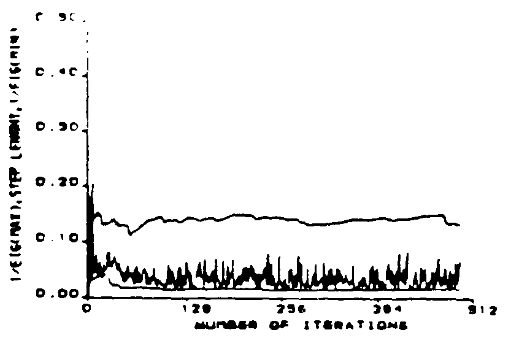


FIG1:RELATIVE BEHAVIOUR OF THE STEP LENGTH AND INVERSE OF LARGEST AND THE SMALLEST EIGENVALUES OF R(K).

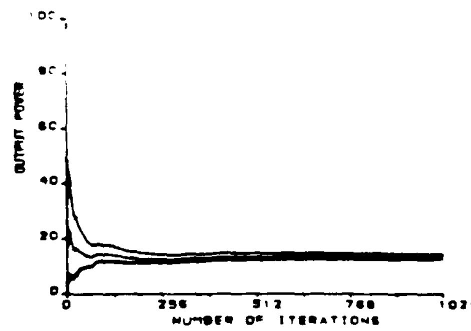


FIG.2:ADAPTIVE NOISE CANCELLING , CGM LEARNING CURVES , SNR=11,-1,-6,-10 DB

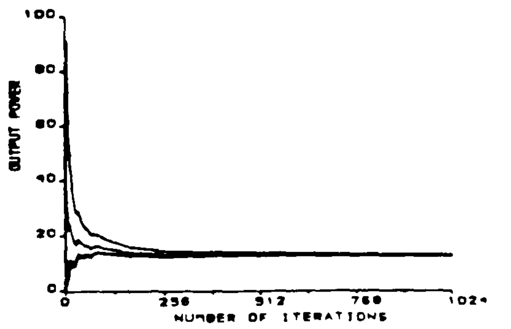


FIG.3:ADAPTIVE NOISE CANCELLING , SODA LEARNING CURVES , SNR=11,-1,-6,-10 DB

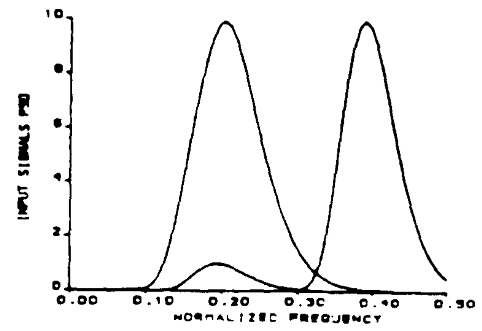


FIG4:INPUT SIGNALS (SIGNAL POWER=1 AT 0 DEG.INTERFERENCE POWER=10 AT 30 DEG.INTERFERENCE POWER=10 AT 45 DEG.AND SENSOR NOISE=1).

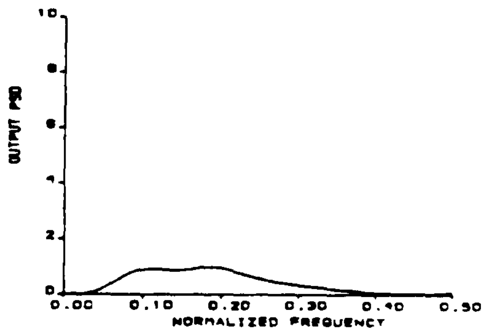


FIG5:ADAPTIVE ARRAY PROCESSING ,LMS

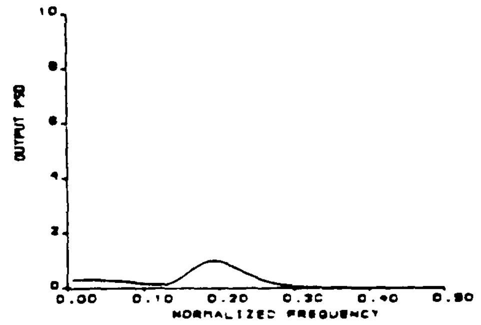


FIG6:ADAPTIVE ARRAY PROCESSING ,SODA

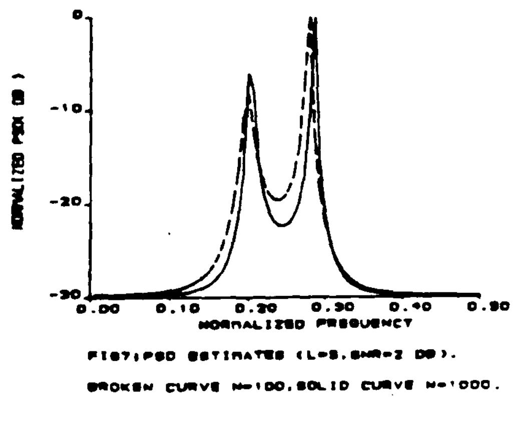


FIG7:PSD ESTIMATES (L=5,SNR=2 DB). BROKEN CURVE N=100,SOLID CURVE N=1000.

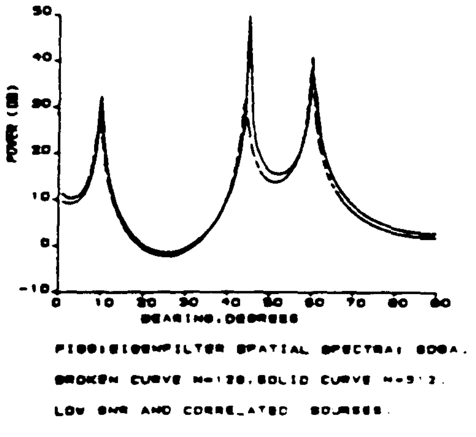


FIG8:EIGENFILTER SPATIAL SPECTRA: SODA. BROKEN CURVE N=128,SOLID CURVE N=512. LOW SNR AND CORRELATED SOURCES.

ADAPTIVE SIGNAL PROCESSING USING A MODIFIED GRADIENT ESTIMATION TECHNIQUE

M Yaminysharif and T S Durrani

Dept of Electronic & Electrical Engineering
University of Strathclyde,
Glasgow G1 1XW Scotland UK

ABSTRACT

An algorithm based on the gradient descent approach is proposed for adaptive signal processing. The algorithm outperforms the least mean square algorithm (LMS) in terms of convergence speed and misadjustment noise, and meets the performance characteristics of the conjugate gradient (CG) and recursive least squares (RLS) methods without the attendant computational complexity. It is simple to implement and lends itself to real time processing. The algorithm can be implemented using lattice or tap-delay line structures. In this paper extensive computational results are presented to illustrate the algorithm performance in terms of convergence properties, and its applications in adaptive noise cancelling and adaptive spectral analysis.

1 INTRODUCTION

In the main, adaptive signal processing is performed using an FIR filter with coefficients that are updated to achieve some prescribed performance criterion.

Many of the algorithms proposed for use in adaptive signal processing are based on the gradient descent approach and can be expressed in the following form:

$$W_{K+1} = W_K + a_K(P_K - R_K W_K) \qquad (1)$$

K = 1, 2, 3 ... where W_K = filter weight vector, a_K = step length, P_K = estimate of the cross correlation vector P, R_K = estimate of the data covariance matrix R, $P = E(d_K X_K)$, $R = E(X_K X_K^t)$, d_K = desired signal, $\underline{X}_K$ = vector of the tap voltage, and data vector, $E(\cdot)$ denotes the expectation, and t denotes transposition. The design choice of a_K, P_K, and R_K depends on the particular application and involves a trade-off between the speed of convergence and computational complexity. The most computationally attractive choice a_K = constant, $P_K = d_K X_K$, and $R_K = X_K X_K^t$ leads to the well known and widely used LMS algorithm. On the other hand, when the speed of convergence is of primary importance more consistent estimates of P and R are required along with a variable step length a_K which must satisfy an optimal bound in order to maximize the convergence speed. In this case, the conjugate gradient methods (CGM) [1] can be utilised for maximizing the speed of convergence.

To implement these algorithms the computation is generally carried out in two steps due to the necessity of generating mutually conjugate vectors. The first step is to compute the estimates P_K and R_K and the second step is to use CGM and compute the weight vector W_{K+1}. This type of implementation is a drawback for on-line signal processing due to increased hardware complexity associated with the implementation and the attendant increased computing burden.

In this paper, we propose an algorithm based on (1) which is simple to implement for both single channel and multichannel data. In the next section, a development of this gradient descent based algorithm (GDBA) is outlined and its convergence properties are discussed. In Section 3.1, GDBA is applied to adaptive noise cancelling and the simulation results are presented to compare its performance with CGM. Simulation results are also presented when the algorithm is implemented using a lattice structure, and are compared with the results obtained when a conventional lattice gradient algorithm is used. In Section 3.2, the application of GDBA to the adaptive spectral analysis, single channel Pisarenko's technique and multichannel directional spectral analysis is discussed, and simulation results are presented to illustrate its performance.

2 PROBLEM FORMULATION

Defining the following:

$$V_K = W_K - W_*$$

$$Z_K = P_K - R_K W_*$$

where W_* = optimum weight vector (for non-stationary case $W_* = W_{o,K}$ = the parameter vector which minimizes the cost function at time K), (1) can be expressed as:

$$V_{K+1} = \left(\prod_{i=1}^{K} f_i\right)\cdot V_1 + \sum_{j=1}^{K}\left(\prod_{m=j+1}^{K} f_m\right)\cdot a_j Z_j \qquad (2)$$

where $f_1 = (I - a_1 R_1)$, $K \gg 1$ and $f_m = I$ if $m > K$, I = identity matrix. In order to analyse the convergence behaviour of (1) using (2) one must find a sufficient condition on the sequences a_K, Z_K, and R_K for which the system output would converge. Extensive studies have been conducted on this issue based on the assumptions; stationary or non-stationary data, dependent or independent observations of P_K and R_K, see for instance [2,3]. In these papers, (1) has been analysed for a_K = constant, $P_K = d_K X_K$, and $R_K = X_K X_K^t$ to obtain a bound on the quadratic mean of $|V_K|$.

However, the most obvious and necessary condition for (2) to converge is:

$$0 < \| f_K \| < 1 \ , \ f_K = (I - a_K R_K) \qquad (3)$$

Equation (3) is not satisfied unless a_K satisfies an optimal bound which is strictly determined by the eigenvalue ratio (EVR) of the estimate of the convariance matrix R_K. Equation (3) would definitely be met if $a_K < \frac{1}{EV_{max}}$ of R_K, but to maximize the speed of convergence and at the same time to meet (3) let us consider two extreme situations:

a) If $EVR = \frac{EV_{max}}{EV_{min}} \ngtr 1$, then (3) is met if

$$0 < a_K \leqslant \frac{1}{EV_{min}}$$

b) If $EVR = \frac{EV_{max}}{EV_{min}} \gg 1$, then (3) is met if

$$0 < a_K \leqslant \frac{2}{EV_{max} + EV_{min}} < \frac{2}{EV_{max}} \qquad [4].$$

To maximize the convergence speed the lower bound (LB) must be kept well away from zero and a suitable and permissible LB would be $\frac{1}{EV_{max}} \leqslant a_K$.

Now for a_K = upper bound (UB), (2) can be analysed in order to obtain a bound for quadratic mean of $|V_K|$ utilizing the same approach as that of [2,3]. Hence, one can state that the actual bound on $E(|V_K|^2)$ would be smaller, since $a_K \leqslant$ UB.

Therefore, to maximize the convergence speed we propose the following recursion to update W_K:

$$\begin{aligned} R_K &= (R_{K-1}*(K-1) + X_K X_K^t)/K \\ P_K &= (P_{K-1}*(K-1) + d_K X_K)/K \\ r_K &= R_K W_K - P_K \qquad\qquad (4) \\ a_K &= r_K^t r_K / r_K^t R_K r_K \\ W_{K+1} &= W_K - a_K r_K \end{aligned}$$

The attractive feature of this algorithm is the recursive estimation of its parameters, in particular the step length a_K. Although this algorithm requires more computations than the LMS algorithm (this is not a great concern with today's computing power cost) it is a rapidly converging algorithm which is an essential requirement for a wide range of practical applications.

Figure 1 shows the evolution of the step length a_K of the proposed algorithm with respect to the inverse of the largest, smallest,

eigenvalues of R_K. Figure 1a shows the variation of a_K for EVR $\gg 1$, and Figure 1b illustrates this behaviour for EVR $\ngtr 1$.

3 APPLICATIONS

3.1 Adaptive Noise Cancelling (ANC)

Figures 2 and 3 represent the learning curves for ANC obtained by OGM and GDBA for different values of SNR. The primary input was a sine wave of amplitude five representing the signal, corrupted by Gaussian white noise. The reference signal was a first order autoregressive sequence. In the case of OGM, P_K and R_K were estimated by the same recursive equations as listed in (4). An arbitrarily small initial estimate of W_1 was chosen for all these simulations and the adaptive filter used was a tap delay line with five taps. These results clearly show the close performance of OGM and GDBA.

The algorithm (4) can also be applied to ANC when a noise canceller is implemented using lattice structure. This type of implementation has its advantages, since the lattice filter preprocesses its input signal (using a set of partial correlation coefficients, PCC, whose values are obtained by an adaptive algorithm) and supplies a set of orthogonal signals (backward residuals) to the combiner structure. Due to these orthogonalized signals, the weight on each tap of the combiner is given a step size which is inversely proportional to the signal power at the corresponding lattice output (this gives all taps the same rate of convergence) [5]. However the lattice filter outputs are a set of orthogonal signal samples only after complete convergence of the PCC. Thus there exists a degree of correlation between these signals which decreases as PCC approach their optimal values. Based on this discussion, a combined update recursion for the combiner weights would increase the initial rate of convergence, in particular if EVR $\gg$ 1.

Figure 4 represents the learning curves for ANC when implemented using a ten stage lattice filter, for the same desired signal (as Figures 2 and 3). The reference signal was a highly correlated first order autoregressive sequence (with correlation coefficient 0·8) resulting in EVR = 45. The solid curve represents the result obtained by a conventional lattice gradient algorithm [5] and the broken curve represents the result using (4). For the latter case, PCC were updated using a conventional recursion procedure [5] while the combiner weights were updated using (4). These results illustrate faster initial rate of convergence that can be achieved using (4). However, this is a preliminary result and related aspects of the algorithm are being investigated in more detail.

3.2 Adaptive Spectral Analysis

The proposed algorithm was tested for computation of spectra of sinusoids (adaptive implementation of Pisarenko's technique [6,7]), and estimating the spatial spectra from data received for an array of narrow band sources (eigenfilter directional spectral estimation) [8]. These techniques involve estimating the minimum eigenvalue of the data covariance matrix and the corresponding eigenvector. If (1) is convergent, then the expected value of the output power would be an estimate of EV_{min}. The algorithm (4) is applicable to these problems simply by estimating $P_K = E(Y_K{}^2)W_K$, and thus, $r_K = R_K W_K - E(Y_K{}^2)W_K$, where Y_K = output of the filter (use complex notation for eigenfilter problem). Figure 5 represents the spectral estimates for two sinusoids of normalized frequency 0·18 and 0·38 for poor SNR = 0 dB. Note that the algorithm is able to pick up the spectral peaks after only N = 100 data points which is a compatible result with those obtained using RLS and CGM [6,7]. Figure 6 represents a computer result for directional spectral estimates for sources located at 10, 45, and 60 degrees. Sources at 45 and 60 degrees are correlated with a mutual correlation coefficient 0·72, the SNR for the sources are 5, 10 and 5 dB. The unambiguous estimation of the spatial spectra even for as short a data length as N = 150 indicates rapid rate of convergence of the algorithm and its performance compatibility with RLS techniques [8].

4 CONCLUSION

In this paper we have presented an algorithm for on-line adaptive filtering. The main objectives of the algorithm were to increase the speed of convergence, to eliminate the requirement for adhoc choice of convergence factors and to reduce the final misadjustment. Extensive computer simulations for different applications of adaptive filter verified the excellent speed, accuracy and convergence properties of the proposed algorithm and its performance compatibility with RLS and CG techniques. To increase the initial rate of convergence the algorithm can be applied to the lattice structures with a trade-off of increased computational complexity.

5 ACKNOWLEDGEMENT

The authors wish to thank the UK Science and Engineering Research Council for supporting this work.

REFERENCES

[1] Hestenes M: "Conjugate direction methods in optimization", Springer-Verlag, NY, 1980.

[2] Farden D C: "Tracking properties of adaptive signal processing algorithms", IEEE Trans ASSP, Vol-29, No 3, June 1981.

[3] Macchi O and Eweda E: "Second order convergence analysis of stochastic adaptive linear filtering", IEEE Trans Aut Cont, Vol-28, No 1, Jan 1983.

[4] Gersho A: "Adaptive equalization of highly dispersive channels for data transmission", Bell Syst Tech J, Vol-48, pp 55-70, Jan 1969.

[5] Grant P M and Rutter M J: "Application of gradient adaptive lattice filters to channel equalisation", IEE Proc, Vol-131, Pt F, No 5, Aug 1984.

[6] Reddy V U et al: "Least squares lattice type algorithm for adaptive implementation of Pisarenko's harmonic retrieval method", IEEE Trans ASSP, Vol-30, No 3, June 1982.

[7] Chen H et al: "Adaptive spectral estimation by the conjugate gradient method", Proc ICASSP-85, Vol-1, pp 81-84, 1985.

[8] Durrani T S and Sharman K C: "Eigenfilter approaches to adaptive array processing", IEE Proc, Vol-130, Pts F and H, No 1, Feb 1983.

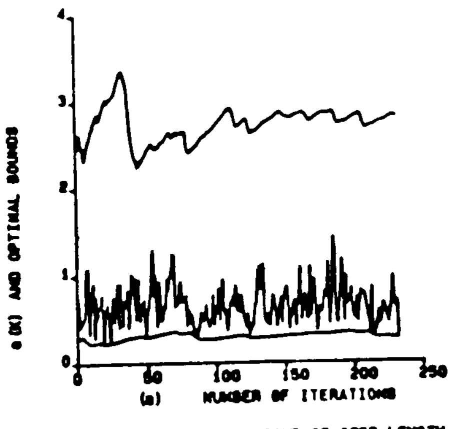


FIG1: RELATIVE BEHAVIOUR OF STEP LENGTH AND INVERSE OF THE EIGENVALUES OF R(K).

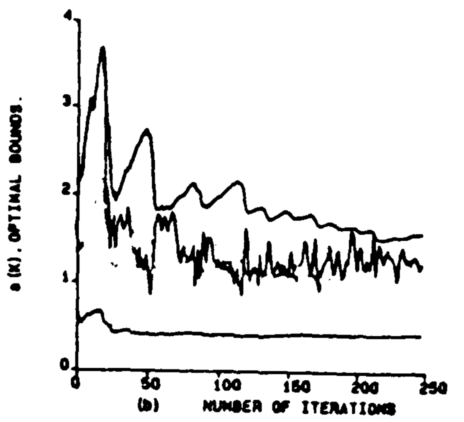


FIG1: RELATIVE BEHAVIUOR OF STEP LENGTH AND INVERSE OF THE EIGENVALUES OF R(K)

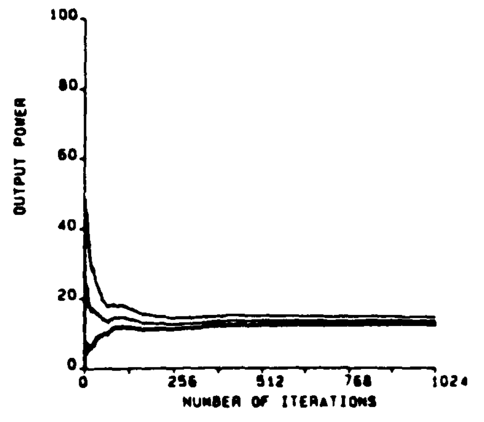


FIG.2: ADAPTIVE NOISE CANCELLING . CGM LEARNING CURVES . SNR=11, -1, -6, -10 DB

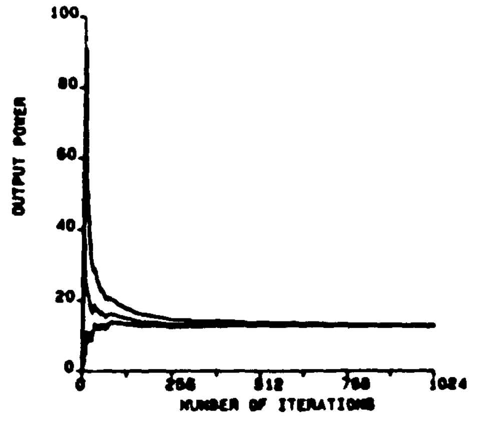


FIG.3: ADAPTIVE NOISE CANCELLING . SOGA LEARNING CURVES . SNR=11, -1, -6, -10 DB

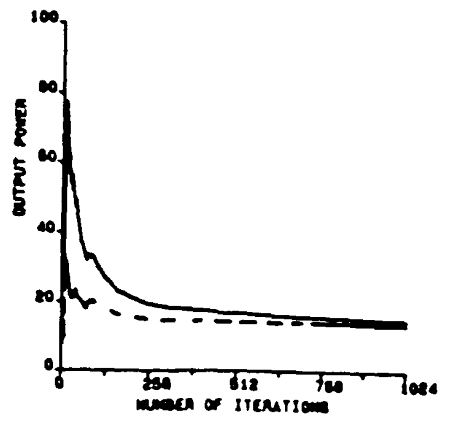


FIG.4: ANC LATTICE STRUCTURE SNR=6.0 DB BROKEN CURVE: SOGA . SOLID CURVE: CONV-LAT

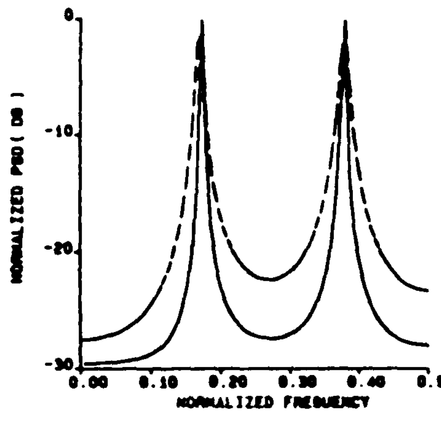


FIG5: PSD ESTIMATES (L=6, SNR=0 DB). BROKEN CURVE N=100, SOLID CURVE N=600.

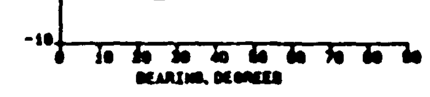


FIG6: EIGENFILTER SPATIAL SPECTRA: SOGA. BROKEN CURVE N=150, SOLID CURVE N=200.