NOVEL TECHNIQUES AND ARCHITECTURES FOR ADAPTIVE BEAMFORMING

By

THUA VAN HO, B.A.Sc, M.A.Sc

A Thesis

Submitted to the School of Graduate Studies

in Partial Fulfillment of the Requirements

for the Degree

Doctor of Philosophy

McMaster University

NOVEL TECHNIQUES AND ARCHITECTURES FOR ADAPTIVE BEAMFORMING

THUA VAN HO, B.A.Sc, M.A.Sc

DOCTOR OF PHILOSOPHY (1990) McMASTER UNIVERSITY
(Electrical Engineering) Hamilton, Ontario

TITLE: Novel Techniques and Architectures for Adaptive Beamforming

AUTHOR: Thua Van Ho, B.A.Sc (University of Toronto)

M.A.Sc (University of Toronto)

SUPERVISOR: Professor John Litva

NUMBER OF PAGES: xv, 183

To my parents

ABSTRACT

Recent progress in VLSI technology has created a major impact on digital signal processing, including array signal processing. Proposals have been made for using high throughput processors for digital adaptive beamforming in radar and communications systems applications. In this thesis, novel techniques and architectures for adaptive beamforming will be developed and presented. These are typified by the development of adaptive beamforming algorithms for planar arrays and by a self–calibration algorithm for antenna arrays. The emphasis however will be placed on modern adaptive beamforming techniques in which the adaptation is carried out by means of a triangular systolic array processor performing the QR decomposition.

Adaptive beamforming algorithms for a planar array or two–dimensional (2–D) adaptive beamforming algorithms, which are typified by the 2–D least–mean–squares (LMS) algorithm and 2–D Howells–Applebaum algorithm, are derived and presented. The concept of 2–D eigenbeams will be given to demonstrate the performance of the 2–D adaptive beamforming techniques. As well, the 2–D adaptive beamforming problem will be formulated in terms of the 1–D case with operation taking place along rows and columns of a planar array. The adaptive processor is then implemented by using a manifold of the least–squares triarray processors, which in the limit takes the form of a 3–D systolic array. It will be shown that the structure is capable of performing adaptation along the rows and columns of the 2–D array simultaneously.

One of the major challenges that face workers in array processing is overcoming the degradation in the output of the high performance algorithms due to errors in the calibration of the array. A new self–calibration technique for solving this difficult problem will be derived and presented herein. The algorithm is based on the use of iteration – whereby the calibration coefficients are refined through repetitive imposition of the calibration procedure. Its derivation is based on the eigen–based method and the least–squares norm minimization. It will be shown that the algorithm is capable of automatical estimating the angles–of–arrival (AOA) of the received signals and calibrating the array with a minimum phase and gain errors. Results obtained by using both simulation and measurement data will be given. In the case of the experimental results, the measurement setup is subjected to multipath scenarios.

ACKNOWLEDGEMENT

I would like to express my sincere thank to my supervisor, Dr. John Litva, for his excellent supervision and continuous support during the course of this thesis. Without his guidance and encouragement several breakthroughs in this thesis would be impossible to be achieved.

I would like to express my special thank to Dr. Simon Haykin for accepting me to the CRL at the first place, and from whom I have learned the most.

I also would like to express my gratitude to staff and students at the Communications Research Laboratory, McMaster University, for their friendship and help by any means during my years at the university.

Finally, I am deeply grateful and most indebted to my wife, Catherine, for her love, patience and support during the course of my study.

TABLE OF CONTENTS

LIST OF SYMBOLS

$\mathbf{X}$	matrix: upper case, bold face letter, e.g., $\mathbf{Q}(n)$.
$\mathbf{x}$	column vector: lower case, bold face letter, e.g., $\mathbf{y}(n)$.
x	element of a column vector $\mathbf{x}$ or a complex constant: lower case, italic letter, e.g., $\mathbf{x}(n) = [x_1(n), x_2(n), \ldots, x_M(n)]$.
X	function or subset of complex space: upper case letter, e.g., $\mathrm{H}(\theta)$: a function of variable θ.
X	complex scaling constant: upper case, italic letter, e.g., M: number of antenna elements.
α,μ	constant: lower case, Greek letter.

And also

θ	spatial angle (elevation direction)
ϕ	spatial angle (azimuth direction)
Φ	covariance matrix
$Tr(\Phi)$	trace of a matrix
$\mathbf{R}$	upper triangular matrix
$\mathbf{Q}$	unitary matrix
E[.]	expected value
L,M	number of elements (dimensions) in an antenna array
N	number of snap shots (n).
K	number of signals
d	inter–element spacing
λ	wavelength

LMS	least–mean–squares
SMI	sample matrix inversion
PE	processing element
AOA	angle–of–arrival
SNR	signal–to–noise ratio
INR	interference–to–noise ratio
SINR	signal–to–interference–and–noise ratio

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1. Introduction to Adaptive Beamforming Problem

The performance of conventional communications and radar antenna systems is usually degraded by undesired signals which intrude via the antenna's mainbeam, as well as its sidelobes. The undesired signals may consist of deliberately generated signals as an electronic counter measure (ECM), RF interference, radar clutter returns, and natural noise sources. Also, antenna motion, poor sitting conditions, multipath, and mutual coupling might further contribute to this degradation. Adaptive beamforming techniques offer possible solutions to these critical problems of interference and system antenna errors.

Adaptive beamforming technology has been actively discussed in the literature for at least two decades and is now, increasingly, finding applications in radar, sonar, and communications systems [1–5]. The reason of all of this interest lies in the ability of adaptive arrays to automatically steer nulls in the direction of interfering sources. Recently, with the rapid growth of VLSI technology, and particularly with the advent of systolic arrays, the use of VLSI array processors in adaptive digital beamforming has become a subject of considerable interest [6–8].

Adaptive beamforming can be carried out with narrowband signals or

broadband signals. In the case of narrowband adaptive beamforming, each channel of the antenna array is associated with an adaptive weight and the adaptive processor is just a linear combiner (Fig. 1.1). In the broadband case, each channel is associated with a tap–delay line and the adaptive processor takes the form of a transversal filter (Fig. 1.2). Nevertheless, from a mathematical point of view the two types of beamforming have similarities, although the broadband case is considerably more complex. Therefore, for simplicity of analysis, only narrowband signals will be considered in this thesis.

Consider the adaptive antenna array, which is illustrated in Fig. 1.1. Essentially, it is an array, consisting of M+1 antenna elements. The first M elements are called auxiliary elements and the signals at their ports are denoted by $x_1(n)$, $x_2(n)$, ..., $x_M(n)$. Prior to summation, the outputs of these elements are multiplied or, in other words, weighted by the parameters $w_1(n)$, $w_2(n)$, ..., $w_M(n)$. The last element of the array is used as a reference antenna. For the purpose of formulating the problem, it is called a primary element and is denoted by $y(n)$. The statement of the problem is as follows. Given a data matrix $\mathbf{X}(n)$ and a primary vector $\mathbf{y}(n)$, find the tap weight vector $\mathbf{w}(n)$ which minimizes the combined output signal $\mathbf{e}(n)$, where

$$\mathbf{e}(n) = \mathbf{y}(n) - \mathbf{X}(n)\mathbf{w}(n) \tag{1.1}$$

with

$$\mathbf{e}(n) = [e(1),\ e(2),\ ...,\ e(n)]^T \tag{1.2}$$

$$\mathbf{y}(n) = [y(1),\ y(2),\ ...,\ y(n)]^T \tag{1.3}$$

and

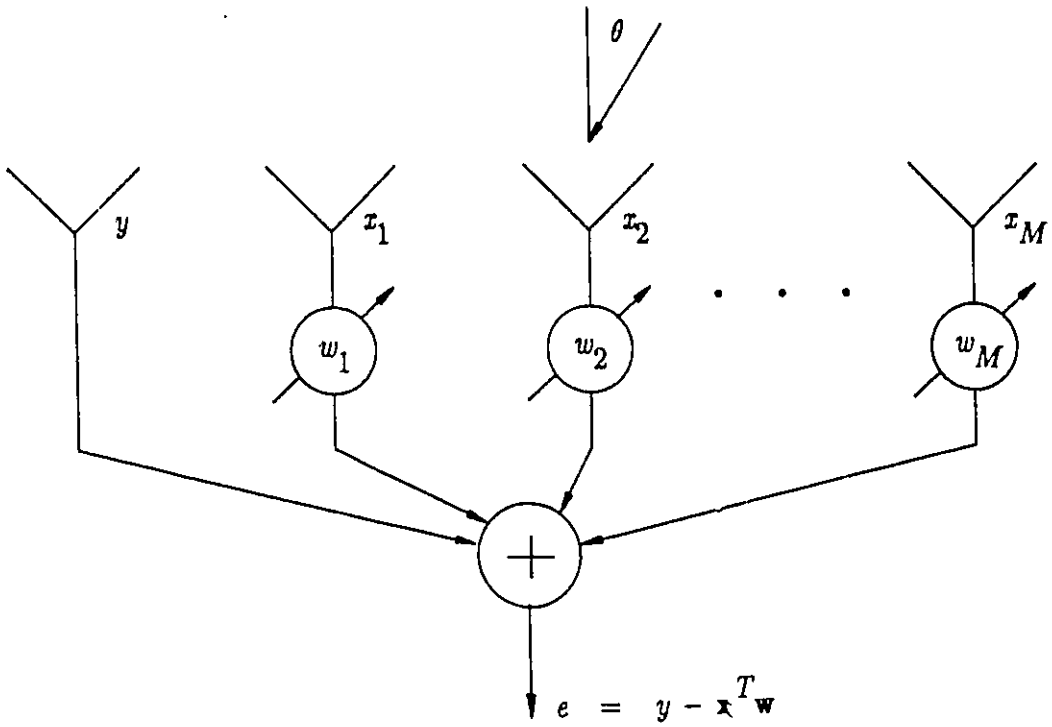


Fig. 1.1 Narrow band adaptive antenna

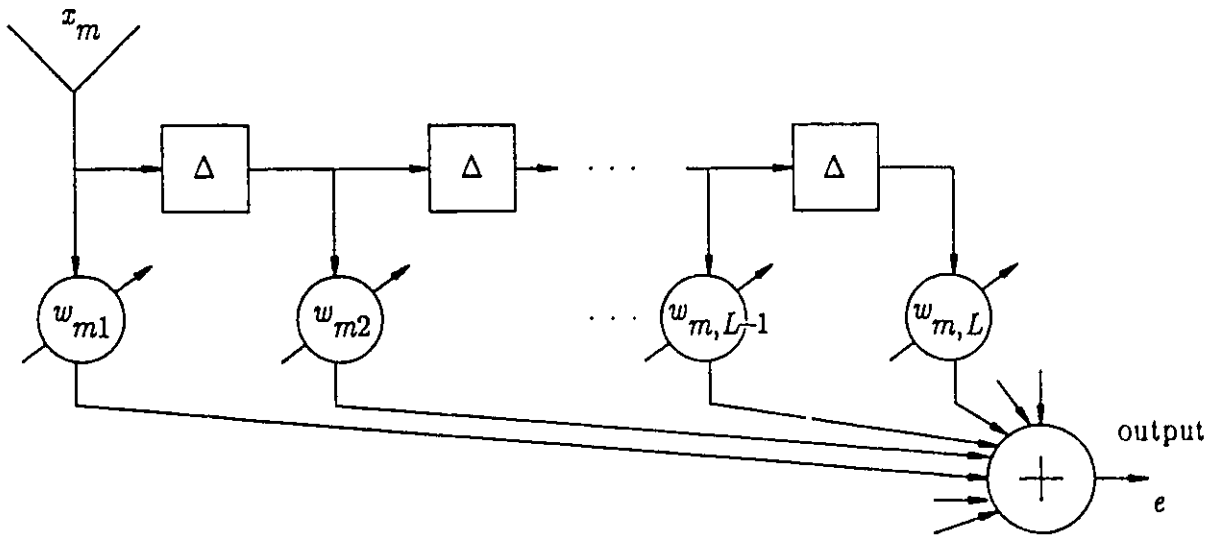


Fig. 1.2 Broad band adaptive antenna

$$\mathbf{X}(n) = \begin{bmatrix} x_1(1) & x_2(1) & \ldots & x_m(1) & \ldots & x_M(1) \\ x_1(2) & x_2(2) & \ldots & x_m(2) & \ldots & x_M(2) \\ \vdots & \vdots & & \vdots & & \vdots \\ x_1(n) & x_2(n) & \ldots & x_m(n) & \ldots & x_M(n) \end{bmatrix} \tag{1.4a}$$

$$= [\mathbf{x}_1(n),\ \mathbf{x}_2(n),\ \ldots,\ \mathbf{x}_M(n)] \tag{1.4b}$$

$$= \begin{bmatrix} \mathbf{x}^T(1) \\ \mathbf{x}^T(2) \\ \vdots \\ \mathbf{x}^T(n) \end{bmatrix} \tag{1.4c}$$

and

$$\mathbf{w}(n) = [w_1(n),\ w_2(n),\ \ldots,\ w_M(n)] \tag{1.5}$$

where the vectors $\mathbf{x}_m(n)$ and $\mathbf{x}^T(n)$ in Eqns. (1.4b) and (1.4c) are defined as

$$\mathbf{x}_m(n) = [x_m(1),\ x_m(2),\ \ldots,\ x_m(n)]^T \tag{1.6}$$

and

$$\mathbf{x}^T(n) = [x_1(n),\ x_2(n),\ \ldots,\ x_M(n)] \tag{1.7}$$

which respectively represent the signals received at element m from time t_1 to t_n and those received by the M elements at time t_n (snapshot). Also, from Eqn. (1.4c), it can be seen that each row of the data matrix $\mathbf{X}(n)$ constitutes a single snapshot $\mathbf{x}^T(n)$. Then from Eqn. (1.1),

$$e(n) \;=\; y(n) - \mathbf{x}^T(n)\mathbf{w}(n). \tag{1.8}$$

In Eqns. (1.1) to (1.8) $n \in N$, is the snapshot number. It is assumed that $x_m(n)$ is the narrow band signal which is received at array element m at time t_n. Within the receiver pass band the signal takes the form

$$x_m(n) \;=\; A_d\, e^{\jmath\{(m-1)\varphi(\theta_d)+\psi_d\}} + \sum_{k=1}^{K} A_k\, e^{\jmath\{(m-1)\varphi(\theta_k)+\psi_k\}} + \nu_m(n) \tag{1.9}$$

where $\varphi(\theta_i)$ is the phase difference between adjacent elements given by

$$\varphi(\theta_i) \;=\; \frac{2\pi d}{\lambda}\,\sin(\theta_i)$$

with d being the interelement spacing and λ the antenna wavelength. Also in Eqn. (1.9), the other variables are defined as follows:

θ_d – direction of angle–of–arrival (AOA) of the desired signal,

θ_k – direction of arrival of the kth interfering source,

A_d – amplitude of the desired signal,

A_k – amplitude of the kth interfering source,

$\nu_m(n)$– receiver noise assumed to be Gaussian; with zero mean, and variance σ^2, and

ψ_d, ψ_k– uniformly distributed random variables with probability density

$$p(\psi_i) \;=\; \begin{cases} 1/2\pi \;:\; 0 \le \psi_i \le 2\pi \\ 0 \quad\;\; :\; \text{elsewhere.} \end{cases}$$

The objective of adaptive beamforming, as expressed by Eqn. (1.1), is to minimize the combined output signal, which includes interference and thermal noise, while maintaining a constant gain in the direction of the desired signal. In general, this leads to the maximization of the signal–to–interference–and–noise ratio (SINR). The desired signal is usually assumed to be both weaker than, and uncorrelated with, the unwanted signal. The first assumption is usually valid in radar applications. This is especially true for those interferers that are generated by hostile or intelligent jammers [4]. This follows from the fact that, in terms of range, radar returns follow the inverse fourth power law whereas interfering signals follow the inverse second power law. However, if the desired signal is at about the same level as the interferers, such techniques as mainbeam constraints [9] or reference signal generators can be employed [4] to separate or/and isolate the wanted signal from the others so that it is not included in the adaptive nulling operation.

Although, in general, the assumption is made that the desired and unwanted signal are uncorrelated, this is not necessary always the case. If the desired signal and the interferers are correlated the adaptive beamformer not only fails to form deep nulls in the directions of the coherent interferers but also partially or completely cancels the desired signal. This problem is particularly relevant in the case where the interferer is a multipath signal and therefore correlated with the wanted signal. There are a number of methods to deal with fully or partially coherent sources [10–14]. The spatial smoothing method [10–12] has acquired some prominence amongst these techniques. It has been shown that this tchnique can reduce the degree of correlation between the signals, thereby improving the performance of the adaptive beamformer.

There are a number of algorithms that can be used to solve the problem defined by Eqn. (1.1). Those, that are classified as the *classical adaptive beamforming algorithms*, are typified by the control–loop algorithm by Howells–Applebaum [15,16], the LMS algorithm by Widrow *et al.* [17], and the SMI algorithm by Reed *et al.* [18]. The Howells–Applebaum algorithm and the LMS algorithm are closed–loop algorithms, which are well–known for slow convergence because their rate of convergence is heavily dependent upon the choice of step size and the condition of subjected environment. On the other hand, the SMI algorithm, which is an open–loop algorithm, involves computation of the covariance matrix (of the input data) and its inversion. As a consequence, this requires a large amount of memory for storage of data during processing. The applicability of these classical algorithms is therefore limited to communications systems, where the response time is not too critical and/or the number of array elements is small. Also, they are well suited for stationary environments where the rate of change of the interference or jammer environments are relatively slow compared with the adaptation rate. For radar applications, the interference environment is usually non–stationary, particularly in the case of hostile or intelligent jammers. Thus, to achieve and/or improve real–time signal processing in adaptive beamforming it is necessary to use faster digital processors. This requirement places a stringent demand on the use of array processors with higher throughput rate.

There has been a considerable amount of discussion of adaptive beamforming techniques that are carried out in beamspace [19–21]. These expositions are typified by the work on partial adaptivity of large arrays by Chapman [19], and the adaptive–adaptive beamforming method by Brookner *et al.* [20]. The retrodirective eigenvector beam concept introduced by Gabriel [21] is a useful tool and gives a valuable insight into the fundamental principles of beam–space

processing. One of the advantages of the beamspace method is that degrees–of–freedom for the array necessary to achieve adaptivity is proportional to the number of of unwanted signals rather than to the number of antenna elements.

Recently, a class of algorithms, which use QR decomposition to perform the least–squares adaptive beamforming described by Eqn. (1.1), has been proposed [7,8], [22]. These algorithms are called *modern adaptive beamforming algorithms.* It has been shown that they are highly stable, converge rapidly when implemented using Givens rotations, and are amenable to the processing with systolic arrays. As well, they permit the incorporation of superresolution techniques for nulling out interferers in those cases where the desired and unwanted signals are separated by less than the antenna beamwidth.

Two–Dimensional Adaptive Beamforming

Most of the work that has been carried out in the past, however, has concentrated on applications to linear array antennas, i.e. the 1–D case. Because beam steering, in the case of linear arrays, is restricted to one plane, its beam adaptation is likewise limited to only nulling interferers in one plane. Since, in practice, most antenna arrays are planar arrays, the focus of future developments in adaptive beamforming must start to shift to the 2–D case.

Two–dimensional adaptive beamforming is rarely discussed in the literature. It is thought that there are two reasons for the lack of 2–D results. First, there is the general impression amongst workers in the field that the principles underlying the 2–D case are a simple extension of the 1–D case [23,24]. Secondly, it is felt that the only way around the complexity that is inherent to the 2–D case

is by means of subarraying, i.e. by reducing the degrees of freedom [19]. As a result, very little work has been carried out to optimize 2–D adaptive beamforming techniques. In fact, one of the earliest discussions in the literature on 2–D adaptive beamforming was provided by [19]. There had not been much discussion beyond this early paper. In this work, the 2–D adaptive beamforming problem is analyzed by using a subarray transformation method to reduce the complexity in computation and implementation. Also, the simple case is treated where the adaptation process takes place on the rows and columns of the array, i.e. contiguous row elements are combined together to form subarrays and the column elements are combined in a similar manner. Recently, though, the problem has been taken up again, albeit in a cursory manner, by [23,24]. These workers propose a solution that is simply an extension of the 1–D method. Their solution is based on lexicographic ordering of the adaptive weight matrix elements. 2–D adaptive beamforming based on the QRD–LS algorithm was only recently introduced by [25]. Again, only recently, a discrete form of a 2–D adaptive LMS algorithm was proposed and presented by [26] for use in image processing.

Calibration Errors in Antenna Arrays

Another problem associated with adaptive beamforming algorithms and AOA estimations is that the data from the antenna are usually assumed to be ideal, i.e. not to require calibration, and at the most may be degraded by additive noise (see Eqn. (1.1)). This assumption, however, does not often hold in practice because of the presence of both gain and phase errors in antenna elements. The development of an effective antenna calibration technique is one of the major practical problems in implementing digital adaptive beamforming. The sensitivity of high performance algorithms to gain and phase errors associated with the elements of phased array

antennas has been studied extensively in the literature [27–29]. These effects consist of: (1) reduced resolution in signal AOA estimations, (2) increased sidelobe levels, and (3) limitation to the nulling performance of adaptive beamformers.

Many previous workers have raised the problem, but very few have attempted to solve it [30–32]. In [30], Zahm proposes injecting pseudo noise into the array processor in order to reduce the degradation caused by array errors. However, this method reduces the signal–to–noise ratio (SNR), thereby reducing the processor's effectiveness in nulling out undesired signals. The eigenstructure based method proposed in [31] relies heavily on the Toeplitz structure of the array covariance matrix; thus it can only be applied to uncorrelated signals and arrays with uniform spacing. The subspace approach presented in [32] has shown some interesting promises in array error estimation and antenna system calibration. However, the method assumes that some of the signals' AOA are known in advance. In practice, this assumption may prove to be a limiting factor, especially when dealing with interference from intelligent jammers or in the case of multipath signals. As well, it will be shown in the thesis (Chapter 5) that the set of AOA derived for the undesired signals is not optimal. Very recently, in [33], a method, which only deals with real elemental gain error, was proposed. And in [34], the AOAs of impinging signals and the calibration errors are recursively computed via the minimum norm least–squares method. However, it has been pointed out in [34] that there exist several local minima, which correspond to different sets of AOAs.

1.2 Scope of Thesis

In this thesis, 2–D adaptive beamforming algorithms typified by the 2–D

LMS algorithm, the 2–D Howells–Applebaum algorithm, and the QRD algorithm, which can be implemented using systolic arrays, will be developed and presented. In the case of the 2–D LMS and 2–D Howells–Applebaum algorithms, both the recursive form and the matrix form are given. Also, their relationships with the 1–D adaptive beamforming algorithms are investigated. The concepts of 2–D eigenbeams will be developed to help in graphically illustrating the fundamentals of the various adaptive beamforming algorithms. In the case of 2–D adaptive beamforming carried out using the QRD–LS algorithm, the systolic array's configuration is three–dimensional (3–D), and the PEs are highly pipelined and locally interconnected. The structure uses a manifold of triangular systolic arrays to carry out the QRD–LS minimization [6]. It will be shown that with a proper flow of input data, in time skew format, adaptations in both directions of a planar array can be computed concurrently and simultaneously.

Also, in this thesis, calibration errors for antenna arrays will be modeled and a self–calibration algorithm will be derived and presented. The derivation of the algorithm is based on the eigen–based method and the least–squares norm minimization. It will be shown that the self–calibration algorithm is capable of automatically estimating the AOA of the received signals and calibrating the array with a minimum mismatch in gain and phase errors. Results obtained using both simulated and measured data will be given. In the case of the experimental results, the measurements were recorded in the presence of multipath signals – one of the signal environment which was identified earlier as being a difficult one.

It should be emphasized that the unique contributions in this thesis are: (1) the derivation of the 2–D adaptive beamforming algorithms and the development of the 2–D eigenbeam concept, (2) the formulation and derivation of the 2–D adaptive

beamforming using the QRD–LS algorithm and its 3–D systolic array implementation, which is capable of performing the adaptation concurrently and simultaneously in both directions of a 2–D array, and (3) the derivation of the self–calibration algorithm for antenna arrays, which is the main contribution of the thesis. In the last contribution, for the first time, the self–calibration schemes for antenna arrays are demonstrated by using both simulated and measured data.

The organization of the thesis is as follows. Chapter 2 presents the classical adaptive beamforming techniques. A short review of the 1–D adaptive beamforming algorithms will be given. Two–dimensional adaptive beamforming algorithms such as the 2–D LMS algorithm and 2–D Howells–Applebaum algorithm will then be derived and presented. Also, the relationship between the 1–D and 2–D adaptive algorithms will be given. Finally, the concept of an 2–D eigenvector beam will be developed. Modern adaptive beamforming techniques are given in Chapter 3. In particular, the QRD–LS algorithm and its systolic array implementation will be presented. Systolic array architectures for both unconstrained and constrained adaptive beamformers will be presented. Also, in this Chapter the QRD–LS algorithm will be formulated for 2–D adaptive beamforming and its systolic array implementation will be given. The performance of the modern adaptive beamforming algorithms will be evaluated in Chapter 4. Results will be presented for both 1–D and 2–D adaptive beamforming algorithms. The self–calibration algorithm for correcting antenna array errors is presented in Chapter 5. As well, calibration results are given to demonstrate the effectiveness of this algorithm. These were obtained by using both simulated and experimental data. Finally, the conclusions and a general discussion of the results of this study are given in Chapter 6 followed by recommendations and suggestions for future work.

CHAPTER 2

CLASSICAL ADAPTIVE BEAMFORMING TECHNIQUES

2.1. Introduction

Classical adaptive beamforming methods have their roots in different fields, including retrodirective and self–phasing RF antenna arrays, sidelobe cancellers, adaptive filters, acoustic or sonar arrays, and seismic arrays.

The development of adaptive beamforming technology can be tracked back to the invention of the IF sidelobe canceller by Howells and the use of phase–locked loops for self–steering in antenna arrays by Altman and Sichak in the late 1950's. The second major contribution to adaptive array antennas is the invention of the IF sidelobe canceller by Howells in 1957 [15]. Further analysis was made by Applebaum in 1966 [16], in which he derived the control law governing the operation of an adaptive antenna using a control loop with each element of the array. The derivation of the Howells–Applebaum algorithm is based on maximizing the signal–to–noise ratio (SNR) at the array output. Another independent work on adaptive antenna systems was developed by Widrow and his co–workers in 1967 [17], who firmly established the least–mean–square error (LMS) algorithm based on the method of steepest descent. It is interesting to note that although the derivations of the maximum SNR algorithm by Applebaum and the LMS algorithm by Widrow *et al.* use different approaches, they are in fact very similar. Both

algorithms derive the adaptive weights that are applied to the array antenna elements by sensing the correlation between element signals, and both converge toward the optimum Wiener solution [35].

Since the pioneering work mentioned above, several adaptive beamforming techniques and algorithms have been proposed in the literature in an attempt to improve and enhance the performance of adaptive antenna systems. In general, these adaptive beamforming algorithms fall under two distinct categories, which are: closed–loop and open–loop techniques. Closed–loop techniques, which are derived using the steepest descent gradient method, consist of the Howells–Applebaum algorithm, the LMS algorithm, and others including the modified LMS algorithm by Griffiths [36], Frost [37], and Compton [38]. Open–loop techniques are developed by using the "direct solution" of the sample covariance matrix based on least–squares minimization. The techniques include the sample matrix inversion (SMI) algorithm developed by Reed and his co–workers in 1974 [18], and others of a similar kind [39–40]. The minimum variance distortionless response (MVDR) beamforming algorithm [40] and the superresolution technique [21] also fall into this category. Good overviews of adaptive beamforming algorithms and related issues are given in [1–5], [35] and also [41–43].

This Chapter begins with a brief review of adaptive beamforming techniques for linear arrays. It then proceeds to the derivation and development of 2–D adaptive beamforming algorithms with planar arrays, namely the 2–D LMS algorithm and 2–D Howells–Applebaum algorithm. The relationship between 1–D and 2–D adaptive beamforming algorithm will be derived. As well, 2–D eigenbeams will be presented.

2.2. 1–D Adaptive Beamforming Algorithms

2.2.1. *LMS algorithm*

Typical of the classical adaptive beamforming algorithms are the LMS algorithm and the Howells–Applebaum algorithm. In general, the key components of a closed–loop adaptive system are illustrated in Fig. 2.1. The weight vector $\mathbf{w}(n)$ is updated at each snapshot by an adaptive processor that responds to the output signal $e(n)$. The derivation of $e(n)$ is based on the steepest descent method. Changes in the weight vector are made along the direction of the estimated gradient vector. Accordingly

$$\mathbf{w}(n+1) = \mathbf{w}(n) - \frac{\mu}{2}\nabla(n) \tag{2.1}$$

where $\nabla(n)$ is the gradient vector of the array output with respect to the weight vector $\mathbf{w}(n)$, which is defined by

$$\nabla(n) = \frac{\partial}{\partial \mathbf{w}^*}\{\mathrm{E}[|e(n)|^2]\} = -2\mathrm{E}[\mathbf{x}^*(n)e(n)] \tag{2.2}$$

where

$$e(n) = y(n) - \mathbf{x}^T(n)\mathbf{w}(n) \tag{2.3}$$

and μ is the update gain factor or step size parameter. In general, μ must be chosen to lie in the range

$$0 \leq \mu \leq 2/\mathrm{Tr}[\Phi] \tag{2.4}$$

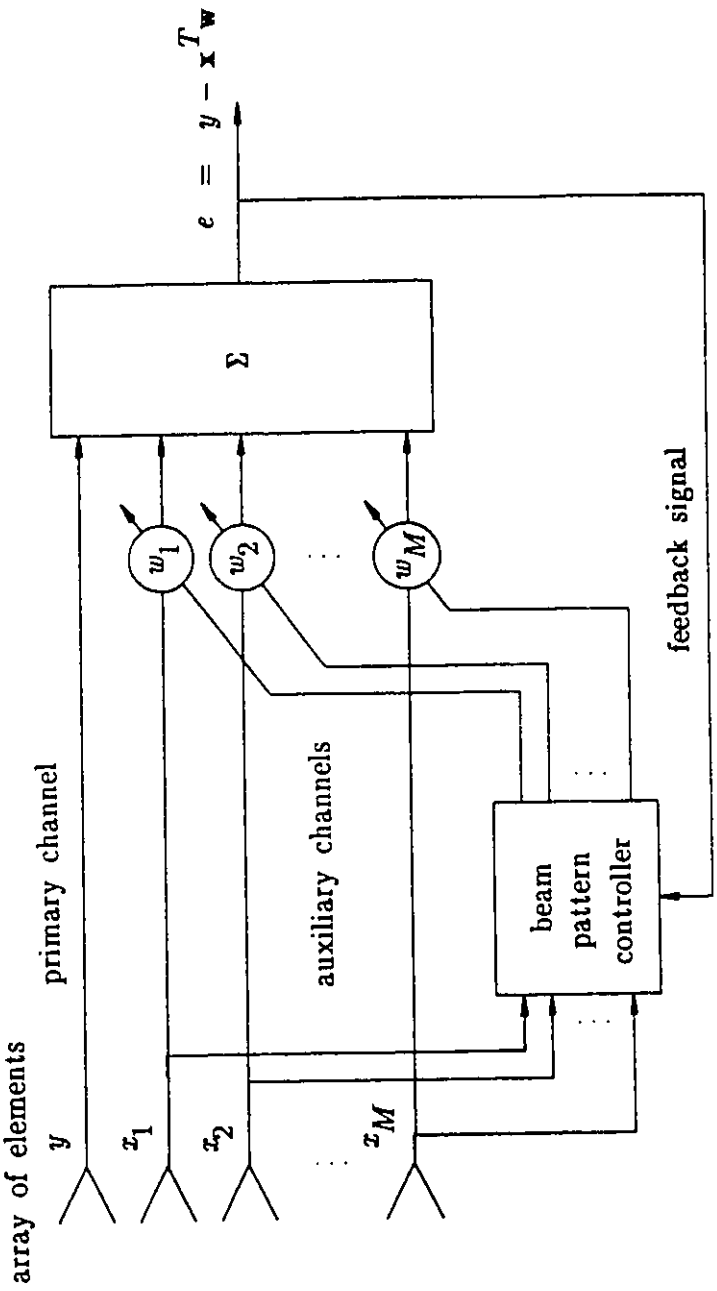


Fig. 2.1 Closed–loop adaptive system

to ensure the stability of the LMS algorithm. In Eqn. (2.4), Φ is the covariance matrix of the data vector given by

$$\Phi = E[\mathbf{x}^*(n)\mathbf{x}^T(n)]. \tag{2.5}$$

Substituting Eqns. (2.2), (2.3), and (2.5) in Eqn. (1.8) yields

$$\mathbf{w}(n+1) + (\mu\Phi - \mathbf{I})\mathbf{w}(n) = \mu\mathbf{p} \tag{2.6}$$

where $\mathbf{p}$ is the correlation vector between the primary signal $y(n)$ and the snapshot vector $\mathbf{x}^T(n)$ and is defined by

$$\mathbf{p} = E[y(n)\mathbf{x}^*(n)]. \tag{2.7}$$

Thus, the optimum weight vector of Eqn (2.6) denoted by $\mathbf{w}^0$, which gives the least mean squares error ($\nabla(n) \rightarrow 0$, $\mathbf{w}(n+1) \rightarrow \mathbf{w}(n)$), satisfies the Wiener–Hopf equation

$$\mathbf{w}^0 = \Phi^{-1}\mathbf{p} \tag{2.8}$$

where Φ^{-1} is the inverse of the covariance matrix Φ.

2.2.2. *Howells–Applebaum algorithm*

In the Howells–Applebaum algorithm, the SINR is maximized by means of cross–correlation filters (Howells–Applebaum loops). A typical arrangement based on the Howell–Applebaum loop is given in Fig. 2.2. Correlation loops can be

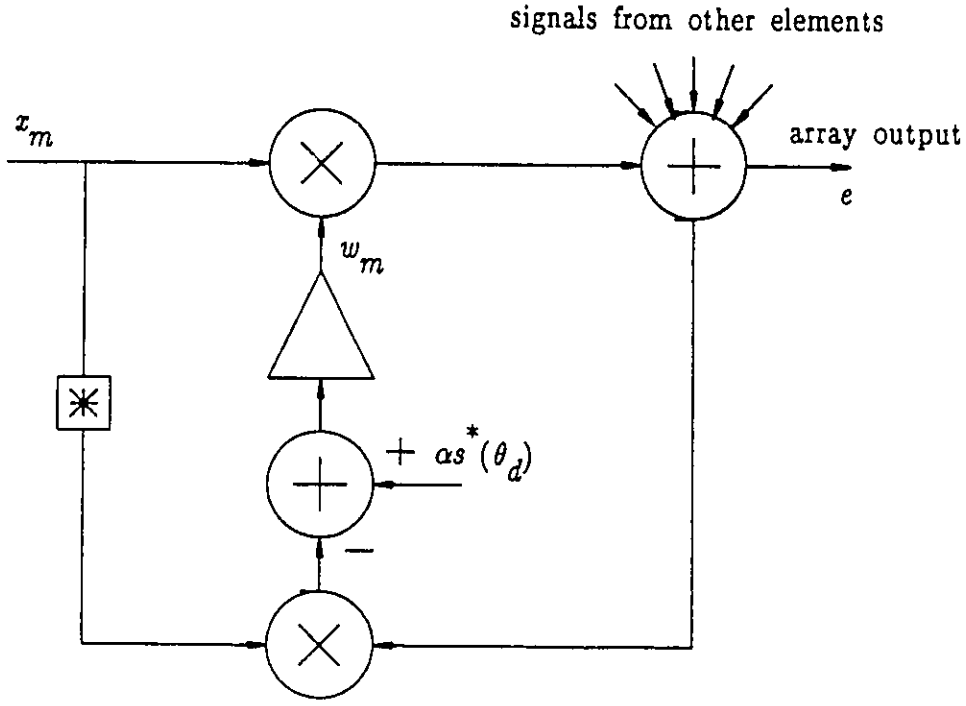


Fig. 2.2 Correlation loop of Howells–Applebaum algorithm

implemented using mixers, high gain amplifiers, and narrow band filters or, alternatively, they can be implemented using digital processors. In the latter case, the algorithm can be formulated in the same manner as the LMS algorithm. The tap weight update is given by

$$\mathbf{w}(n+1) \;=\; \mathbf{w}(n) - \frac{\mu}{2}\nabla_s(n) \tag{2.9}$$

where it follows from Fig. 2.2 that

$$\nabla_s(n) \;=\; -2\alpha \mathbf{s}^*(\theta_d) + 2\mathbf{x}^*(n)e_s(n). \tag{2.10}$$

Central to the technique is the vector

$$\mathbf{s}^*(\theta_d) \;=\; [1,\; e^{-j\varphi(\theta_d)},\; e^{-j2\varphi(\theta_d)},\; \ldots,\; e^{-j(M-1)\varphi(\theta_d)}]. \tag{2.11}$$

This is called the steering vector and is used to set up a shaped received beam which is pointed in the desired signal direction, θ_d. Furthermore, in Eqn. (2.10), the quantity α is chosen so as to keep the gain of the receiver beam constant. The error signal $e_s(n)$ is given by

$$e_s(n) = \mathbf{x}^T(n)\mathbf{w}(n). \tag{2.12}$$

With the introduction of the covariance matrix Φ, Eqn. (2.9) becomes

$$\mathbf{w}(n+1) + (\mu\Phi - \mathrm{I})\mathbf{w}(n) \;=\; \mu\alpha\mathbf{s}^*(\theta_d). \tag{2.13}$$

The optimum weight vector follows directly from Eqn. (2.13) and is given by

$$\mathbf{w}^{O} = \alpha\Phi^{-1}\mathbf{s}^{*}(\theta_d). \tag{2.14}$$

Clearly, the Howells–Applebaum algorithm has the same form as the LMS algorithm except for the steering vector, $\alpha\mathbf{s}^{*}(\theta_d)$, being used in place of the correlation vector $\mathbf{p}$. This difference is due to the fact that in the Howells–Applebaum beamformer the direction of desired signal is known in advance which is not the case for the LMS beamformer. With the Howells–Applebaum algorithm, it is assumed that the desired signal is not contained in the data processed by the beamformer, therefore one is not able to use a reference signal $y(n)$ in formulating Eqn. (2.12) as in the case of the LMS algorithm. It should be noted that the reference signal used in the LMS algorithm, which is often referred as the desired signal, need not be perfect replica of the desired signal. It is only necessary that it correlates with the desired signal. Several techniques have been proposed in the literature for generating the reference signals that are used with the LMS algorithm. However, this subject is beyond the scope of this thesis. Interested readers are referred to papers by Compton [44] and Winters [45].

2.2.3. *Eigenbeam concept*

Eigenvector beams are useful for providing insight into the interference nulling performance of adaptive beamforming techniques. They are formed by applying an orthonormal transformation to the covariance matrix of data signals [32]. The orthonormal transformation can be carried out using the singular value decomposition (SVD) as follows.

The SVD of the covariance matrix Φ is given by

$$\Phi = \mathbf{U}\Sigma\mathbf{U}^H \tag{2.15}$$

where

$$\mathbf{U} = [\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_K, \ldots, \mathbf{u}_M] \tag{2.16}$$

is an unitary matrix of M columns which are eigenvectors of the covariance matrix Φ. The first K eigenvectors ($\mathbf{u}_1$, $\mathbf{u}_2$, ..., $\mathbf{u}_K$), called the *principal eigenvectors*, constitute the *signal subspace*, and the last M–K eigenvectors ($\mathbf{u}_{K+1}$, $\mathbf{u}_{K+2}$, ..., $\mathbf{u}_M$) constitute the *noise subspace*. Σ is a diagonal matrix of dimensions M by M. With the assumption that the noise is white, it then has the form

$$\Sigma = \text{diag}(\lambda_1, \lambda_2, \ldots, \lambda_K, \sigma^2, \ldots, \sigma^2) \tag{2.17}$$

with

$$\lambda_1 \geq \lambda_2 \geq \ldots, \geq \lambda_K > \lambda_{K+1} = \sigma^2$$

where λ_k is the eigenvalue corresponding to the kth source.

The matrix Φ then may be written as

$$\Phi = \sum_{k=1}^{K} \lambda_k \mathbf{u}_k \mathbf{u}_k^H + \sigma^2 \sum_{k=K+1}^{M} \mathbf{u}_k \mathbf{u}_k^H. \tag{2.18}$$

It follows that

$$\Phi^{-1} = \sum_{k=1}^{K} \frac{1}{\lambda_k} \mathbf{u}_k \mathbf{u}_k^H + \frac{1}{\sigma^2} \sum_{k=K+1}^{M} \mathbf{u}_k \mathbf{u}_k^H$$

$$= \frac{1}{\sigma^2}\mathbf{I} - \frac{1}{\sigma^2}\sum_{k=1}^{K}(1 - \frac{\sigma^2}{\lambda_k})\mathbf{u}_k\mathbf{u}_k^H. \tag{2.19}$$

Equation (2.10) now becomes

$$\mathbf{w}^O = \frac{\alpha}{\sigma^2}\mathbf{s}^*(\theta_d) - \frac{\alpha}{\sigma^2}\sum_{k=1}^{K}(1 - \frac{\sigma^2}{\lambda_k})\gamma_k\mathbf{u}_k. \tag{2.20}$$

with

$$\gamma_k = \mathbf{u}_k^T\mathbf{s}^*(\theta_d) \tag{2.21}$$

The first term on the RHS of Eqn. (2.20) is the quiescent weight vector (unadapted) of the antenna array and the second term represents the sum of the eigenbeam vectors, which are subtracted from the first term to obtain the adapted weight vector $\mathbf{w}^O$. The magnitudes of the eigenbeams are such that when summed together and subtracted from the quiescent beam the K interferers are nulled. Also, the gain factor α must be adjusted to keep the main antenna beam constant at certain value above the noise level (given by σ^2). As the noise power increases the eigenbeams will be distorted resulting in shallow nulls at the jammer locations, which subsequently leads to a decrease in the SINR. Eqn. (2.20) can be adapted for use in conjunction with the LMS algorithm by substituting the correlation vector $\mathbf{p}$ in place of $\alpha\mathbf{s}^*(\theta_d)$.

It is worth noting that Eqn. (2.20) also establishes the fundamental concept of beamspace adaptive beamforming [22,23]. In this case, the eigenvectors $\mathbf{u}_k$ are replaced by directional beam vectors $\mathbf{a}(\theta_k)$, in which

$$\mathbf{a}(\theta_k) = [1,\ e^{-j\varphi(\theta_k)},\ e^{-j2\varphi(\theta_k)},\ \ldots,\ e^{-j(M-1)\varphi(\theta_k)}]. \tag{2.22}$$

where θ_k is the signal direction of the kth interferer.

2.2.4. *Sample Matrix Inversion Algorithm*

The major problem associated with an adaptive beamformer based on the gradient descent method, which is true of the LMS algorithm and the control-loop method of the Howells–Applebaum algorithm, is that under certain external signal environments the convergence is poor. In recent years, the least-squares minimization method, a direct solution of Eqn. (1.1), has been proposed to improve the rate of convergence. This is known as the sample matrix inversion (SMI) algorithm [21]. The minimum norm of the output signal of Eqn. (1.1) is given by

$$\|\mathbf{e}(n)\|^2 = \|\mathbf{y}(n) - \mathbf{X}(n)\mathbf{w}(n)\|^2 \tag{2.23}$$

Its gradient is of the form

$$\nabla\|\mathbf{e}(n)\|^2 = 2\{\ \mathbf{X}^H(n)\mathbf{y}(n) - \mathbf{X}^H(n)\mathbf{X}(n)\mathbf{w}(n)\ \} \tag{2.24}$$

where $\mathbf{X}^H(n)$ denotes the Hermitian transpose of the data matrix $\mathbf{X}(n)$. And the least-squares solution is given by

$$\mathbf{w}(n) = \hat{\Phi}(n)^{-1}\hat{\mathbf{p}}(n) \tag{2.25}$$

where $\hat{\Phi}(n)^{-1}$ denotes the inverse of the maximum likelihood estimate of the covariance matrix

$$\hat{\Phi}(n) = \mathbf{X}^H(n)\mathbf{X}(n) \tag{2.26}$$

and

$$\hat{\mathbf{p}}(n) = \mathbf{X}^H(n)\mathbf{y}(n). \tag{2.27}$$

Equation (2.27) gives the estimate of the vector of correlations between the data matrix $\mathbf{X}(n)$ and the reference signal vector $\mathbf{y}(n)$.

Note that Eqn. (2.25) is similar to that of Eqn. (2.8) except for the fact that estimates of the covariance matrix and the correlation vector are used in place of their true forms. It has been shown in [21] that the SMI algorithm converges to within 3 dB of the steady state in $2M$ iterations, i.e, with the use of approximately $2M$ data samples (snapshots), where M is the number of array elements. In many applications, because of the fast convergence feature of the SMI algorithm, it is much preferred over the slower closed–loop algorithms [33]. Usually these applications are ones where the external signal scenario undergoes rapid changes. In these instances the closed–loop techniques are too slow to be effective.

A schematic diagram of an adaptive beamformer which is based on the SMI algorithm is given in Fig. 2.3. The adaptive processor has three key components: (1) a estimate covariance processor to form and store the covariance matrix estimate, (2) a high–speed arithmetic processor to solve the matrix operation defined by (2.25), and (3) a weight–combining processor to apply the weight vector to a digital beamformer. Obviously, this type of architecture is complicated and

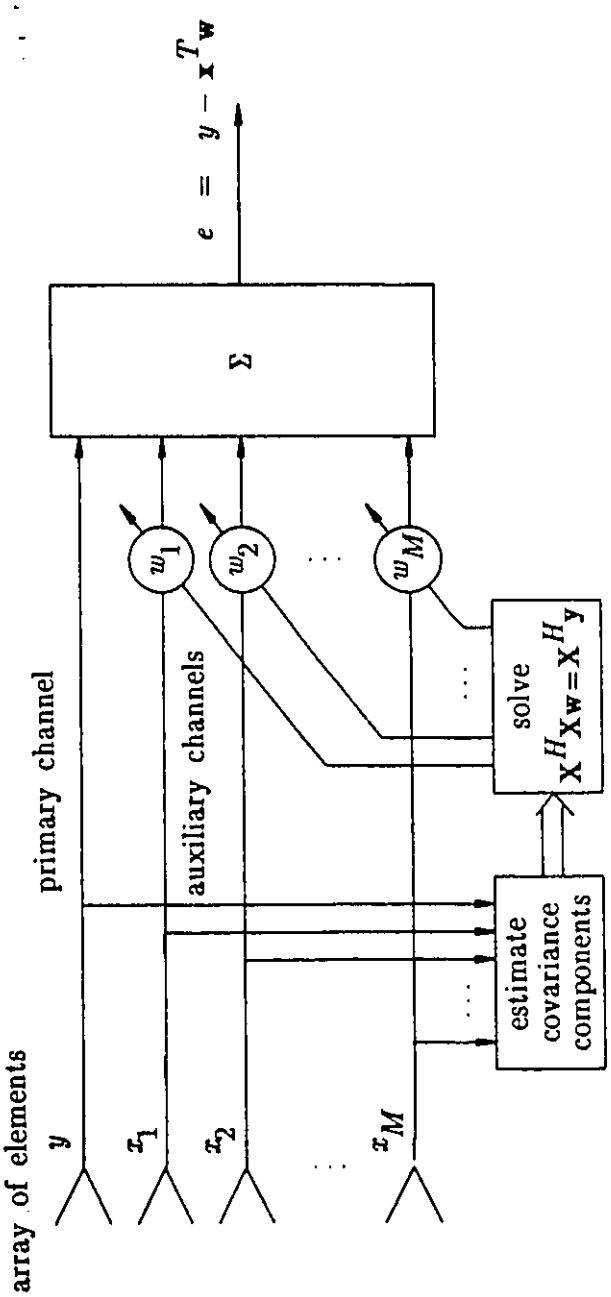


Fig. 2.3 Open-loop adaptive system

difficult to design and in general not suitable for VLSI implementation. The second problem associated with the SMI algorithm is its numerical instability for signal space conditions, as revealed by the condition number of the matrix $\hat{\Phi}(n)$ in Eqn. (2.26), which is

$$C\{\hat{\Phi}(n)\} = C\{\mathbf{X}^H(n)\mathbf{X}(n)\} = [\beta_{max}/\beta_{min}]^2. \tag{2.28}$$

β_{max} and β_{min} are the largest and the smallest singular values of the data matrix $\mathbf{X}(n)$, respectively. Eqn. (2.28) indicates that the condition number of the estimated covariance matrix $\hat{\Phi}(n)$ is much greater than that of the data matrix $\mathbf{X}(n)$. Since there is a direct linkage between condition number and the stability of the solution to a system of linear equations, an algorithm that avoids forming the covariance matrix explicitly, and which operates directly on the data matrix, is likely to have much better numerical stability than one based on a covariance matrix.

Another problem associated with the SMI is the computational load on the beam controller. The formation of the sample matrix and the solution of the optimum weight require respectively about M^3 and $M^3/6$ complex multiplications, leading to a total of $7M^3/6$ complex operations [18]. With an M^3–dependence, it is clear that for real–time application there would be a strong need to explore parallel processing techniques to maintain the solution throughput rate implied by current system demands.

The above problems can be solved with the use of VLSI array processors, in particular the systolic array architectures, which are associated with the modern adaptive beamforming techniques presented in the next Chapter.

2.3. 2–D Adaptive Beamforming Algorithms

The configuration for a 2–D antenna array with dimensions L by M is given in Fig. 2.4. The angles–of–arrival of signals impinging onto the array are described by the polar angle θ and the azimuthal angle ϕ. The far field signal received at the array element lm is

$$x_{lm}(n) = A_d e^{j\frac{2\pi}{\lambda}\{(l-1)d_x\cos\alpha_d+(m-1)d_y\cos\beta_d+\psi_d\}} + \sum_{k=1}^{K} A_k e^{j\frac{2\pi}{\lambda}\{(l-1)d_x\cos\alpha_k+(m-1)d_y\cos\beta_k+\psi_k\}} + \nu_{lm}(n) \qquad (2.29)$$

where d_x, d_y are element spacings along the rows and columns, respectively, and

$$\cos\alpha_i = \sin\theta_i\cos\phi_i \text{ and } \cos\beta_i = \sin\theta_i\sin\phi_i \qquad (2.30)$$

in which θ_i and ϕ_i are the elevation and azimuthal angles of arrival. Also in Eqn. (2.29), $\nu_{lm}(n)$ is the receiver noise component assumed to be Gaussian; with zero mean, and variance σ^2.

At a glance, Eqn. (2.29) indicates that each snapshot for the 2–D array is a 2–D array of numbers. Therefore the data matrix $\mathbf{X}(n)$ is a 3–D matrix. It follows from the 1–D case that the error signal for a 2–D array can be expressed as

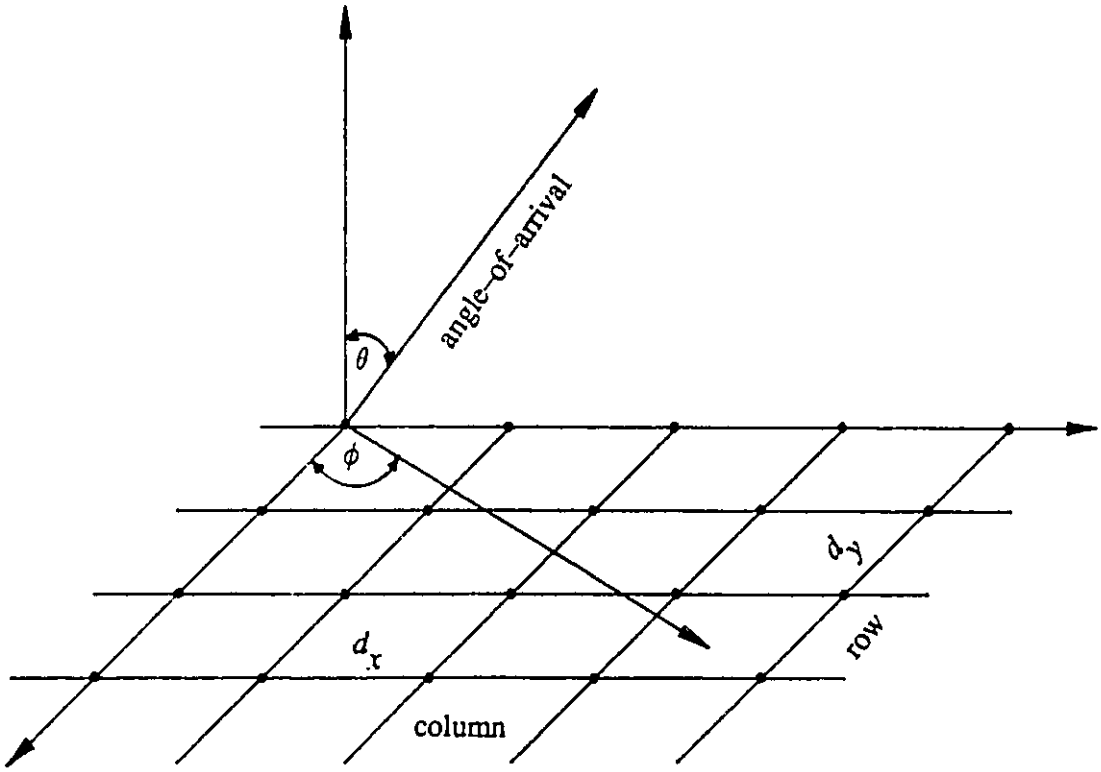


Fig. 2.4 Geometry of a planar array

$$e(n) = y(n) - \sum_{l=1}^{L} \sum_{m=1}^{M} w_{lm}(n) x_{lm}(n). \tag{3.31}$$

A statement of the optimization procedure required for implementing 2–D adaptive beamforming is as follows. Given a primary vector $\mathbf{y}(n) = \{ y(n) \}$ and a 3–D data matrix $\mathbf{X}(n)$ consisting of data matrices $\mathbf{X}(n) = \{ x_{lm}(n) \}$, estimate the adaptive weight matrix $\mathbf{W}(n)$ consisting of weight elements $w_{lm}(n)$, which minimizes the residual power $\|e(n)\|^2$ at the output of the beamformer. Note that the primary signal $y(n)$ described in Eqn. (2.31) is obtained by using either a high gain antenna (feed horn) or a primary planar array.

2.3.1. *2–D LMS Algorithm*

Using the 1–D LMS algorithm as a guide, the 2–D LMS algorithm can be defined as follows. Estimate the weight matrix $\mathbf{W}(n)$ of a L–by–M array in such a manner as to minimize the least–mean–squares of the output signal $e(n)$ defined by Eqn. (2.31).

The LMS estimate for Eqn. (2.31) is given by

$$\mathrm{E}[|e(n)|^2] = \mathrm{E}[|y(n) - \sum_{l=1}^{L} \sum_{m=1}^{M} w_{lm}(n) x_{lm}(n)|^2]$$

$$= \mathrm{E}[\{y(n) - \sum_{l=1}^{L} \sum_{m=1}^{M} w_{lm}(n) x_{lm}(n)\}\{y^*(n) - \sum_{l=1}^{L} \sum_{m=1}^{M} w^*_{lm}(n) x^*_{lm}(n)\}]$$

$$= \mathrm{E}[|y(n)|^2] - \sum_{l=1}^{L} \sum_{m=1}^{M} w_{lm}(n) \mathrm{E}[y^*(n) x_{lm}(n)] - \sum_{l=1}^{L} \sum_{m=1}^{M} w^*_{lm}(n) \mathrm{E}[y(n) x^*_{lm}(n)]$$

$$+ \sum_{l=1}^{L} \sum_{m=1}^{M} \sum_{p=1}^{L} \sum_{q=1}^{M} w_{lm}(n) w^{*}_{pq}(n) \mathrm{E}[x_{lm}(n) x^{*}_{pq}(n)]. \tag{2.32}$$

In discrete form, the weight matrix can be updated as follows

$$\mathbf{W}(n+1) = \mathbf{W}(n) - \frac{\mu}{2} \nabla(n) \tag{2.33a}$$

and adaptive weight element $w_{lm}(n)$ is of the form

$$w_{lm}(n+1) = w_{lm}(n) - \frac{\mu}{2} \nabla_{lm}(n) \tag{2.33b}$$

where $\nabla(n)$ is 2-D instantaneous gradient matrix defined by

$$\nabla(n) = \{ \nabla_{lm}(n) \} = \{ \frac{\partial}{\partial w^{*}_{lm}} \{ \mathrm{E}[| e(n) |^{2}] \} \}. \tag{2.34}$$

Moreover, it follows from Eqns. (2.31) and (2.32) that

$$\frac{\partial}{\partial w^{*}_{lm}} \{ \mathrm{E}[| e(n) |^{2}] \} = -2\mathrm{E}[e(n) x^{*}_{lm}(n)]. \tag{2.35a}$$

Thus

$$\nabla_{lm}(n) = -2\mathrm{E}[e(n) x^{*}_{lm}(n)]. \tag{2.35b}$$

Substitution of Eqns. (2.35b) and (2.31) into Eqn. (2.33b) yields

$$w_{lm}(n+1) = w_{lm}(n) - \mu(p_{lm} - \sum_{p=1}^{L} \sum_{q=1}^{M} w_{pq} r_{p-l, q-m}) \tag{2.36}$$

in which

$$p_{lm} = \mathrm{E}[y(n)x^{*}_{lm}(n)] \tag{2.37a}$$

and

$$r_{p-l,q-m} = \mathrm{E}[x^{*}_{lm}(n)x_{pq}(n)]. \tag{2.38a}$$

It follows from Eqn. (2.36) that the optimal solution for the weight elements satisfies

$$p_{lm} = \sum_{p=1}^{L}\sum_{q=1}^{M} w^{o}_{pq} r_{p-l,q-m} \tag{2.39}$$

which is the 2-D Wiener–Hopf equation of the first kind [46].

It should be noted that p_{lm} and $r_{p-l,q-m}$ described in Eqns. (2.37a) and (2.38a) are the elements of the cross–correlation matrix P and the correlation matrix Φ, respectively, i.e,

$$\mathrm{P} = \{\, p_{lm} \,\} \tag{2.37b}$$

and

$$\Phi = \{\, r_{p-l,q-m} \,\}$$

$$= \begin{bmatrix} [\Phi_0] & [\Phi_1] & \cdots & [\Phi_{L-1}] \\ [\Phi_{-1}] & [\Phi_0] & \cdots & [\Phi_{L-2}] \\ \vdots & \vdots & & \vdots \\ [\Phi_{-L+1}] & [\Phi_{-L+2}] & \cdots & [\Phi_0] \end{bmatrix} \tag{2.38b}$$

It can be seen that Φ is of Block Toeplitz structure [46] of dimensions LM by LM, and the partitions Φ_i are Toeplitz matrices of dimensions M by M, where the index i is computed as $(p-l)$, i.e.,

$$\Phi_i = \Phi_{p-l} = \{ r_{i,q-m} \}$$

$$= \{ E[x^*_{lm}(n) x_{pq}(n)] \}. \tag{2.40}$$

In matrix form, Eqn. (2.39) becomes

$$\begin{bmatrix} [\Phi_0] & [\Phi_1] & \cdots & [\Phi_{L-1}] \\ [\Phi_{-1}] & [\Phi_0] & \cdots & [\Phi_{L-2}] \\ \vdots & \vdots & & \vdots \\ [\Phi_{-L+1}] & [\Phi_{-L+2}] & \cdots & [\Phi_0] \end{bmatrix} \begin{bmatrix} \mathbf{w}_1^o \\ \mathbf{w}_2^o \\ \vdots \\ \mathbf{w}_L^o \end{bmatrix} = \begin{bmatrix} \mathbf{p}_1 \\ \mathbf{p}_2 \\ \vdots \\ \mathbf{p}_L \end{bmatrix} \tag{2.41}$$

where $\mathbf{w}_i^o$'s denote the row vectors of the optimum weight matrix $\mathbf{W}^o$, and $\mathbf{p}_i$'s are the row vectors of the cross–correlation matrix P, i.e.,

$$\mathbf{W}^{oT} = [\mathbf{w}_1^o, \mathbf{w}_2^o, \ldots, \mathbf{w}_L^o] \tag{2.42a}$$

and

$$\mathbf{P}^T = [\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_L] \tag{2.43a}$$

with

$$\mathbf{w}_i^o = [w_{i,1}, w_{i,2}, \ldots, w_{i,M}]^T \tag{2.42b}$$

and

$$\mathbf{p}_i = [p_{i,1}, p_{i,2}, \ldots, p_{i,M}]^T. \tag{2.43b}$$

As it can be seen in Eqn. (2.33), the adaptive weight elements $w_{lm}(n)$ are computed by operating on all elements of the 2–D antenna array, as denoted by the gradient $\nabla_{lm}(n)$ in Eqn. (2.35). The discrete form of the 2–D LMS algorithm, as given by Eqns. (2.33), was recently proposed in [26]. This algorithm has been

found to be cost–effective and useful in image processing, especially in data compression and image enhancement applications.

As in the case of the 1–D LMS algorithm, the step size parameter μ in Eqn. (2.33) must be chosen within the range 0 and $1/\mathrm{Tr}(\Phi)$. However, from Eqn. (2.38b), it is observed that

$$\mathrm{Tr}(\Phi) \;=\; L\ \mathrm{Tr}(\Phi_0). \tag{2.44}$$

Hence

$$0 \leq \mu \leq \frac{1}{L\ \mathrm{Tr}(\Phi_0)}. \tag{2.45}$$

2.3.2. *Relationship with the 1–D LMS algorithm*

Equation (2.33), which gives the weight updates for the 2–D case, can be derived directly from the corresponding expression for the 1–D case. We proceed by converting the weight matrix $\mathbf{W}(n)$ and the data matrix $\mathbf{X}(n)$ to LM by 1 column vectors by lexicographic ordering. These are denoted by $\mathbf{w}_v(n)$ and $\mathbf{x}_v(n)$, respectively, and given by

$$\mathbf{w}_v(n) = [w_{11}(n),\ w_{12}(n),\ \ldots, w_{1,M}(n),\ w_{2,1}(n),\ \ldots,\ w_{L,M}(n)]^T \tag{2.46}$$

and

$$\mathbf{x}_v(n) = [x_{11}(n),\ x_{12}(n),\ \ldots, x_{1,M}(n),\ x_{2,1}(n),\ \ldots,\ x_{L,M}(n)]^T \tag{2.47}$$

It follows from (2.3) that Eqn. (2.31) becomes

$$e(n) = y(n) - \mathbf{x}_v^T(n)\mathbf{w}_v(n) \tag{2.48}$$

and that

$$\mathbf{w}_v(n+1) = \mathbf{w}_v(n) - \frac{\mu}{2}\nabla_v(n) \tag{2.49}$$

in which

$$\nabla_v(n) = -2\mathbf{x}_v^*(n)e(n) \tag{2.50}$$

is the instantaneous gradient vector. Thus the optimum weight vector $\mathbf{w}_v^o$ satisfies the equation

$$\Phi_v \mathbf{w}_v^o = \mathbf{p}_v \tag{2.51}$$

where Φ_v and $\mathbf{p}_v$, respectively, are the covariance matrix of dimensions LM by LM and the correlation vector of dimensions LM by 1 given by

$$\Phi_v = \mathrm{E}[\mathbf{x}_v^*(n)\mathbf{x}_v^T(n)] \tag{2.52}$$

$$\mathbf{p}_v = \mathrm{E}[y(n)\mathbf{x}_v^*(n)]. \tag{2.53}$$

It is interesting to note that the matrix Φ_v in (2.52) is mathematically equivalent to the matrix Φ in Eqn. (2.38b), as well the weight vector of Eqn. (2.51) is mathematically equivalent to the weight matrix of Eqn. (2.41). It follows then that the analysis procedures and results for the 1–D LMS algorithm can be applied to the 2–D LMS algorithm.

2.3.3. *2–D Howells–Applebaum Algorithm*

It follows from the derivation of the 2–D LMS algorithm that adaptive

weights of the 2–D Howells–Applebaum algorithm can be expressed in a recursive form as

$$\mathbf{W}(n+1) \;=\; \mathbf{W}(n) - \tfrac{\mu}{2}\nabla_S(n) \tag{2.54}$$

where

$$\nabla_S(n) \;=\; -2\alpha \mathbf{S}^*(\theta_d,\phi_d) + 2e(n)\mathbf{X}^*(n),$$

in which

$$\mathbf{S}^*(\theta_d,\phi_d) = \{\, s^*_{lm}(\theta_d,\phi_d) \,\} = \{e^{-j\frac{2\pi}{\lambda}\{(l-1)d_x\cos\alpha_d + (m-1)d_y\cos\beta_d\}}\} \tag{2.55}$$

is the steering matrix in the direction of the desired signal, denoted by spherical angles θ_d and ϕ_d.

$$e(n) \;=\; \sum_{l=1}^{L}\sum_{m=1}^{M} w_{lm}(n)x_{lm}(n) \tag{2.56}$$

and

$$\mathbf{X}(n) \;=\; \{\, x_{lm}(n) \,\} \tag{2.57}$$

are the combined output signal, and the received data signals at the 2–D array, respectively.

Adaptive weight elements $w_{lm}(n)$ are then updated in the form

$$w_{lm}(n+1) \;=\; w_{lm}(n) - \tfrac{\mu}{2}\nabla_s(n) \tag{2.58}$$

where

$$\nabla_s(n) \;=\; -2\alpha s^*_{lm}(\theta_d,\phi_d) + 2e(n)x^*_{lm}(n) \tag{2.59}$$

Hence, optimum weights w^o_{lm} can be found by solving the equation

$$\alpha s^{*}_{lm}(\theta_d,\phi_d) = \sum_{p=1}^{L}\sum_{q=1}^{M} w^{o}_{pq} r_{p-l,q-m} \tag{2.60}$$

where $r_{p-l,q-m}$ denote elements of the covariance matrix Φ of the receiver signals which has the form of Eqn. (2.38a).

In matrix form, the optimum adaptive weight matrix $\mathbf{W}^{o}$ is found by solving

$$\begin{bmatrix} [\Phi_0] & [\Phi_1] & \cdots & [\Phi_{L-1}] \\ [\Phi_{-1}] & [\Phi_0] & \cdots & [\Phi_{L-2}] \\ \vdots & \vdots & & \vdots \\ [\Phi_{-L+1}] & [\Phi_{-L+2}] & \cdots & [\Phi_0] \end{bmatrix} \begin{bmatrix} \mathbf{w}^{o}_1 \\ \mathbf{w}^{o}_2 \\ \vdots \\ \mathbf{w}^{o}_L \end{bmatrix} = \begin{bmatrix} \mathbf{s}^{*}_1(\theta_d,\phi_d) \\ \mathbf{s}^{*}_2(\theta_d,\phi_d) \\ \vdots \\ \mathbf{s}^{*}_L(\theta_d,\phi_d) \end{bmatrix} \tag{2.61}$$

where $\mathbf{w}^{o}_i$, $\mathbf{s}^{*}_i(\theta_d,\phi_d)$ are row vectors of the adaptive weight matrix $\mathbf{W}^{o}$ and of the steering matrix $\mathbf{S}^{*}(\theta_d,\phi_d)$, respectively, i.e.,

$$\mathbf{W}^{oT} = [\mathbf{w}^{o}_1, \mathbf{w}^{o}_2, \ldots, \mathbf{w}^{o}_L] \tag{2.62}$$

and

$$\mathbf{S}^{H}(\theta_d,\phi_d) = [\mathbf{s}^{*}_1(\theta_d,\phi_d), \mathbf{s}^{*}_2(\theta_d,\phi_d), \ldots, \mathbf{s}^{*}_L(\theta_d,\phi_d)] \tag{2.63}$$

The relationship between the 2–D and 1–D Applebaum algorithms can be derived in the same manner as for the case of the LMS algorithm. Using the result derived in the last section, we substitute a LM–by–1 lexicographic ordered form of the steering vector $\alpha s^{*}_{v}(\theta_d,\phi_d)$ for the cross–correlation vector $\mathbf{r}_v$ in Eqn. (2.51).

2.3.4. *2–D Eigenvector Beam*

The SVD of the covariance matrix Φ of Eqn. (2.38b) or (2.52) is given by

$$\Phi = \mathbf{U}\Sigma\mathbf{U}^H \tag{2.64}$$

where $\mathbf{U}$ is an unitary matrix of dimensions LM by LM, whose columns are eigenvectors of the covariance Φ, i.e.,

$$\mathbf{U} = [\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_i, \ldots, \mathbf{u}_K, \mathbf{u}_{K+1}, \ldots, \mathbf{u}_{LM}] \tag{2.65}$$

and Σ is a diagonal matrix of dimensions LM by LM,

$$\Sigma = \text{diag}\,\{\lambda_1, \lambda_2, \ldots, \lambda_i, \ldots, \lambda_K, \sigma^2, \sigma^2, \ldots, \sigma^2\}. \tag{2.66}$$

Equation (2.64) can be written in the form

$$\Phi = \sum_{k=1}^{K} \lambda_k \mathbf{u}_k \mathbf{u}_k^H + \sigma^2 \sum_{i=K+1}^{LM} \mathbf{u}_i \mathbf{u}_i^H. \tag{2.67}$$

Then

$$\Phi^{-1} = \sum_{k=1}^{K} \frac{1}{\lambda_k} \mathbf{u}_k \mathbf{u}_k^H + \frac{1}{\sigma^2} \sum_{i=K+1}^{LM} \mathbf{u}_i \mathbf{u}_i^H$$

$$= \frac{1}{\sigma^2}\mathbf{I} - \frac{1}{\sigma^2} \sum_{k=1}^{K} \left(1 - \frac{\sigma^2}{\lambda_k}\right) \mathbf{u}_k \mathbf{u}_k^H \tag{2.68}$$

where $\mathbf{I}$ is a LM by LM identity matrix. It follows from Eqn. (2.51) that the

optimum weight is given by

$$\mathbf{w}_v^o = -\frac{\alpha}{\sigma^2}\mathbf{s}_v^*(\theta_d,\phi_d) - \frac{\alpha}{\sigma^2}\sum_{k=1}^{K}(1 - \frac{\sigma^2}{\lambda_k})\gamma_k\mathbf{u}_k \tag{2.69}$$

with

$$\gamma_k = \mathbf{u}_k^H\mathbf{s}_v^*(\theta_d,\phi_d).$$

Now, if the orthonormal transformation is applied to Eqn (2.69) and the constant α/σ^2 is neglected, Eqn. (2.69) becomes

$$\hat{w}_{v_i}^o = \hat{s}_{v_i}^*(\theta_d,\phi_d) - (1 - \frac{\sigma^2}{\lambda_k})\,\hat{s}_{v_k}^*(\theta_d,\phi_d) \tag{2.70}$$

where

$$i = 1,2, \ldots, LM \quad \text{and} \quad k = 1,2, \ldots, K.$$

The quantities $\hat{w}_{v_i}^o$, and $\hat{s}_{v_i}^*(\theta_d,\phi_d)$, respectively represent the transformed weight vector $\mathbf{w}_v^o$ and the transformed steering vector $\mathbf{s}_v^*(\theta_d,\phi_d)$ corresponding to the ith eigenvector, i.e.,

$$\hat{w}_{v_i}^o = \mathbf{u}_i^H\mathbf{w}_v^o \tag{2.71}$$

and

$$\hat{s}_{v_i}^*(\theta_d,\phi_d) = \mathbf{u}_i^H\mathbf{s}_v^*(\theta_d,\phi_d). \tag{2.72}$$

The adapted beam pattern for the 2–D array which is denoted by $G_a(\theta,\phi,n)$ is given by

$$G_a(\theta,\phi,n) = \mathbf{s}_v^T(\theta,\phi)\mathbf{w}_v(n) = \hat{\mathbf{s}}_v^T(\theta,\phi)\hat{\mathbf{w}}_v(n)$$

$$= \sum_{i=1}^{LM} \hat{w}_{v_i}(n)\hat{s}_{v_i}(\theta,\phi). \tag{2.73}$$

Now, by substituting $\hat{w}_{v_i}^o$ from (2.70) for $\hat{w}_{v_i}(n)$ in (2.73), the steady state of adapted pattern can be expressed as

$$G_a(\theta,\phi) = G_q(\theta,\phi) - \sum_{k=1}^{K} (1 - \frac{\sigma^2}{\lambda_k}) g_k(\theta,\phi), \tag{2.74}$$

where $G_q(\theta,\phi)$ is the quiescent pattern of the array, which is defined by

$$G_q(\theta,\phi) = \mathbf{s}_v^T \mathbf{s}_v^* = \sum_{i=1}^{LM} s_{v_i}(\theta,\phi) s_{v_i}^*(\theta_d,\phi_d)$$

$$= \sum_{l=1}^{L} \sum_{m=1}^{M} s_{lm}(\theta,\phi) s_{lm}^*(\theta_d,\phi_d), \tag{2.75}$$

in which

$$s_{lm}(\theta,\phi) = s_{v_i}(\theta,\phi)$$

$$= e^{j\frac{2\pi}{\lambda}\{(l-1)d_x \sin\theta\cos\phi + (m-1)d_y \sin\theta\sin\phi\}} \tag{2.76a}$$

and

$$s_{lm}^*(\theta_d,\phi_d) = s_{v_i}^*(\theta_d,\phi_d)$$

$$= e^{-j\frac{2\pi}{\lambda}\{(l-1)d_x \sin\theta_d\cos\phi_d + (m-1)d_y \sin\theta_d\sin\phi_d\}}. \tag{2.76b}$$

The quantity $g_k(\theta,\phi)$ in Eqn. (2.74) is the 2–D eigenvector beam pattern corresponding to the kth source, which is given by

$$g_k(\theta,\phi) \;=\; g_{k,l,m}(\theta,\phi) \;=\; \hat{s}^{*}_{v_k}(\theta_d,\phi_d)\;\hat{s}_{v_k}(\theta,\phi). \tag{2.77}$$

Equation (2.74) can be interpreted in such a manner as to reach a similar conclusion to that arrived at for the 1–D case, i.e. the adapted beam pattern of a 2–D array is formed by subtracting the quiescent beam pattern from the retrodirective eigenvector beam pattern. The quiescent pattern is represented by the first term in RHS of the equation. Moreover, from Eqn. (2.74) it follows that an eigenbeam pattern is formed to point at each of the interfering sources and that the nulling capability of the eigenbeam pattern is dependent upon the jammer power, denoted by the eigenvalue λ_k.

Note that eigenvectors $\mathbf{u}_i$ of the covariance matrix Φ can be written as

$$\mathbf{u}_i^T \;=\; [u_{i,1,1},\; u_{i,1,2},\; \cdots,\; u_{i,1,M},\; u_{i,2,1},\; \cdots,\; u_{i,L,M}] \tag{2.78a}$$

$$\;=\; [\mathbf{u}_{i,1}^T,\; \mathbf{u}_{i,2}^T,\; \cdots,\; \mathbf{u}_{i,l}^T,\; \cdots,\; \mathbf{u}_{i,L}^T] \tag{2.78b}$$

where $\mathbf{u}_{i,l}$ are M by 1 vectors, i.e.,

$$\mathbf{u}_{i,l} \;=\; [u_{i,l,1},\; u_{i,l,2},\; \cdots,\; u_{i,l,M}]^T. \tag{2.78c}$$

It follows that the row vectors of the adaptive weight matrix $\mathbf{W}^O$ in Eqn. (2.61) satisfy the equation

$$\begin{bmatrix} \mathbf{w}_1^o \\ \mathbf{w}_2^o \\ \vdots \\ \mathbf{w}_L^o \end{bmatrix} = \frac{\alpha}{\sigma^2}\begin{bmatrix} \mathbf{s}_1^*(\theta_d,\phi_d) \\ \mathbf{s}_2^*(\theta_d,\phi_d) \\ \vdots \\ \mathbf{s}_L^*(\theta_d,\phi_d) \end{bmatrix} - \frac{\alpha}{\sigma^2}\sum_{k=1}^{K}\sum_{p=1}^{L}(1-\frac{\sigma^2}{\lambda_k})\gamma_{k,p}\begin{bmatrix} \mathbf{u}_{k,1} \\ \mathbf{u}_{k,2} \\ \vdots \\ \mathbf{u}_{k,L} \end{bmatrix} \quad (2.79a)$$

or

$$\mathbf{w}_l^o = \frac{\alpha}{\sigma^2}\mathbf{s}_l^*(\theta_d,\phi_d) - \frac{\alpha}{\sigma^2}\sum_{k=1}^{K}\sum_{p=1}^{L}(1-\frac{\sigma^2}{\lambda_k})\gamma_{k,p}\mathbf{u}_{k,l} \quad (2.79b)$$

with the constant

$$\gamma_{k,p} = \mathbf{u}_{k,p}^H \mathbf{s}_p^*(\theta_d,\phi_d).$$

And adaptive weight elements w_{lm}^o are

$$w_{lm}^o = \frac{\alpha}{\sigma^2}s_{lm}^*(\theta_d,\phi_d) - \frac{\alpha}{\sigma^2}\sum_{k=1}^{K}\sum_{p=1}^{L}(1-\frac{\sigma^2}{\lambda_k})\gamma_{k,p}u_{k,lm}. \quad (2.79c)$$

It should be noted that Eqn. (2.79a) is mathematically equivalent to the Eqn. (2.69). However, Eqn. (2.79a) is expressed in matrix form whereas Eqn. (2.69) is in vector form.

To further interpret the relationship between 2–D weight vectors and 1–D weight vectors, consider the following theorem.

Theorem: The eigenvector beam pattern $g_k(\theta,\phi)$ for a 2–D array can be formed by using two independent 1–D eigenbeams, which are eigenvector beams along the

rows and columns of the 2–D array, respectively, i.e.,

$$g_k(\theta,\phi) \;=\; g_{k,l,m}(\theta,\phi) \;=\; g_{k,l}(\theta,\phi)g_{k,m}(\theta,\phi). \tag{2.80}$$

Proof: It follows from Eqn. (2.70) that adaptive weights along row l and column m of a 2–D are given by

$$\hat{w}_l^o \;=\; \hat{s}_l^*(\theta_d,\phi_d) - (1 - \frac{\beta_0}{\beta_{k,l}})\hat{s}_{k,l}^*(\theta_d,\phi_d) \tag{2.81a}$$

$$\hat{w}_m^o \;=\; \hat{s}_m^*(\theta_d,\phi_d) - (1 - \frac{\beta_0}{\beta_{k,m}})\hat{s}_{k,m}^*(\theta_d,\phi_d) \tag{2.81b}$$

where β_0 is a constant, and $\beta_{k,l}$, $\beta_{k,m}$ are eigenvalues for the covariance matrix formed using row l and column m of the 2–D array corresponding to the kth source. Also, in Eqns. (2.81a) and (2.81b)

$$\hat{s}_{k,l}^*(\theta_d,\phi_d) \;=\; \hat{s}_l^*(\theta_d,\phi_d)$$

and

$$\hat{s}_{k,m}^*(\theta_d,\phi_d) \;=\; \hat{s}_m^*(\theta_d,\phi_d)$$

for $l = 1, 2, \ldots, K$, and $m = 1, 2, \ldots, K$.

The adapted beam pattern can be expressed as

$$G_a(\theta,\phi) \;=\; \sum_{l=1}^{L} \sum_{m=1}^{M} \hat{w}_{lm}^o \; \hat{s}_{lm}(\theta,\phi). \tag{2.82}$$

It follows from the orthogonal transformation of Eqns. (2.71) and (2.72) that

$$\hat{w}^{o}_{lm} = \hat{w}^{o}_{l}\,\hat{w}^{o}_{m} \tag{2.83}$$

and

$$\hat{s}_{lm}(\theta,\phi) = \hat{s}_{l}\,\hat{s}_{m} \tag{2.84}$$

in which

$$\hat{s}_{l} = \mathbf{u}^{T}_{k,l}\mathbf{s}_{l} \quad \text{and} \quad \hat{s}_{m} = \mathbf{u}^{T}_{k,m}\mathbf{s}_{m}$$

where $\mathbf{u}_{l}$, $\mathbf{u}_{m}$ denote the eigenvectors of the covariance matrix formed using row l and column m of the 2-D array, respectively. And $\mathbf{s}_{l}$ and $\mathbf{s}_{m}$ are steering vectors, which have the form

$$\mathbf{s}_{l}(\theta,\phi) = e^{j(l-1)\varphi_{x}}\,[1,\ e^{j\varphi_{y}},\ e^{j2\varphi_{y}},\ \ldots,\ e^{j(M-1)\varphi_{y}}]^{T} \tag{2.85a}$$

and

$$\mathbf{s}_{m}(\theta,\phi) = e^{j(m-1)\varphi_{y}}\,[1,\ e^{j\varphi_{x}},\ e^{j2\varphi_{x}},\ \ldots,\ e^{j(L-1)\varphi_{x}}]^{T} \tag{2.85b}$$

with

$$\varphi_{x} = \frac{2\pi}{\lambda}d_{x}\sin\theta\cos\phi \quad \text{and} \quad \varphi_{y} = \frac{2\pi}{\lambda}d_{y}\sin\theta\sin\phi.$$

Then by substituting Eqns. (2.81a,b) into (2.82) and performing some simple manipulations, we obtain

$$G_{a}(\theta,\phi) = G_{q}(\theta,\phi) - \sum_{k=1}^{K}\{1 - \frac{\beta_{0}^{2}}{\beta_{k,l}\beta_{k,m}}\}g_{k,l}(\theta,\phi)g_{k,m}(\theta,\phi) \tag{2.86}$$

$$\text{for } l = 1,2,\ \ldots,\ L, \text{ and } m = 1,2,\ \ldots,M,$$

where

$$g_{k,l}(\theta,\phi) = \hat{s}^{*}_{l}(\theta_{d},\phi_{d})\ \hat{s}_{l}(\theta,\phi) \tag{2.87a}$$

and

$$g_{k,m}(\theta,\phi) = \hat{s}_m^*(\theta_d,\phi_d)\ \hat{s}_m(\theta,\phi) \tag{2.87b}$$

are eigenvector beams for row l and column m, corresponding to the kth source, respectively.

Moreover, since λ_k is the eigenvalue of the covariance matrix Φ of dimensions LM by LM and $\beta_{k,l}$, $\beta_{k,m}$ are respectively the eigenvalues of the covariance matrices Φ_l of dimensions L by L and Φ_m of dimensions M by M, therefore,

$$\beta_0 = \sigma^2$$

and

$$\beta_{k,l}\beta_{k,m} = \lambda_{lm} = \lambda_k.$$

Also, since the second term on the RHS of Eqn. (2.86) is equivalent to that of Eqn. (2.74), it is concluded that the 2–D eigenbeam $g_k(\theta,\phi)$ corresponding to the kth source can be formed by using the two 1–D eigenbeams which correspond to row l and column m of the 2–D array, i.e,

$$g_k(\theta,\phi) = g_{k,l}(\theta,\phi)g_{k,m}(\theta,\phi).$$

Furthermore, it follows from Eqn. (2.86) that with a single desired beam a 2–D adaptive beamformer is capable of nulling up to LM–1 jammers. In other words, there are up to LM–1 degrees of freedom for a 2–D adaptive array.

2.4. Concluding Remarks

In this Chapter, some of the classical adaptive beamforming algorithms that have been derived for linear arrays have been extended for planar arrays. The formulation for the 2–D algorithms followed closely from the theory of 1–D beamforming algorithms. In large measure the 2–D algorithms are natural extensions of corresponding 1–D algorithms. The 2–D LMS and 2–D Applebaum algorithms have been presented. It has been shown that in the steady state the optimum weights for 2–D adaptive arrays satisfy 2–D Weiner–Hopf equation. The relationship between the 2–D and 1–D adaptive algorithms was derived. It has been shown that by rearranging the data from an 2–D array into a lexicographic order the two algorithms are mathematically equivalent to each other.

The concept of 2–D eigenvector beams was introduced to interpret the interference nulling performance of the 2–D adaptive beamforming techniques. It has been further shown that the eigenvector beam of a 2–D array can be formed by using two independent 1–D eigenbeams, which respectively are formed by operating on the rows and columns of the 2–D array.

CHAPTER 3

MODERN ADAPTIVE BEAMFORMING TECHNIQUES

3.1. Introduction

The major problem underlining the performance of the closed–loop algorithms, typified by the LMS algorithm, is one of poor convergence. In comparison, the SMI algorithm is cumbersome and numerically unstable. An alternative approach to the minimum least–squares, i.e. an open–loop algorithm, which is particularly good in the numerical sense is that of orthogonal triangularization [47]. This is typified by the method known as QR decomposition (QRD) which involves the application of a sequence of unitary transformations to reduce the measured data matrix to an upper triangular form.

Recently, with the growth of VLSI technology, workers have proposed using systolic arrays to solve the recursive least–squares problem, which is based on the QRD algorithm [22], [48]. In particular, the systolic array represents a highly pipelined and parallel architecture. Thus, an adaptive beamformer implemented with systolic arrays is potentially capable of performing real–time processing. In fact, systolic array processors have shown much promise in adaptive beamforming applications [6–8].

In this Chapter, the adaptive beamforming problem is formulated as a

recursive least-squares problem and is solved using the QRD algorithm. The adaptive beamformer is implemented using an efficient pipelined architecture in the form of a triangular systolic array. Adaptive beamforming for linear arrays and planar arrays (1–D and 2–D) are presented. Linearly constrained adaptive beamforming will be also given. It is further shown that in the 2–D case the systolic array implementation is arranged in a 3–D form, in which PEs are highly pipelined and locally interconnected. With a proper data flow in time skew format, adaptation along rows and columns of a 2–D array can be computed concurrently and simultaneously.

3.2. QRD–LS Algorithm and Adaptive Beamforming

3.2.1. *QR Decomposition and Givens Rotation*

Equation (1.1) may be implemented by using the QRD algorithm as follows. If we premultiply both sides of Eqn. (1.1) with a composite unitary transformation, $\mathbf{Q}(n)$, we obtain

$$\mathbf{Q}(n)\mathbf{e}(n) = \mathbf{Q}(n)\mathbf{y}(n) - \mathbf{Q}(n)\mathbf{X}(n)\mathbf{w}(n) \tag{3.1}$$

$$= \begin{bmatrix} \mathbf{u}(n) \\ \mathbf{v}(n) \end{bmatrix} - \begin{bmatrix} \mathbf{R}(n) \\ \mathbf{O} \end{bmatrix} \mathbf{w}(n) \tag{3.2}$$

where $\mathbf{Q}(n)$ is an n by n matrix, $\mathbf{u}(n)$, $\mathbf{v}(n)$ are column vectors of dimensions $M\times 1$ and $(n-M)\times 1$, respectively. $\mathbf{R}(n)$ is an M by M upper triangular matrix, and $\mathbf{O}$ is an $(n-M)$ by $(n-M)$ null matrix. Note that multiplication by $\mathbf{Q}(n)$ in Eqn. (3.1) does not alter the least-squares minimization process, because

$$\|\mathbf{Q}(n)\mathbf{e}(n)\|^2 = \|\mathbf{e}(n)\|^2 \tag{3.3}$$

It follows from Eqn. (3.2) that the condition for a minimum in the residual error is

$$\mathbf{u}(n) - \mathbf{R}(n)\mathbf{w}(n) = \mathbf{0} \tag{3.4}$$

and hence

$$\|\mathbf{e}(n)\|^2 = \|\mathbf{v}(n)\|^2. \tag{3.5}$$

Equation (3.4) therefore defines the least–squares solution for the weight vector and can be easily solved by back substitution, which is much easier than trying to solve the Wiener–Hopf equation described in Chapter 2. Eqn. (3.4) is also much better conditioned since the condition number of $\mathbf{R}(n)$ is given by

$$C(\mathbf{R}(n)) = C(\mathbf{Q}(n)\mathbf{X}(n)) = C(\mathbf{X}(n)). \tag{3.6}$$

This property follows directly from the fact that $\mathbf{Q}(n)$ is unitary.

The triangularization process of Eqn. (3.1) can be carried out using either the Householder transformations [47] or the Givens rotations [49]. The Givens rotation method however has been found to be particularly suitable for adaptive antenna applications since the triangularization process is recursively updated as each new row of data enters the computation [22]. With the Givens rotation method, the triangularization is performed by a process whereby coefficients of the the lower half of the data matrix are successively eliminated by manipulating pairs of row vectors. An elementary transformation of the Givens rotation is of the form

$$\begin{bmatrix} C & S^* \\ -S & C \end{bmatrix} \begin{bmatrix} 0 \ldots 0, & r_i, & r_j \ldots, & r_k, \ldots \\ 0 \ldots 0, & x_i, & x_j \ldots, & x_k, \ldots \end{bmatrix} = \begin{bmatrix} 0 \ldots 0, & r_i', & r_j' \ldots, & r_k', \ldots \\ 0 \ldots 0, & 0, & x_j' \ldots, & x_k', \ldots \end{bmatrix} \tag{3.7}$$

where the rotation coefficients, C and S, satisfy

$$-Sr_i + Cx_i = 0 \tag{3.8a}$$

$$S^* S + C^* C = 1 \tag{3.8b}$$

$$C^* = C \tag{3.8c}$$

from which the rotation coefficients may be specified as

$$C = \frac{|r_i|}{(|x_i|^2 + |r_i|^2)^{1/2}} \tag{3.9a}$$

and

$$S = \frac{x_i}{r_i} C. \tag{3.9b}$$

3.2.2. *Recursive QRD–LS Algorithm*

An important feature of the QR method using the Givens rotations is the ability to apply the algorithm in a recursive form in which the elements of the upper triangular matrix can be updated on a sample by sample basis as each new row of data enters the computation [22], [48]. In the recursive problem, a sequence of elementary transformations may be used to triangularize the matrix X(n) as follows.

Assume that the matrix $\mathbf{X}(n-1)$ has already been reduced to triangular form by the unitary transformation, i.e.,

$$\mathbf{Q}(n-1)\mathbf{X}(n-1) = \begin{bmatrix} \mathbf{R}(n-1) \\ \mathbf{O} \end{bmatrix} \tag{3.10}$$

with $\mathbf{O}$ is $(n-M-1)$ by M null matrix.

Now define the unitary matrix

$$\overline{\mathbf{Q}}(n-1) = \left[\begin{array}{c|c} \mathbf{Q}(n-1) & \mathbf{0} \\ \hline \mathbf{0}^T & 1 \end{array}\right] \tag{3.11}$$

with $\mathbf{0}$ is a $(n-1)$ by 1 null vector. It then follows that

$$\overline{\mathbf{Q}}(n-1)\mathbf{X}(n) = \overline{\mathbf{Q}}(n-1)\begin{bmatrix} \mathbf{X}(n-1) \\ \mathbf{x}^T(n) \end{bmatrix} = \begin{bmatrix} \mathbf{R}(n-1) \\ \mathbf{O} \\ \mathbf{x}^T(n) \end{bmatrix} \tag{3.12}$$

and thus the triangularization process may be completed by the following sequence of operations. Rotate the snapshot vector $\mathbf{x}^T(n)$ with the first row of $\mathbf{R}(n-1)$ so that the leading element of $\mathbf{x}^T(n)$ is eliminated producing a reduced vector $\mathbf{x}^{T'}(n)$. Then rotate the vector $\mathbf{x}^{T'}(n)$ with the second row of $\mathbf{R}(n-1)$ so that the leading element of $\mathbf{x}^{T'}(n)$ is eliminated and so on until every element has been eliminated. The resulting triangular matrix $\mathbf{R}(n)$ then corresponds to a complete triangularization of the matrix $\mathbf{X}(n)$ as define in Eqn. (3.1), i.e.,

$$\mathbf{G}(n)\begin{bmatrix} \mathbf{R}(n-1) \\ \mathbf{O} \\ \mathbf{x}^T(n) \end{bmatrix} = \begin{bmatrix} \mathbf{R}(n) \\ \mathbf{O} \\ \mathbf{0}^T \end{bmatrix} \tag{3.13}$$

where $\mathbf{G}(n)$ is a unitary matrix, representing the combined effects of Givens rotations. The matrix $\mathbf{G}(n)$ is simply the product of the elementary rotations g_i (i=1,2,..., M) which are needed to update the existing triangular system with the introduction of new data, i.e.,

$$\mathbf{G}(n) = g_M \cdots g_2 g_1 \tag{3.14}$$

in which

$$g_i = \begin{bmatrix} 1 & & & & & & \\ & \ddots & & & & \mathbf{0} & \\ & & 1 & & & & \\ & & & C_i & & & S_i^* \\ & & & & 1 & & \\ & \mathbf{0} & & -S_i & & 1 & C_i \end{bmatrix} \tag{3.15}$$

where the only nonzero off-diagonal elements occur in the last row and last column. Consequently, the corresponding unitary matrix $\mathbf{Q}(n)$ is simply given by the recursive expression

$$\mathbf{Q}(n) = \mathbf{G}(n)\overline{\mathbf{Q}}(n-1). \tag{3.16}$$

Equation (3.16) shows that, given the $(n-1)\times(n-1)$ unitary matrix $\mathbf{Q}(n-1)$ that performs the QR-decomposition of the data matrix at time $n-1$ and given the sequence of Givens rotations represented by the unitary matrix $\mathbf{G}(n)$, the updated unitary matrix $\mathbf{Q}(n)$ may be computed.

In the same manner, the desired vector $\mathbf{y}(n)$ can be partitioned as follows. Let

$$\mathbf{y}(n) = \begin{bmatrix} \mathbf{y}(n-1) \\ y(n) \end{bmatrix} \tag{3.17}$$

then

$$\mathbf{Q}(n)\mathbf{y}(n) = \mathbf{G}(n)\begin{bmatrix} \mathbf{Q}(n-1)\mathbf{y}(n-1) \\ \mathbf{y}(n) \end{bmatrix}$$

$$= \mathbf{G}(n)\begin{bmatrix} \mathbf{u}(n-1) \\ \mathbf{v}(n-1) \\ \mathbf{y}(n) \end{bmatrix} = \begin{bmatrix} \mathbf{u}(n) \\ \mathbf{v}(n-1) \\ \alpha(n) \end{bmatrix} = \begin{bmatrix} \mathbf{u}(n) \\ \mathbf{v}(n) \end{bmatrix}. \tag{3.18}$$

Equation (3.18) shows that both the vectors $\mathbf{u}(n)$ and $\mathbf{v}(n)$ can be updated recursively using the same sequence of Givens rotations, $\mathbf{G}(n)$, that is used to update $\mathbf{R}(n)$. The least–squares weight vector $\mathbf{w}(n)$ may then be computed using back substitution in Eqn. (3.4). Note that the solution is not defined if $n < M$ but the recursive triangularization procedure may be initialized by setting $\mathbf{R}(0) = \mathbf{O}$ and $\mathbf{u}(0) = 0$. The algorithm defined by Eqns. (3.4), (3.14), and (3.16) is referred to as the recursive QR decomposition least–squares (QRD–LS) algorithm.

3.2.3. *Use of QRD–LS Algorithm in Adaptive Beamforming*

In the adaptive beamforming problem as described in Eqn. (1.1), the least–squares weight vector $\mathbf{w}(n)$ is not of direct interest since the main objective is to compute the least–squares residual power $\|\mathbf{e}(n)\|^2$ at the beamforming output. In a recent paper [22], a modified version of the recursive QRD–LS algorithm is described in which the least–squares residual is produced directly at each stage of the recursive process without computing the weight vector explicitly. The modified algorithm is more r. ecause it avoids the solution of a system of linear

equations which could be ill–conditioned. Also, since the back substitution processor and the separate beamforming network are no longer needed, the algorithm offers a significant reduction in the complexity of the subsequent hardware implementation.

The derivation of the updated residual error $\mathbf{e}(n)$ at the beamforming output can be summarized as follows. From the results in the last section, by combining Eqns. (3.13) and (3.18) the new result becomes

$$\mathbf{Q}(n)\mathbf{e}(n) = \begin{bmatrix} \mathbf{u}(n) \\ \mathbf{v}(n-1) \\ \alpha(n) \end{bmatrix} - \begin{bmatrix} \mathbf{R}(n) \\ \mathbf{O} \end{bmatrix} \mathbf{w}(n). \tag{3.19}$$

Since the weight vector must satisfy Eqn. (3.4), it follows that the residual error vector is given by

$$\mathbf{Q}(n)\mathbf{e}(n) = \mathbf{G}(n)\overline{\mathbf{Q}}(n-1)\begin{bmatrix} \mathbf{e}(n-1) \\ e(n) \end{bmatrix} = \begin{bmatrix} \mathbf{O} \\ \mathbf{v}(n-1) \\ \alpha(n) \end{bmatrix} \tag{3.20}$$

Now, it is noted that since $\mathbf{G}(n)$ is unitary (Eqn. (3.14)), so

$$\overline{\mathbf{Q}}(n-1)\begin{bmatrix} \mathbf{e}(n-1) \\ e(n) \end{bmatrix} = \mathbf{G}^{H}(n)\begin{bmatrix} \mathbf{O} \\ \mathbf{v}(n-1) \\ \alpha(n) \end{bmatrix}. \tag{3.21}$$

To derive the residual error $e(n)$, consider the product of the bottom row of the individual matrices $\mathbf{g}_i^H$ (Eqn. (3.15)) with the column vector containing $\mathbf{v}(n-1)$ and $\alpha(n)$ in Eqn. (3.21). By inspection, the current least–squares residual is given by

$$\epsilon(n) = \gamma(n).\alpha(n) \tag{3.22}$$

where

$$\gamma(n) = \prod_{i=1}^{M} C_i \tag{3.23}$$

is the product of all cosine parameters produced during the sequence of Givens rotations used to eliminate the vector $\mathbf{x}^T(n)$.

3.2.4. *Systolic Array Implementation*

Triangular Least–Squares Array

The systolic array configuration that is well suited for solving the least–squares problem of Eqn. (1.1), based on the use of the QR–decomposition and Givens rotations, is shown in Fig. 3.1. This was proposed by [22] and is a modification of an earlier processor [48]. The systolic array's structure takes the form of a triangle and consists of three types of processing elements (PEs), which are: (1) boundary cells, (2) internal cells, and (3) a final processing cell. The first two types of cells perform Givens rotations and store processed output data while the final processing cell is simply a multiplier. Computations performed by these PE cells are given in Fig. 3.2a. It is to be noted in Fig. 3.1 that the vectors $\mathbf{u}(n)$ and $\mathbf{v}(n)$ are stored in the internal cells located on the right–hand side of the array; the boundary cells and other internal cells stores the upper triangular matrix $\mathbf{R}(n)$, and the final processing cell computes the residual error vector $\mathbf{e}(n)$. In addition, it can seen that the flow of input data is arranged in time–skew format to compensate for propagation delays. These are a consequence of the manner in

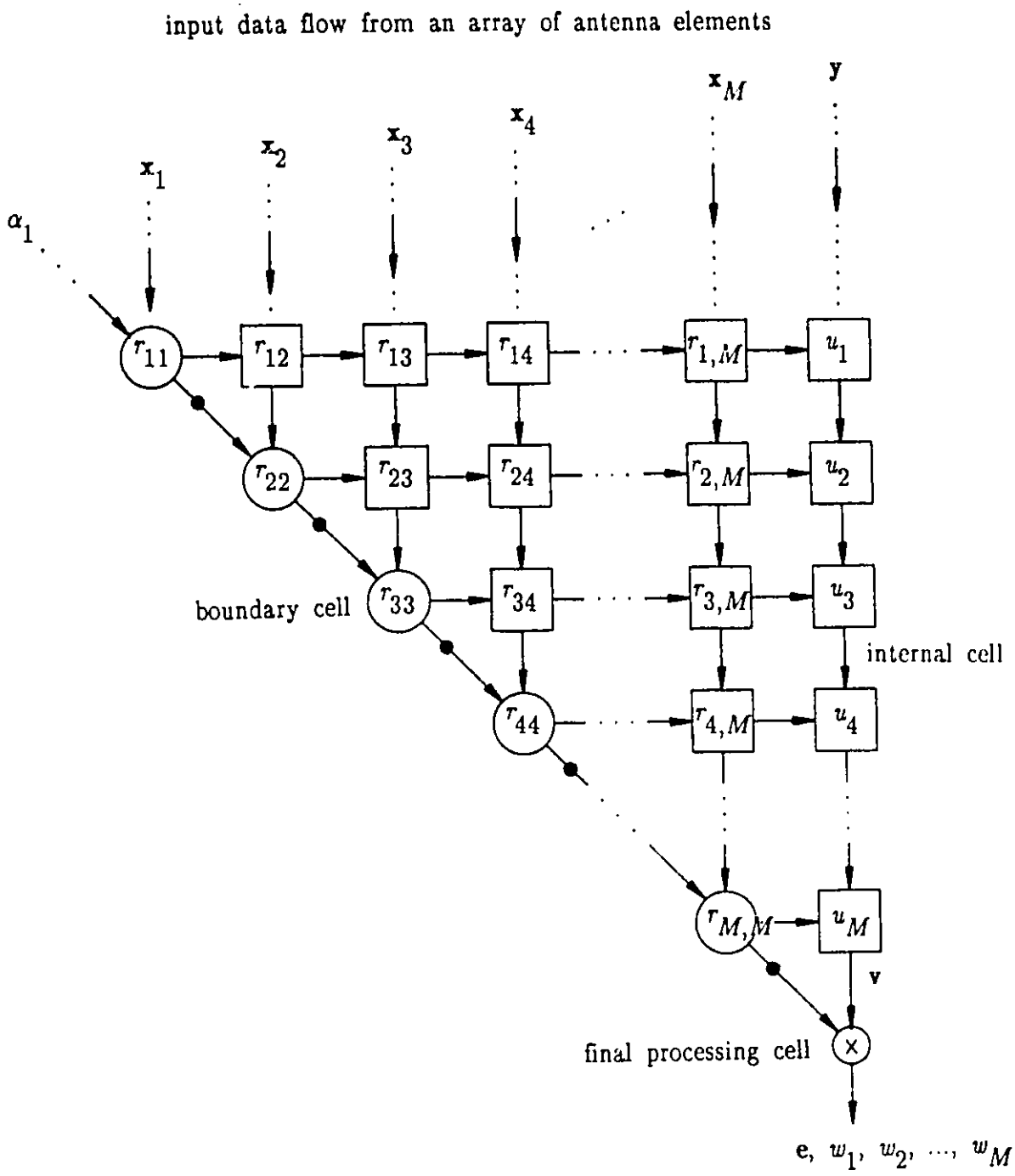
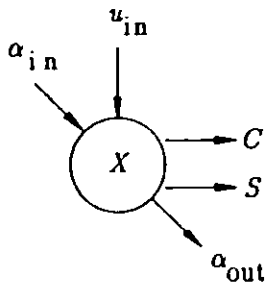


Fig. 3.1 Triangular systolic implementation for adaptive beamforming

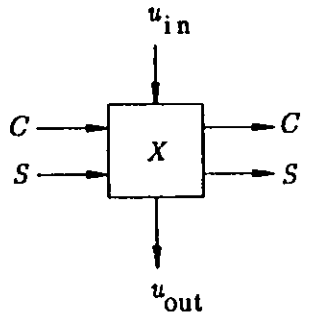
a)

Boundary Cell



if $u_{in} = 0$ then
$C \leftarrow 1,\ S \leftarrow 0$
$\alpha_{out} \leftarrow \alpha_{in}$
else
$X' \leftarrow (X^2 + |u_{in}|^2)^{\frac{1}{2}}$
$S \leftarrow u_{in}/X'$
$C \leftarrow |X|/|X'|$
$\alpha_{out} \leftarrow C\alpha_{in}$

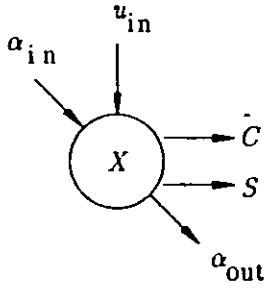
Internal Cell



$u_{out} \leftarrow Cu_{in} - SX$
$X' \leftarrow CX + S^* u_{in}$

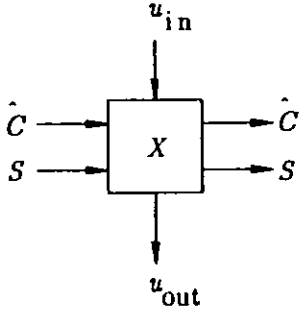
b)

Boundary Cell



$\alpha_{in} = 1$
$\hat{C} \leftarrow u_{in}$
if $\hat{C} = 0$, $S \leftarrow 0$, $\alpha_{out} \leftarrow \alpha_{in}$
else
$X' \leftarrow X + \alpha_{in}|\hat{C}|^2$
$S \leftarrow \alpha_{in} u_{in}/X'$
$\alpha_{out} \leftarrow X\alpha_{in}/X'$

Internal Cell



$u_{out} \leftarrow u_{in} - \hat{C}X$
$X' \leftarrow X + S^* u_{out}$

Fig. 3.2 Computations performed by processing elements.
a) Givens rotations b) square-root-free

which the data are processed as they pass through the PEs. The array is controlled by means of a global and synchronous clock. As well, additional delays are added to the signals which move form one boundary cell to another. These delays are indicated by the black dots in Fig. 3.1.

Square–Root–Free Algorithm

If one makes use of the square–root–free algorithm [50], the upper triangular matrix $\mathrm{R}(n)$ can be factorized as follows

$$\mathrm{R}(n) = \mathrm{D}^{1/2}(n)\mathrm{K}(n) \tag{3.24}$$

where

$$\mathrm{D}^{1/2}(n) = \mathrm{diag}\{\ \mathrm{R}(n)\ \} \tag{3.25}$$

and $\mathrm{K}(n)$ is an upper triangular matrix with elements in the diagonal equal to one. In this version of the algorithm, the elements of the matrix $\mathrm{K}(n)$ are stored in both the boundary cells and the internal cells of Fig. 3.1, and those of the diagonal matrix $\mathrm{D}^{1/2}(n)$ are stored in the boundary cells alone . The computations that are performed by the processing cells, when using the square–root–free algorithm, are given in Fig. 3.2b.

It is worth noting that the systolic structure of Fig. 3.1 is primarily used to compute the residual error $\|\mathrm{e}(n)\|^2$. However, it can also be used to extract the weight vector $\mathrm{w}(n)$. This can be achieved at any time after initialization of the upper triangular matrix $\mathrm{R}(n)$ takes place (i.e., $n \geq 2M$) with no additional control of the array required. If, for example, at time t_n the update of $\mathrm{R}(n)$ is suppressed

by setting $\alpha_1 = 0$ with the input vectors becoming

$$y(n) = 0,\ x_i(n) = 1,\ \text{and}\ x_j(n) = 0 \text{ if } j \neq i;$$

then Eqn. (1.8) becomes

$$e(n) = -w_i(n), \tag{3.26}$$

from which the tap weights follow directly. This is the so–called weight flushing method for extracting the weight vector [7]. It does not require the use of a linear systolic section as proposed in [48]. In fact, the weight flushing can be easily carried out by using the QRD–LS systolic structure which is implemented using the square–root–free form of the Givens rotation algorithm. It follows from Figs. 3.1 and 3.2b that when α_1 is set equal to 0, all elements of matrix R(n) are frozen, i.e., unadapted. At the moment that α_1 is reset to 1, the adaptation process will resume, and will not be affected thereafter by the weight flushing procedure that had been implemented previously.

Adaptive Beamformer with a Clamped Element

The systolic structure of Fig. 3.1 has been applied to antenna arrays with the primary signal being derived by using a separate high gain antenna [6,7]. When the high gain antenna is not available for supplying the primary signal, one of the elements of the antenna array can be used in its place to derive the reference signal. For example, if element 1 is selected as a primary element having its weight coefficient, $w_1(n)$, constrained to unity, then other (M–1) elements are used as auxiliary elements. The least–squares problem of adaptive beamforming can be expressed as

$$\mathbf{e}(n) = \mathbf{X}(n)\mathbf{w}(n)$$

$$= \mathbf{x}_1(n) + \tilde{\mathbf{X}}(n)\tilde{\mathbf{w}}(n) \tag{3.27}$$

which is similar to that of Eqn. (1.1).

In Eqn. (3.27) the data matrix $\tilde{\mathbf{X}}(n)$ and the weight vector $\tilde{\mathbf{w}}(n)$ are defined as follows

$$\tilde{\mathbf{X}}(n) = [\mathbf{x}_2(n),\ \mathbf{x}_3(n),\ \ldots,\ \mathbf{x}_M(n)] \tag{3.28}$$

and

$$\tilde{\mathbf{w}}(n) = [w_2(n),\ w_3(n),\ \ldots,\ w_M(n)]. \tag{3.29}$$

Note that, in this case, prior to adaptation the overall beam pattern of the adaptive processor is solely determined by the response of the primary "clamped" element x_1. This configuration provides no inherent mechanism to inhibit the adaptive process from nulling the desired signal. In fact, the system is only allowed to adapt when the desired signal is absent. When it is present the adaptive weight vector is frozen thus allowing signal reception. Note that in this case only M–1 weights are computed. Therefore the M–element array has M–1 degrees of freedom.

3.3. *Linearly Constrained Adaptive Beamforming*

The triarray processor described in the last section, which minimizes the

least–squares norm of the residual vector $\mathbf{e}(n)$, is only applied to unconstrained adaptive beamformers. In these systems the primary signal constitutes the output from either a main (high gain) antenna or a clamped element of the array, in which the desired signal is assumed to be absent during the adaptation process. A particularly important application of adaptive arrays requires the power of a combined signal

$$\mathbf{e}(n) = \mathbf{X}(n)\mathbf{w}(n) \tag{3.30}$$

to be minimized subject to P linear beam constraints of the form

$$\mathbf{C}\mathbf{w}(n) = \mu. \tag{3.31}$$

In Eqn. (3.31), C and μ, are respectively the constraint matrix and the gain vector. They are defined as

$$\mathbf{C}^T = [\mathbf{c}(\theta_1), \mathbf{c}(\theta_2), \ldots, \mathbf{c}(\theta_P)] \tag{3.32}$$

in which, the constraint steering vector, $\mathbf{c}(\theta_p)$ has the form

$$\mathbf{c}^T(\theta_p) = [1, e^{j\varphi_p}, e^{j2\varphi_p}, \ldots, e^{j(M-1)\varphi_p}] \tag{3.33}$$

with

$$\varphi_p = \frac{2\pi d}{\lambda} \sin\theta_p$$

and the constraint gain vector is given by

$$\mu^T = [\mu_1, \mu_2, \ldots, \mu_P]. \tag{3.34}$$

3.3.1. *Pre-processing Technique*

Several techniques have been proposed in the literature to implement the adaptive beamforming problem with multiple linear constraints. In all cases, however, the resulting implementation is extremely cumbersome [17], [28], [9]. In this section, the problem is formulated in a much simpler manner, which is easily implemented using the triarray least-squares processor given in Fig. 3.1. The result presented follows directly the development described in [54]. It is to be noted that each constraint vector $\mathbf{c}(\theta_p)$ constitutes the beam steering direction of the desired signal. Arbitrary linear constraints have been studied extensively in [51].

The constraint matrix C may be reduced to an upper trapezoidal matrix C' by using the QR decomposition. Denoting the orthogonal transformation matrix by $\mathbf{Q}_1$, then

$$\mathbf{Q}_1[\mathbf{C}\mathbf{w}(n) - \mu] = \mathbf{C}'\mathbf{w}(n) - \mu' = 0. \tag{3.35}$$

Consider the first transformed constraint equation

$$\mathbf{c}_1'^{T}\mathbf{w}(n) = c_{11}' w_1(n) + \bar{\mathbf{c}}_1'^{T}\bar{\mathbf{w}}(n) = \mu_1'. \tag{3.36}$$

where

$$\mathbf{c}_1'^{T} = [c_{11}', \bar{\mathbf{c}}_1'^{T}]$$

and

$$\mathbf{w}^{T}(n) = [w_1(n), \bar{\mathbf{w}}^{T}(n)].$$

The residual vector can now be rewritten as

$$\mathbf{e}(n) = \mathbf{x}_1(n)w_1(n) + \bar{\mathbf{X}}(n)\bar{\mathbf{w}}(n) \tag{3.37}$$

where

$$\mathbf{X}(n) = [\mathbf{x}_1(n),\ \bar{\mathbf{X}}(n)].$$

Since the constraint matrix $\mathbf{C}$ is a matrix of the Vandermonde type [2], its diagonal elements c'_{pp} are non–zero. Now, by substituting $w_1(n)$ of Eqn. (3.36) into (3.37), we obtain

$$\mathbf{e}(n) = \mathbf{y}'(n) + \mathbf{X}'(n)\bar{\mathbf{w}}(n) \tag{3.38}$$

where

$$\mathbf{X}'(n) = \bar{\mathbf{X}}(n) - \frac{1}{c'_{11}}\mathbf{x}_1(n)\bar{\mathbf{c}}'^{T}_1 \tag{3.39a}$$

and

$$\mathbf{y}'(n) = \frac{\mu'_1}{c'_{11}}\mathbf{x}_1(n) \tag{3.39b}$$

Similarly, the introduction of the second constraint is as follows

$$\mathbf{c}'^{T}_2\mathbf{w}(n) = c'_{22}w_2(n) + \bar{\mathbf{c}}'^{T}_2\tilde{\bar{\mathbf{w}}}(n) = \mu'_2. \tag{3.40}$$

where

$$\mathbf{c}'^{T}_1 = [c'_{11},\ \bar{\mathbf{c}}'^{T}_1]$$

and

$$\bar{\mathbf{w}}^{T}(n) = [w_2(n),\ \tilde{\bar{\mathbf{w}}}^{T}(n)].$$

The residual vector then becomes

$$e(n) = \mathbf{x}_2'(n)w_2(n) + \mathbf{X}'(n)\tilde{\tilde{\mathbf{w}}}(n) \tag{3.41}$$

where

$$\mathbf{X}'(n) = [\mathbf{x}_2'(n), \tilde{\mathbf{X}}'(n)].$$

And, by substituting $w_2(n)$ of Eqn. (3.40) into (3.41), the resultant is

$$e(n) = \mathbf{y}''(n) + \mathbf{X}''(n)\tilde{\tilde{\mathbf{w}}}(n) \tag{3.42}$$

with

$$\mathbf{X}''(n) = \tilde{\mathbf{X}}'(n) - \frac{1}{c_{22}'}\,\mathbf{x}_2'(n)\tilde{\mathbf{c}}_1'^{\,T} \tag{3.43a}$$

and

$$\mathbf{y}''(n) = \mathbf{y}'(n) + \frac{\mu_2'}{c_{22}'}\,\mathbf{x}_2'(n) \tag{3.43b}$$

The above procedure can be extended to incorporate all P constraints in Eqn. (3.32). Each constraint transformation is implemented at the expense of one degree of freedom, i.e., the elimination of control of the weight element, $w_p(n)$ (p=1,2,..., P). Thus after the application of P constraint transformations, the transformed data is reduced to M–P columns, i.e, the adaptive weight vector has been reduced in size from M to M–P. In the next section, it will be shown that the above procedure can be implemented by using the triarray least–squares processor in Fig. 3.1. The top P rows of the processor have two distinct functional modes. First, the adapted mode, which performs the QR decomposition, transforms the constraint matrix C to the trapezoidal matrix C'. Secondly, the unadapted (frozen) mode performs pre–processing operations of the input data.

3.3.2. *Systolic Implementation*

The architecture for a systolic array that has been configured for solving the adaptive least–squares beamforming problem with P linear constraints, is given in Fig. 3.3. The top P rows of the processor are used to transform the initial constraint matrix **C** into the trapezoidal form **C'**. The elements of matrix **C'** are stored as the elements of the **R** matrix within the corresponding PEs in the first P rows. Note that no connections between boundary cells of the top P rows are required, since the residual error is always zero at these stages (Eqn. (3.35)). If the square–root–free algorithm is used the elements of matrix $\mathbf{K}(n)$ (see Eqn. (3.24)), which are stored in the internal cells, will have the normalized value of c'_{pq}/c'_{pp}. Furthermore, if the adaptation within these rows is subsequently frozen by setting $\alpha_p = 0$ (p=1,2,...,P), it follows that the cell functions specified in Fig. 3.2b perform the same function as that of the pre–processing operations of Eqns. (3.39) and (3.43). In fact, by denoting

$$t_{pq} = c'_{pq}/c'_{pp} \quad \text{and} \quad t_p^y = -\mu/c'_{pp} \tag{3.44}$$

where

$$p,\ q = 1,\ 2,\ \ldots,\ P$$

then by inspection of Fig. 3.2b, which gives the processing at the nodes of the systolic array, Eqns. (3.39) and (3.43) follow directly [54]. Therefore, the role of the top P rows of the triarray processor is that of a pre–processing network. The remainder of the minimization procedure is carried out in the remaining (M–P) rows, i.e., they act as a triarray least–squares processor.

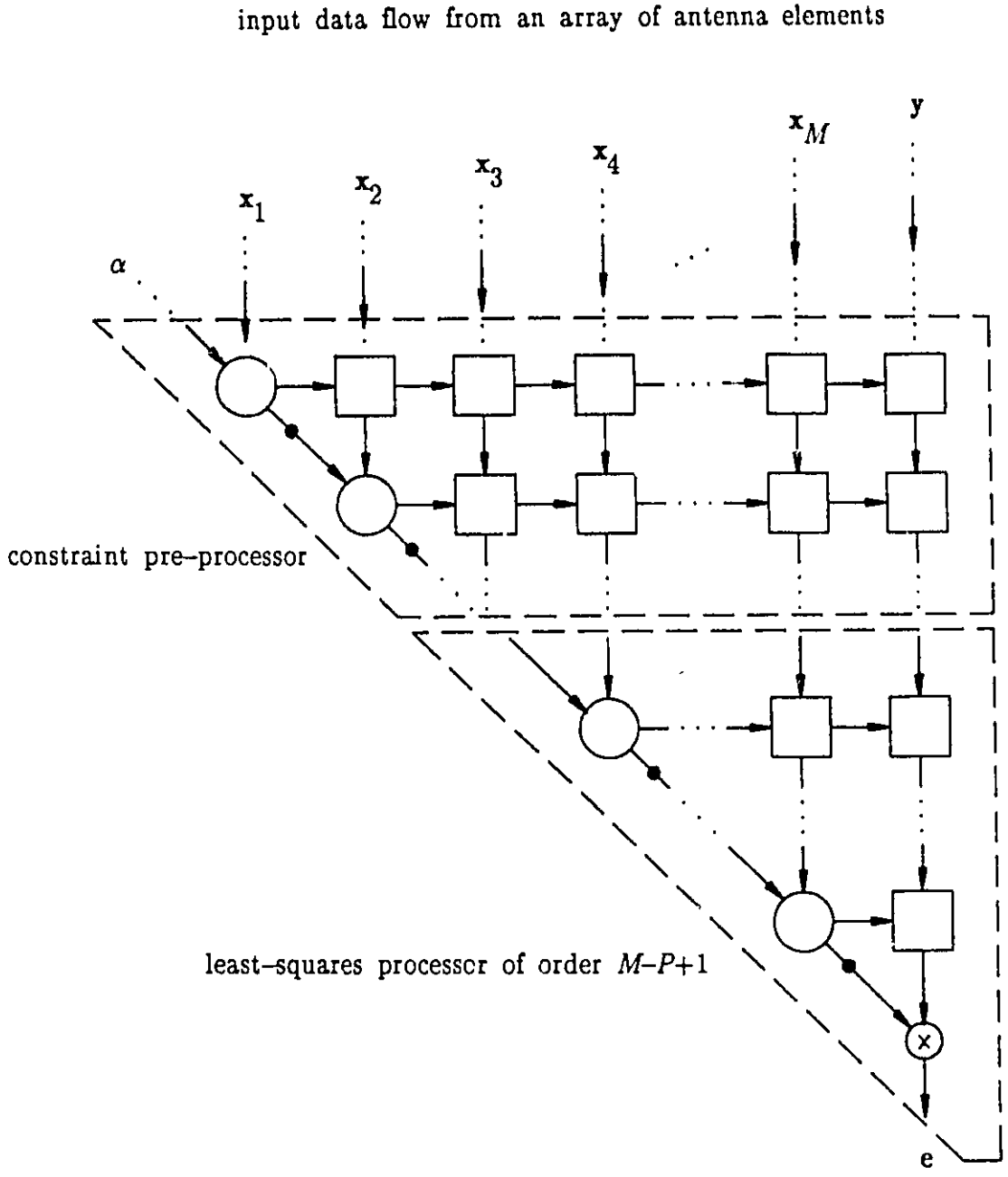


Fig. 3.3 LS triarray processor with P linear constraints

3.3.3. *Special Cases*

A common problem in beamforming is the *minimum variance distortionless response* (MVDR) beamforming or adaptive beamforming with a single constraint. The problem is summarized as follows. Minimize the residual vector $\mathbf{e}(n)$ subject to the constraint

$$\mathbf{c}^T(\theta_d)\mathbf{w}(n) = 1 \tag{3.45}$$

where $\mathbf{c}(\theta_d)$ is the steering vector corresponding to the desired signal direction, θ_d.

Several algorithms and implementations for realizing the MVDR beamforming have been studied and proposed [46]. However, these methods are rather complex. From the results given in the last two subsections (sections 3.3.1. and 3.3.2), it is clear that the problem can be easily handled by the triarray processor described in Fig. 3.3, in which the first row of the processor is used for pre–processing the data. Another approach, which is simple and specifically designed for the MDVR beamforming, is described in the following.

Equation (3.45) may be rewritten as

$$w_1(n) = 1 - \bar{\mathbf{c}}^T(\theta_d)\bar{\mathbf{w}}(n) \tag{3.46}$$

where $\bar{\mathbf{c}}(\theta_d)$ and $\bar{\mathbf{w}}(n)$ denote the last M–1 elements of the vectors $\mathbf{c}(\theta_d)$ and $\mathbf{w}(n)$, respectively, i.e,

$$\mathbf{c}^T(\theta_d) = [1,\ \bar{\mathbf{c}}^T(\theta_d)] \tag{3.47a}$$

and

$$\mathbf{w}^T(n) = [w_1(n), \bar{\mathbf{w}}^T(n)]. \tag{3.47b}$$

Substitution of the quantity $w_1(n)$ in Eqn. (3.46) into Eqn. (3.30) yields

$$\mathbf{e}(n) = \mathbf{x}_1(n) + \{\bar{\mathbf{X}}(n) - \mathbf{x}_1(n)\bar{\mathbf{c}}^T(\theta_d)\}\bar{\mathbf{w}}(n) \tag{3.48a}$$

$$= \mathbf{x}_1(n) + \mathbf{X}'(n)\bar{\mathbf{w}}(n) \tag{3.48b}$$

where

$$\mathbf{X}(n) = [\mathbf{x}_1(n), \bar{\mathbf{X}}(n)],$$

has been used. It follows from Eqn. (3.48b) that the residual vector of the adaptive beamformer may be computed directly using the triarray processor of Fig. 3.1. The term $\mathbf{x}_1(n)$ corresponds to the primary vector and the transformed matrix $\mathbf{X}'(n)$ corresponds to the auxiliary data matrix. Since the constraint has been absorbed explicitly by removing the coefficient $w_1(n)$, and thereby reducing the degrees of freedom to M–1, the residual power of the combined signal, $\|\mathbf{e}(n)\|^2$, may now be minimized by using the unconstrained M–1 element weight vector $\bar{\mathbf{w}}(n)$. A complete schematic of the MVDR beamformer is given in Fig. 3.4. Note that the constraint processor precedes and is separate from the triarray processor. Thus no control is necessary to freeze the processing in the first row of the adaptive processor as was the case in Fig. 3.3. However, more circuitry is involved in this implementation due to the constraint pre–processor to be attached.

It should be pointed out that the clamped element configuration described in Section 3.2.3 is also a special form of linearly constrained process, in which the constraint vector is of the form

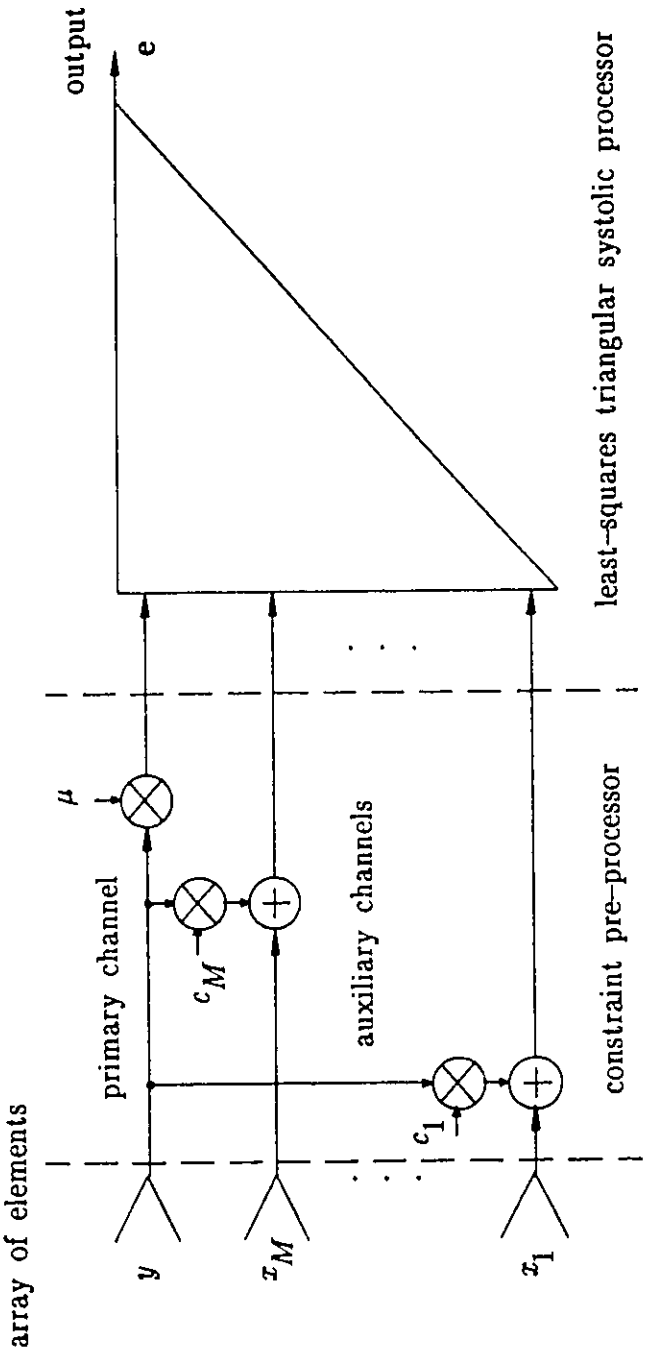


Fig. 3.4 Adaptive beamforming processor with a linear constraint

$$\mathbf{c}^T = [1, 0, 0, \ldots, 0]. \tag{3.49}$$

In fact, if the vector **c** is substituted for $\mathbf{c}(\theta_d)$ in Eqn. (3.47a) it can be shown that the Eqn. (3.48b) becomes identical to Eqn. (3.29).

3.4. Adaptive Beamforming with Planar Arrays

3.4.1. *QRD–LS Algorithm*

Based on the earlier discussion of 2–D adaptive beamforming in Chapter 2, the output signal from the beamformer can be expressed as

$$\mathbf{e}_l(n) = \mathbf{y}_l(n) - \mathbf{X}_l(n)\mathbf{w}_l(n) \tag{3.50a}$$

$$\mathbf{e}_m(n) = \mathbf{y}_m(n) - \mathbf{X}_m(n)\mathbf{w}_m(n) \tag{3.50b}$$

where subscripts l, m are, respectively, indices along the row and column directions. Without any loss of generality, we may assume that $L = M$, thereby allowing Eqns. (3.50a,b) to be written in the combined form

$$\mathbf{e}_{l,m}(n) = \mathbf{y}_{l,m}(n) - \mathbf{X}_{l,m}(n)\mathbf{w}_{l,m}(n). \tag{3.51}$$

By applying unitary transformations $\mathbf{Q}_{l,m}(n)$ to Eqn. (3.51), we obtain

$$\mathbf{Q}_{l,m}(n)\mathbf{e}_{l,m}(n) = \mathbf{Q}_{l,m}(n)\mathbf{y}_{l,m}(n) - \mathbf{Q}_{l,m}(n)\mathbf{X}_{l,m}(n)\mathbf{w}_{l,m}(n)$$

$$= \begin{bmatrix} \mathbf{u}_{l,m}(n) \\ \mathbf{v}_{l,m}(n) \end{bmatrix} - \begin{bmatrix} \mathbf{R}_{l,m}(n) \\ \mathbf{0} \end{bmatrix}. \tag{3.52}$$

Referring to Figure 3.5 it can be seen that the receiver signal at element m of row l is equal to that at element l of column m, i.e.,

$$x_{\substack{l \\ m}}(n) = x_{\substack{m \\ l}}(n). \tag{3.53}$$

An evaluation of Eqn. (3.52) along with the condition given in Eqn. (3.53), leads to the conclusion that 2–D adaptive beamforming can be implemented by applying a sequence of unitary transformations to the measured data matrix. If the data flow is managed in a suitable manner, the processing can be carried out using a distributed configuration of PEs. As a result, the beamformer output can be derived by concurrent processing of the rows and columns of the antenna array.

The triangularization process using the Givens rotation method can be applied to the 2–D problem in the same manner as for the 1–D case. As before, the elementary transformations are of the form

$$\begin{bmatrix} C_{l,m} & S^{*}_{l,m} \\ -S_{l,m} & C_{l,m} \end{bmatrix} \begin{bmatrix} 0 \ \ldots 0, & r_{l_i,m_j} \ldots, & r_{l_p,m_q} \ldots \\ 0 \ \ldots 0, & x_{l_i,m_j} \ldots, & x_{l_p,m_q} \ldots \end{bmatrix}$$

$$= \begin{bmatrix} 0 \ \ldots 0, & r'_{l_i,m_j} \ldots, & r'_{l_p,m_q} \ldots \\ 0 \ \ldots 0, & 0 \ \ldots, & x'_{l_p,m_q} \ldots \end{bmatrix} \tag{3.54}$$

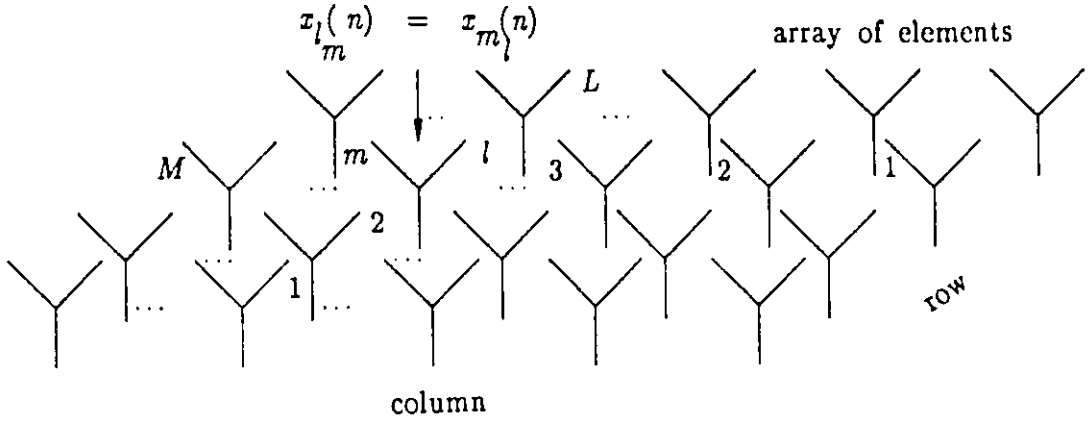


Fig. 3.5 Configuration of a 2–D array

where the rotation coefficients, $C_{l,m}$ and $S_{l,m}$, satisfy

$$-S_{l,m}r_{l_i,m_j} + C_{l,m}x_{l_i,m_j} = 0 \tag{3.55a}$$

$$S^*_{l,m}S_{l,m} + C^*_{l,m}C_{l,m} = 1 \tag{3.55b}$$

$$C^*_{l,m} = C_{l,m}. \tag{3.55c}$$

Therefore the rotation coefficients are uniquely specified by

$$C_{l,m} = \frac{|r_{l_i,m_j}|}{(|x_{l_i,m_j}|^2 + |r_{l_i,m_j}|^2)^{1/2}}, \tag{3.56a}$$

and

$$S_{l,m} = \frac{x_{l_i,m_j}}{r_{l_i,m_j}} C_{l,m}. \tag{3.56b}$$

3.4.2. *Systolic Array Implementation*

Implementation

In the last section, it was shown that triangularization of the rows and columns of the input matrix can be computed independently. In other words, 2–D adaptive beamforming can be performed on the rows and columns separately. The overall performance of the adaptive system is based on the combined results. Decoupled row–column adaptive beamforming could be carried out with a single

triangular–form systolic structure, similar to the one used for linear antennas (Fig. 3.1). This, however, is a slow and time consuming approach, since a large memory is required to store unprocessed data for later use. For instance, with an array of dimensions L by M, the storage requirement for unprocessed data can reach as high as $L \times M \times N$. It follows, as well, that an equivalent number of time delay cycles will apply to the processing required to obtain the beamformer output. This may become impractical, especially when the size of the antenna array is large, as is the case of some radar applications.

The second approach to be considered in this thesis for implementing 2–D adaptive beamforming is given in Fig. 3.6. This arrangement consists of M layers. It is assumed that $M \geq L$. Each layer is a triarray of PEs that is configured to perform adaptive beamforming on either a row or a column of the 2–D array. The advantage of this configuration is that the processing on the rows and columns of the antenna array can be carried out concurrently. However, the amount of unprocessed data that must be stored for latter use is again large: it is equal to $L \times M \times N$. And the total delay in processing the data is about $M \times N$.

The third configuration to be considered is shown in Fig. 3.7 [25]. It will be shown that the processing is extremely robust. The systolic array arrangement is a 3–D structure. It is composed of four types of cells: (1) boundary, (2) internal, (3) dual, and (4) a final processing cell. The computations performed by these PEs are given in Fig. 3.7, as well. The functions of the boundary and internal cells are as discussed in Section 3.2.3. A dual cell is one that has two functions, i.e. it can act either as a boundary cell or as an internal cell. The specific function that it executes is selected and set by an internal clock. As can be seen in Fig. 3.7, all of the PEs are highly pipelined and interconnections exist between

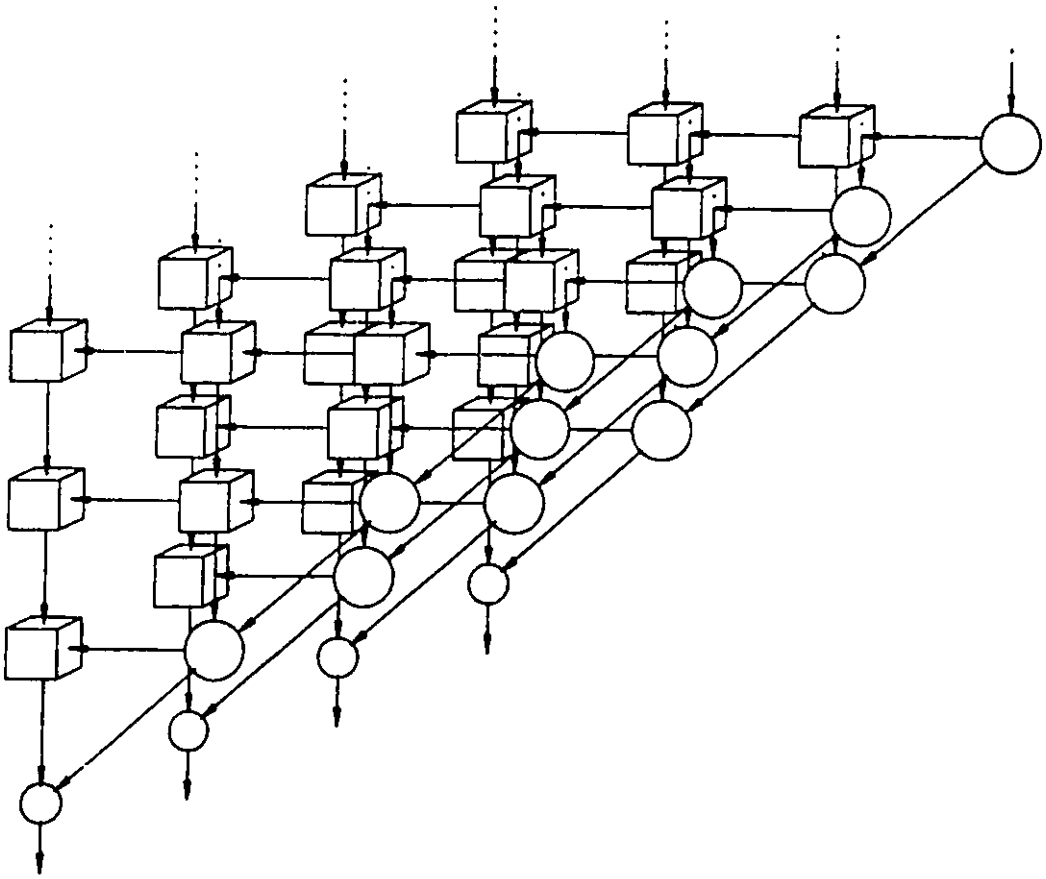


Fig. 3.6 2–D systolic layers for 2–D adaptive beamforming

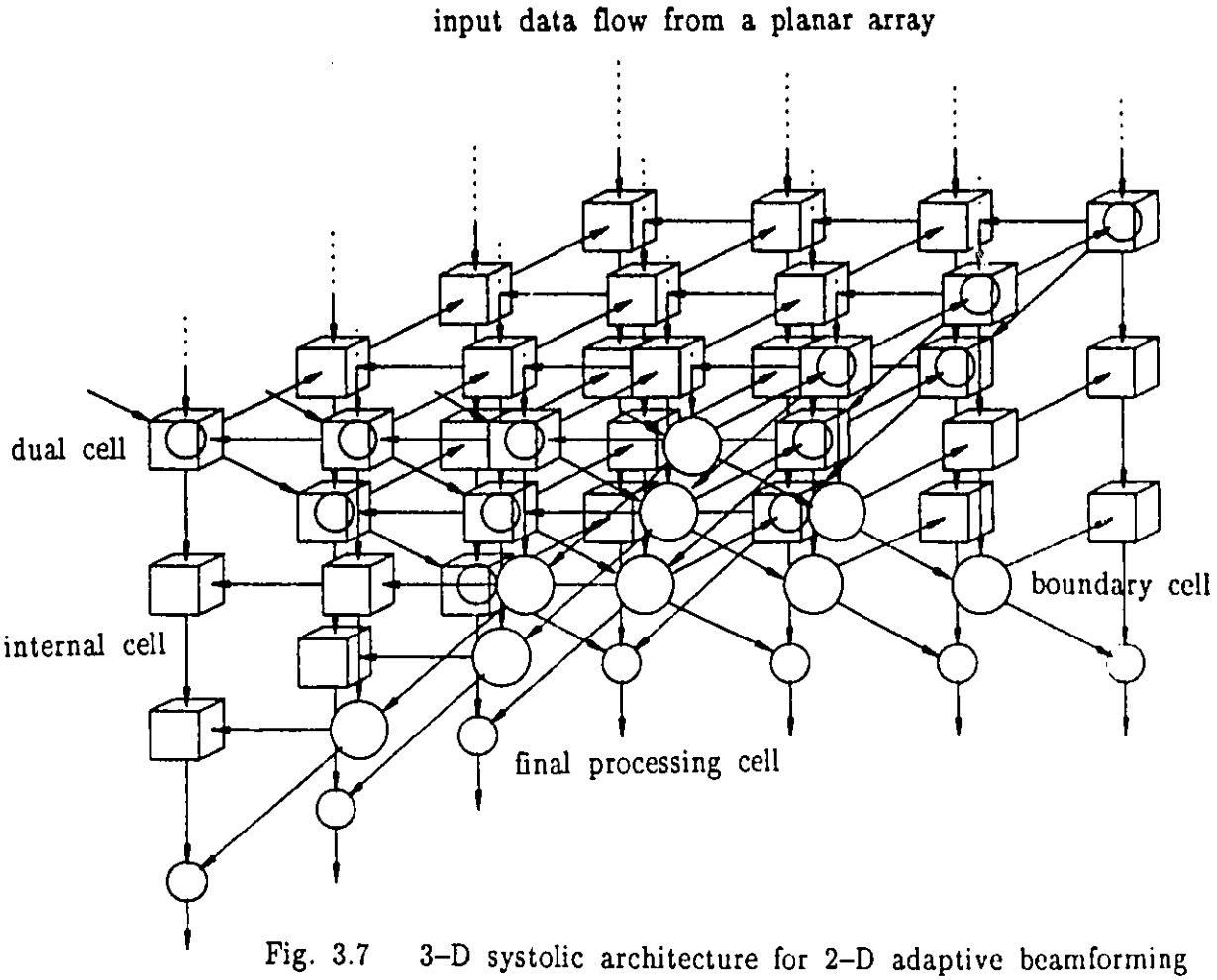
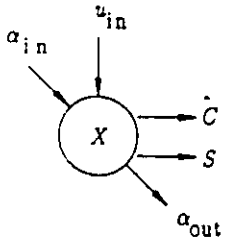


Fig. 3.7 3–D systolic architecture for 2–D adaptive beamforming

Boundary Cell



$\alpha_{in} = 1$

$\dot{C} \leftarrow u_{in}$

if $\dot{C} = 0$, $S \leftarrow 0$, $\alpha_{out} \leftarrow \alpha_{in}$

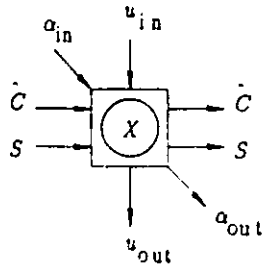
else

$X' \leftarrow X + \alpha_{in}|\dot{C}|^2$

$S \leftarrow \alpha_{in} u_{in} / X'$

$\alpha_{out} \leftarrow X \alpha_{in} / X'$

Dual Cell



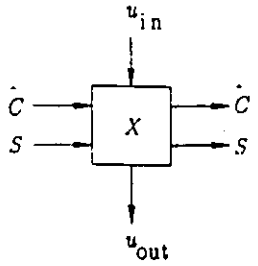
If (flag=1) then

dual cell ← internal cell

else

dual cell ← boundary cell

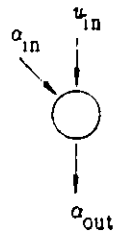
Internal Cell



$u_{out} \leftarrow u_{in} - \dot{C}X$

$X' \leftarrow X + S^* u_{out}$

Final Processing Cell



$\alpha_{out} \leftarrow u_{in} \alpha_{in}$

nearest neighbours throughout the 3–D systolic structure. The main feature of this highly pipelined structure is its concurrent adaptation of both the rows and columns of the 2–D array. The input data flow, as in the 1–D case, is arranged in time skew format, as a consequence of the internal operation of the systolic array, and is controlled by a global and synchronous clock. It is also to be noted that the processing of input data occurs at the instant that they enter the systolic array. At most, only $2M$ time delay–cycles are executed prior to results appearing at the output of the beamformer: $1M$ delay cycles are due to the time skew format of the 2–D input data as compared with the 1–D case ($2M$ delay cycles are associated with the triarray processor), and another $1M$ delay cycles are due to the use of a directional clock signal, which will be discussed in the next section. In conclusion, a total of $4M$ snap shots is required to initialize the 3–D systolic processor for 2–D adaptive beamforming. If $L \neq M$ this value is $4max(L,M)$.

A comparison of the processing requirement for the three approaches described above as well as the conventional approach is given in Table 3.1. It should be recalled that the conventional approach to 2–D beamforming treats an L–by–M array as an 1–D array with LM elements. In Table 3.1 it is assumed that $L = M$. It is to be noted that of the four techniques considered the 3–D systolic array approach seems to be the best. Its time delay is the least of all the approaches described. From the point of view of the number of PEs required there is a tradeoff between the first approach (2–D systolic with weight decoupling) and the last approach (2–D systolic without weight decoupling).

Table 3.1.

A comparison of different systolic structures for 2–D adaptive beamforming.

	Method used	Unprocessed data storage	Time delay cycles	Number of PEs required ICs	BCs	DCs	FCs
(1)	2—D systolic (M elements)	$M\times M\times N$	$(M-1)\times M\times N$	$M\times(M-1)/2$	$M-1$	0	1
(2)	2–D systolic layers	$M\times M\times N$	$M\times N$	$M^2(M-1)/2$	$M(M-1)$	0	M
(3)	3-D systolic	0	$2M$	$\frac{M(M-1)(4M-5)}{6}$	$(M-1)^2$	$M(M-1)$	$2M-1$
(4)	2—D systolic (M^2 elements)	0	$2M^2$	$M^2\times(M^2-1)/2$	M^2-1	0	1

ICs – internal cells

BCs – boundary cells

DCs – dual cells

FCs – final processing cells

Data Flow Format and Control Signals

Besides the time–skewed format required for the input data, which is a consequence of the highly pinelined nature of the systolic array implementation, the 3–D systolic structure also requires the control of data flow to achieve adaptation in both directions, i.e. along the rows and columns of the 2–D array simultaneously. We have yet, as well, to carefully define the types of cells that are used in the computations – of particular interest here are the dual cells. Furthermore, the introduction of dual cells leads to the requirement for defining two other clock signals (control flags), which are: (1) the direction control clock; and (2) the dual–cell control clock. The functions performed by these clocks are as follows.

i. Direction Control

```
If (direction-flag=1) then
    direction-of-adaptation ⟵ row-wise
else
    direction-of-adaptation ⟵ column-wise
end.
```

ii. Dual–Cell Control

```
If (cell-flag = 1) then
    dual-cell ⟵ boundary-cell
else
    dual-cell ⟵ internal-cell
end.
```

Both a directional control and a global clock are used in Fig. 3.7 to ensure that the incoming data flow in the proper direction. The signals for these clocks are given in Figs. 3.8a and 3.8b. Note that the frequency of the direction clock is

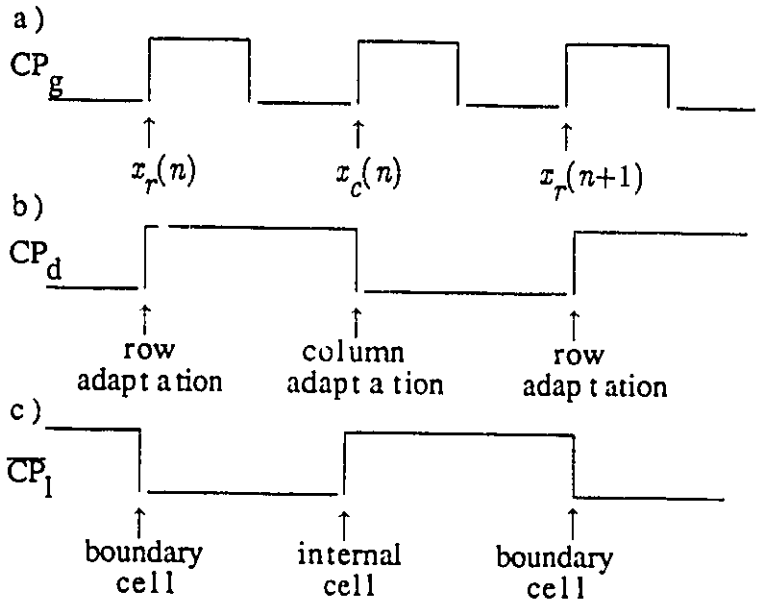


Fig. 3.8 Control clocks for the 3-D systolic array processor
a) global clock b) direction clock c) dual-cell clock

only one–half of that for the global clock. As a result, new row or column input data enter the processor every two clock cycles, which results in $1M$ extra time delay cycles at the beamforming output, as pointed out in the last section.

A dual cell control clock, on the other hand, is just a local clock, which operates as a switch between boundary and internal cells. It is only used for controlling dual cells. As an example, when used to control the dual cell in the far left corner of Fig. 3.7 the form of the clock signal is given in Fig. 3.8c. It is important to note that the dual–cell clock has the same frequency as the direction clock. The phase between these clocks is either 0^{o} or 180^{o}. Its use therefore will not affect the synchronization of the processors which form the 3–D systolic structure. In other words, the time delay at each processing cell is constant, depending only on the global and direction clock rates, whether or not the dual cell is used in the computation.

It should be noted that computations performed by 3–D processing cells in Fig. 3.7 are the same as those performed by 2–D cells, except that extra control clocks are added to the 3–D cells. In essence, the introduction of control clocks will require more memory for storing row data while the adaptation is performed along the columns and vice versa. The amount of data storage, however, varies within each cycle of the computations. Since the requirement for storage is restricted to the PEs, sufficient memory must be included in each PE to ensure that the requirement for memory never exceeds the resident memory in the PE.

To conclude, it has been found that a 3–D systolic array provides a significant improvement in memory storage and overall system time delay over that which can be achieved using 2–D configuration. 3–D adaptive processors are

potentially capable of performing real time processing. The key to the successful operation of the 3–D systolic array has been the introduction of two additional control clocks, above and beyond the global system clock, namely the direction control clock and the dual–cell control clock. The direction control clock is used to control adaptation to the rows and columns. The dual cell clock, which is a local clock, is used to switch processing data between boundary and internal cells.

At first sight there appears to be a number of drawbacks to the 3–D arrangement. In the main, they revolve around the requirement for: (1) complex I/O interfaces, (2) large numbers of PEs, and (3) an extensive communications network to link the PEs. The development of a 3–D systolic array therefore is strongly dependent on the availability of processors with high I/O and throughput rates. The development of these, in turn, depends heavily on the progress that is made in VLSI design methods and technologies. The rapid development that is and continues to occur in VLSI and VHSIC suggests that 3–D systolic arrays are a distinct possibility in the not too distant future.

3.5. Concluding Remarks

Adaptive beamforming algorithms, which are formulated and implemented using systolic arrays, comprise what we call the modern adaptive beamforming techniques. These methods have been analyzed, investigated, and undergone further development in this Chapter. Adaptive beamforming algorithms for both linear arrays and planar arrays have been presented. Linearly constrained adaptive beamforming was also studied and presented. It has been shown that the recursive QRD–LS algorithm, which is computed based on the Givens rotation and which

can be implemented using systolic arrays, is the cornerstone in the development of the modern beamforming technology. VLSI technology is well suited for the development of systolic arrays: thus the successful development of adaptive beamformers with a real–time processing capability in the near future looks fairly promising.

CHAPTER 4

PERFORMANCE EVALUATION

4.1. Introduction

In this chapter, results giving the performance of adaptive beamforming algorithms will be presented. Both 1–D and 2–D adaptive beamforming algorithms will be evaluated. The emphasis however will lie on the modern techniques. In particular, adaptive beamforming techniques which is based on the QRD–LS algorithm and therefore amenable to having the processing carried out using systolic arrays will be of primary interest. In addition, comparative studies between the modern and classical approaches will be given whenever applicable.

In the 1–D adaptive beamforming problem, results will be derived for both unconstrained and linearly constrained adaptive beamforming techniques. 2–D adaptive beamforming is typified by the 2–D Howells–Applebaum algorithm in which 2–D eigenbeams are used to demonstrate the nulling capability of an adaptive beamformer. And finally, the performance of the 2–D adaptive beamforming algorithm based on the QR decomposition and amenable to processing by 3–D systolic arrays will be presented.

4.2. 1–D Adaptive Beamforming

4.2.1. *Unconstrained Method*

As discussed previously, adaptive beamformers in which the primary signal is constituted from either a main (high gain) antenna or a clamped element of the antenna array are classified as unconstrained beamformers. In these systems, the desired signal is assumed to be either totally absent during the adaptation process or to be very weak in comparison to interference signals (20–30 dB below).

Figure 4.1 shows the configuration of an M–element (M=10) antenna array utilized in the simulation results presented in the following sections. All ten elements are uniformly weighted and linearly combined to form the primary channel, while individual elements are also used to form the auxiliary channels. The signal model for the mth channel of the antenna array was described in Eqn. (1.9). It is repeated here for convenience

$$x_m(n) = A_d e^{\jmath\{(m-1)\varphi(\theta_d)+\psi_d\}} + \sum_{k=1}^{K} A_k e^{\jmath\{(m-1)\varphi(\theta_k)+\psi_k\}} + \nu_m(n) \qquad (4.1)$$

where $\varphi(\theta_i)$ is the phase difference between adjacent elements

$$\varphi(\theta_i) = \frac{2\pi d}{\lambda} \sin(\theta_i)$$

and

d – the inter–element spacing,

λ – antenna wavelength,

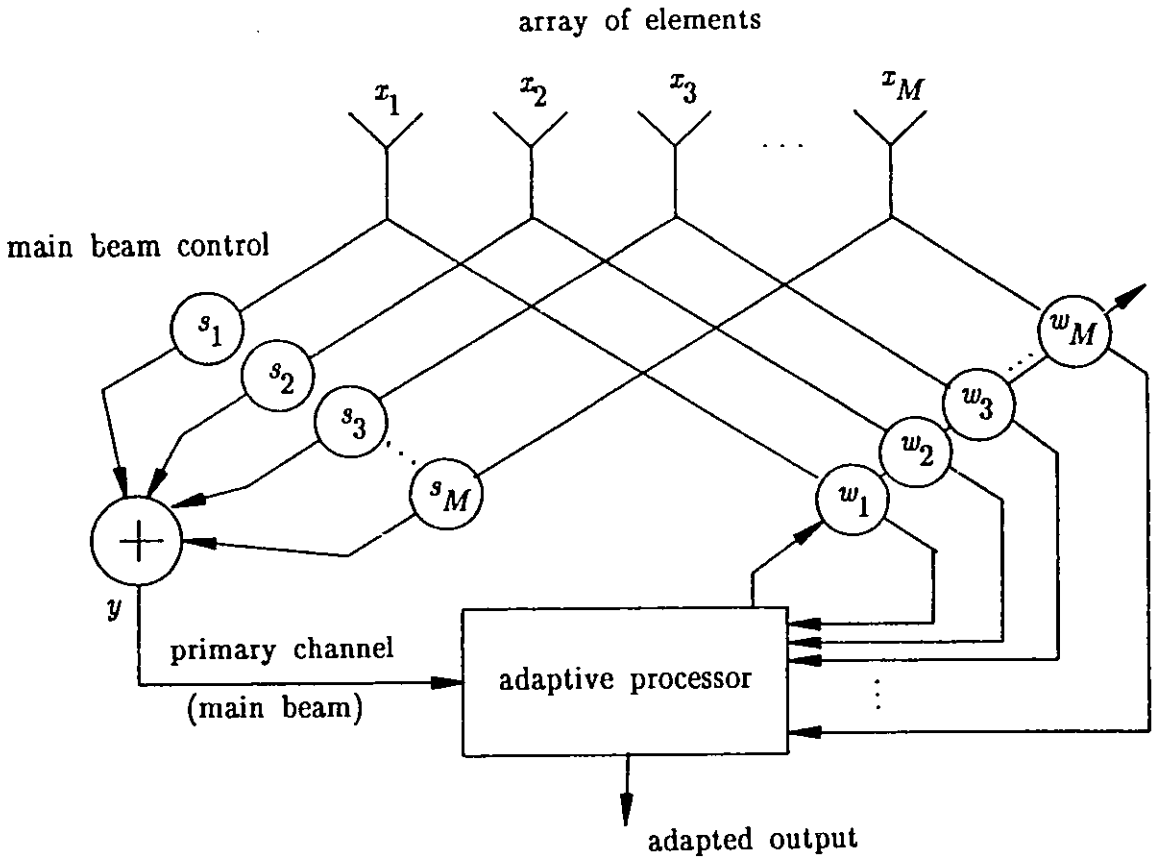


Fig. 4.1 Configuration of the antenna array used in simulation studies

θ_d – direction of arrival of the desired signal,

θ_k – direction of arrival of the kth interfering source,

A_d – amplitude of the desired signal,

A_k – amplitude of the kth interfering source,

$\nu_m(n)$– receiver noise assumed to be Gaussian with zero mean, and variance σ^2, and

ψ_d, ψ_k– uniformly distributed random variables with probability density

$$p(\psi_i) = \begin{cases} 1/2\pi : 0 \leq \psi_i \leq 2\pi \\ 0 \quad : \text{elsewhere} \end{cases}$$

The processing for adaptive beamforming can be carried out by using the triarray systolic processor given in Fig. 3.1. Table 4.1 depicts four different scenarios for the simulated data. Unless it is otherwise specified, the inter–element spacing in terms of wavelength, i.e. (d/λ) ratio, is chosen to be equal to 1/2. Some results of systolic array beamforming are shown in Figs. 4.2a to 4.5a.

As can be seen, deep nulls occur in the directions of the interferers. The depth of each null is dependent upon the corresponding jammer power in such a manner that the higher the interference–to–noise ratio (INR) is, the deeper is the null obtained. Performance characteristics of different adaptive beamforming algorithms are further illustrated in Fig. 4.6 and 4.7. It can be seen that the rate of convergence of the LMS algorithm is quite poor and particularly sluggish when the INR is low as compared with that of the SMI algorithm or the QRD–LS algorithm. Furthermore, it can be seen that the residual curve of the QRD–LS algorithm is quite independent of the environment conditions; it converges in a few

Table 4.1. Cases of simulation studies

case #	number of jammers	angle of arrival (θ_i)	interference to noise ratio (INR)
1	2	7^o, 20^o	35 dB, 40 dB
2	3	-15^o, 7^o, 20^o	15, 20, 10 dB
3	4	-18^o, -3^o, 15^o, 20^o	20, 30, 30, 35 dB
4	4	-15^o, 3^o, 7^o, 20^o	20, 30, 30, 35 dB

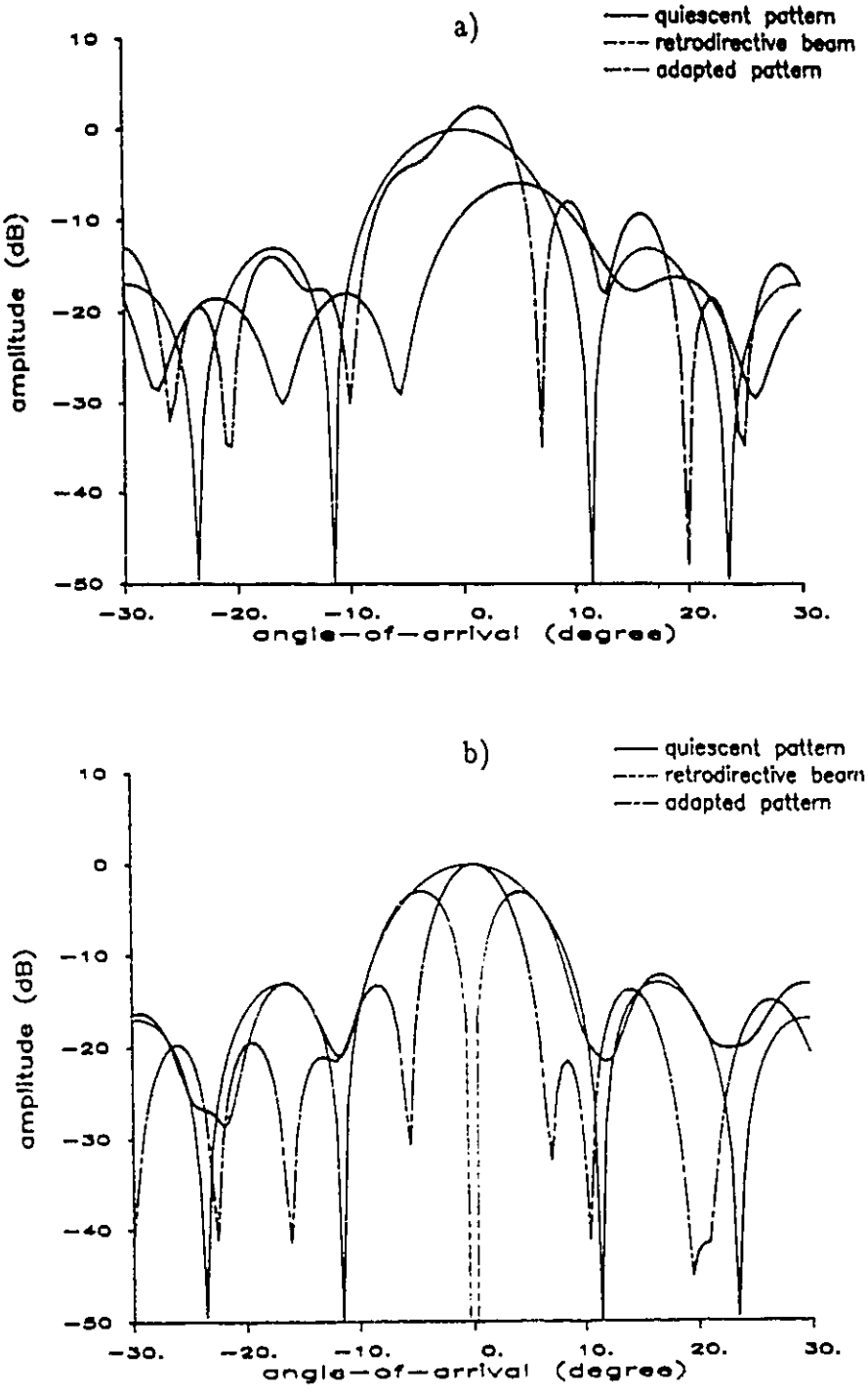


Fig. 4.2 Adapted patterns of a 10–element array, jammers at 7^{o} and 20^{o}
a) unconstrained method b) constrained method
after 50 iterations

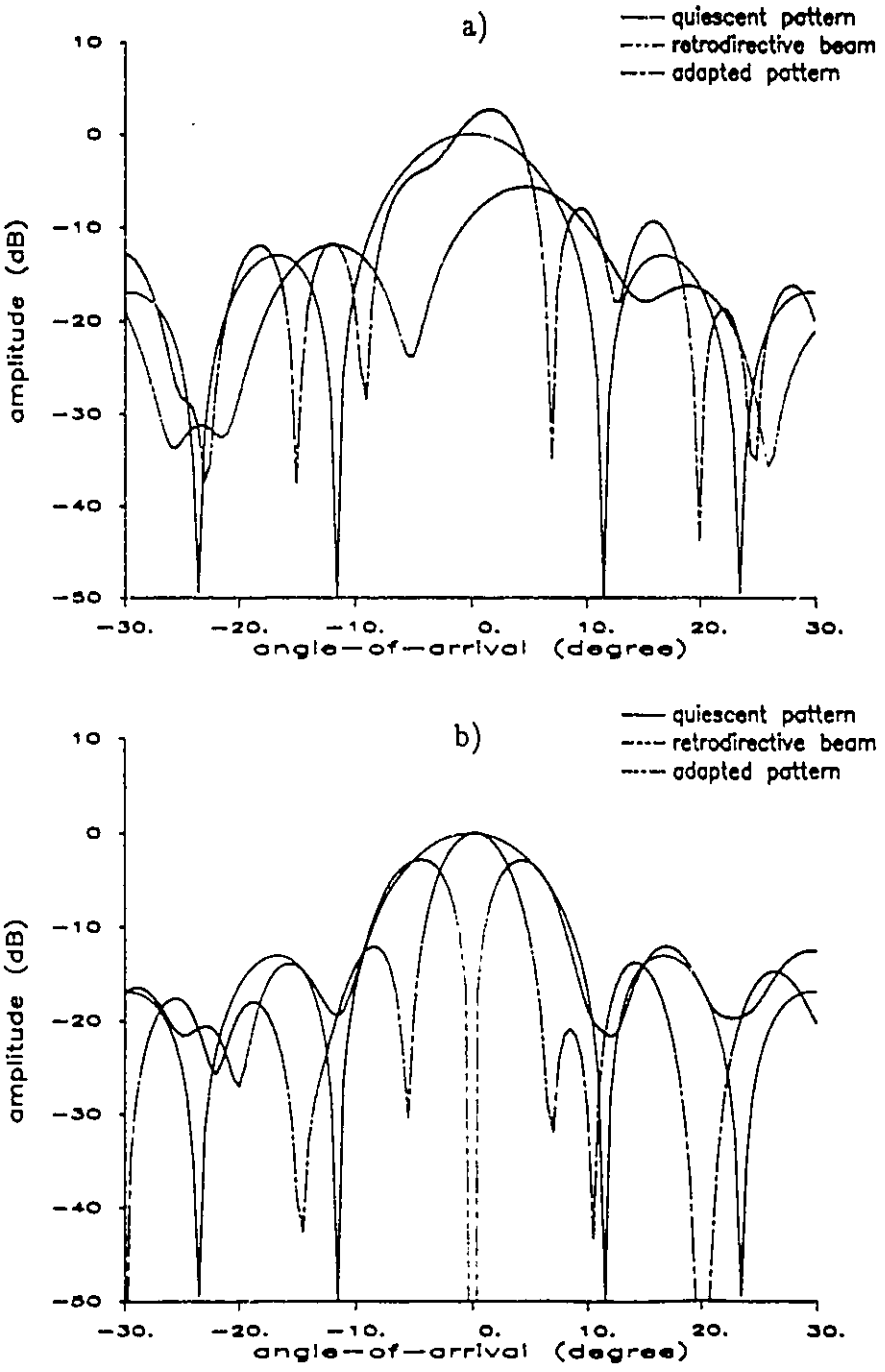


Fig. 4.3 Adapted patterns of a 10–element array, jammers at -15^{0}, 7^{0} and 20^{0} from the boresight
a) unconstrained method b) constrained method

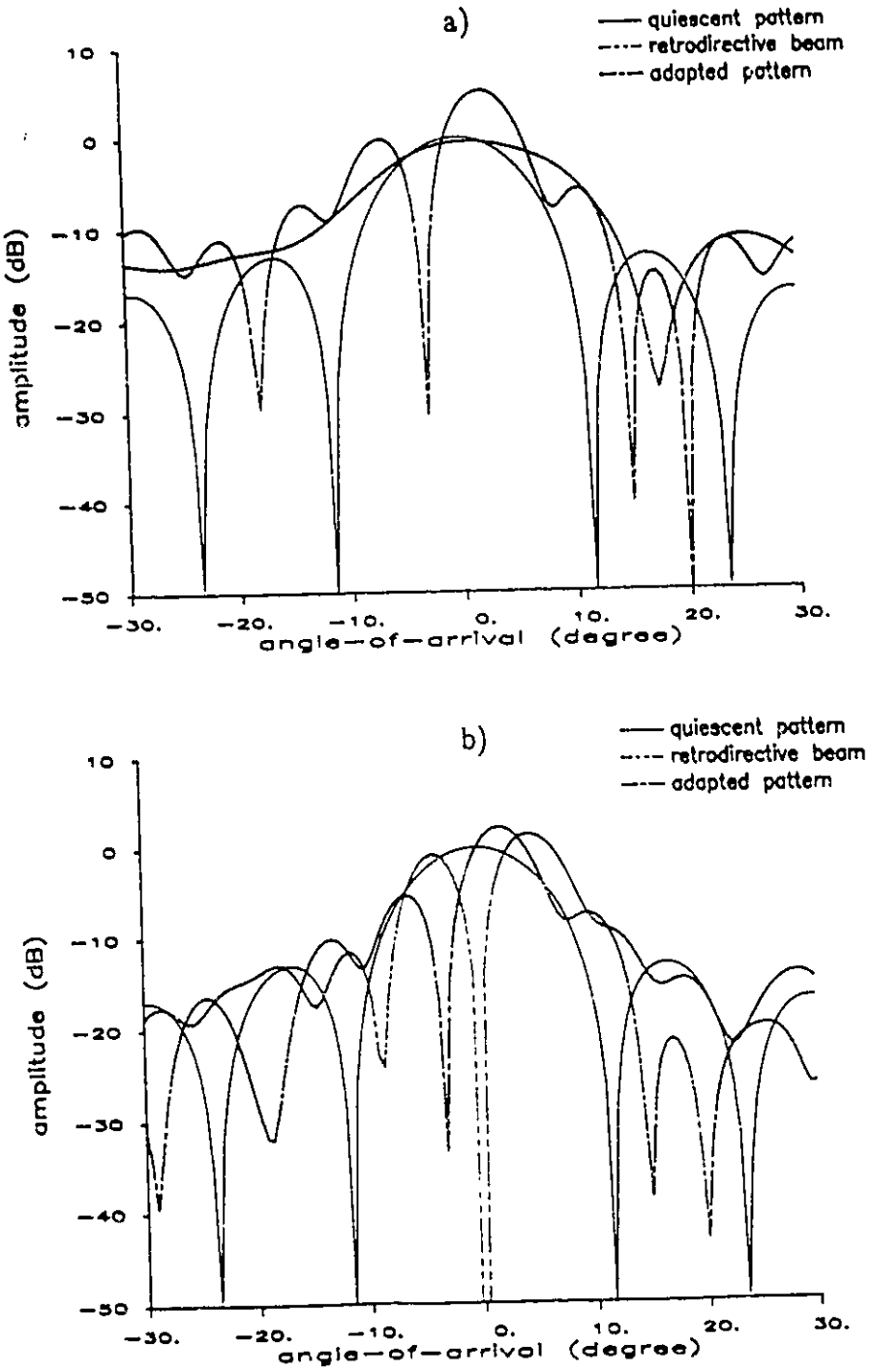


Fig. 4.4 Adapted patterns of a 10-element array, jammers at -18^0, -3^0, 15^0 and 20^0 from the boresight
a) unconstrained method b) constrained method

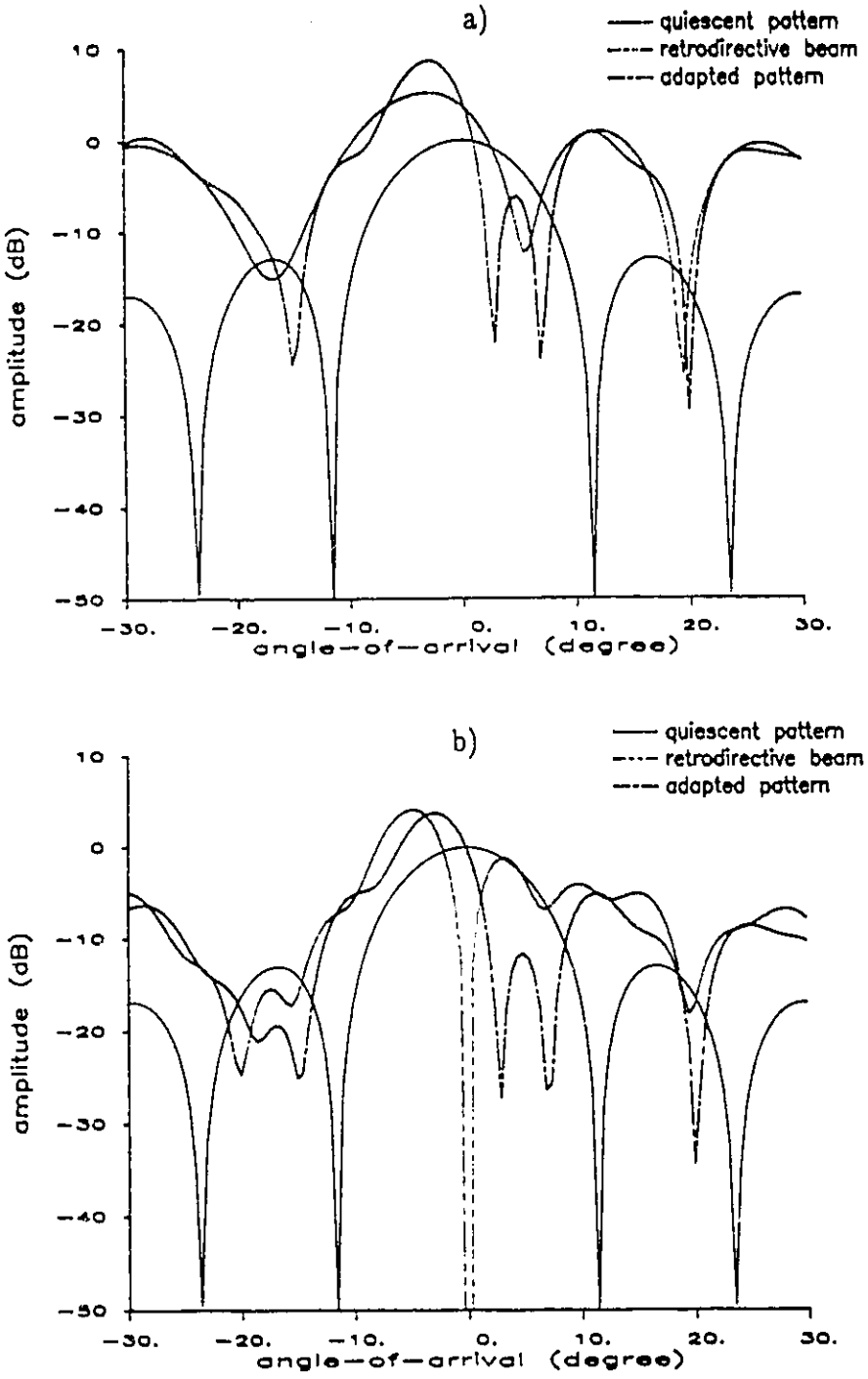


Fig. 4.5 Adapted patterns of a 10-element array, jammers at -15^{0}, 3^{0}, 7^{0} and 20^{0} from the boresight
a) unconstrained method b) constrained method

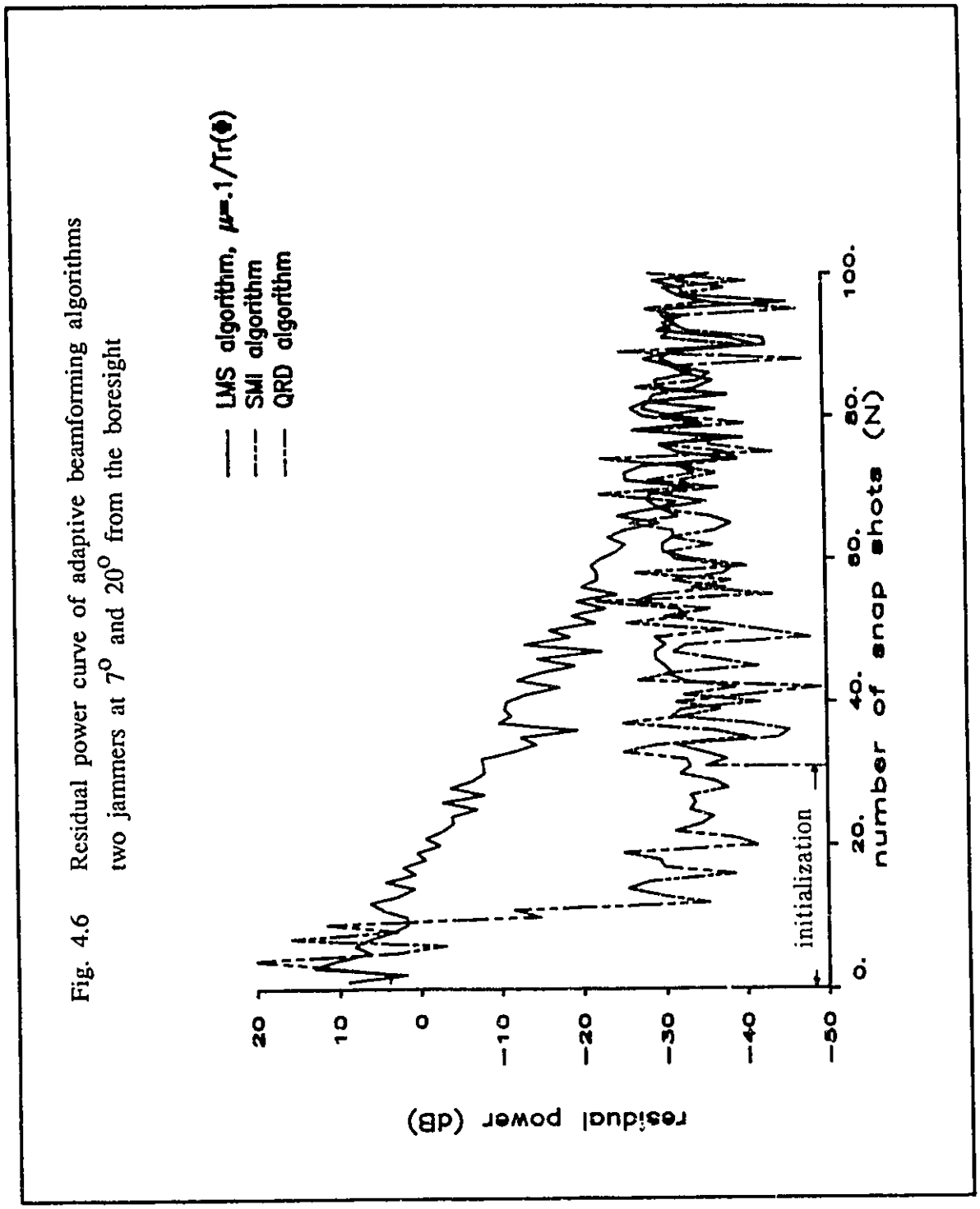


Fig. 4.6 Residual power curve of adaptive beamforming algorithms two jammers at 7^{o} and 20^{o} from the boresight

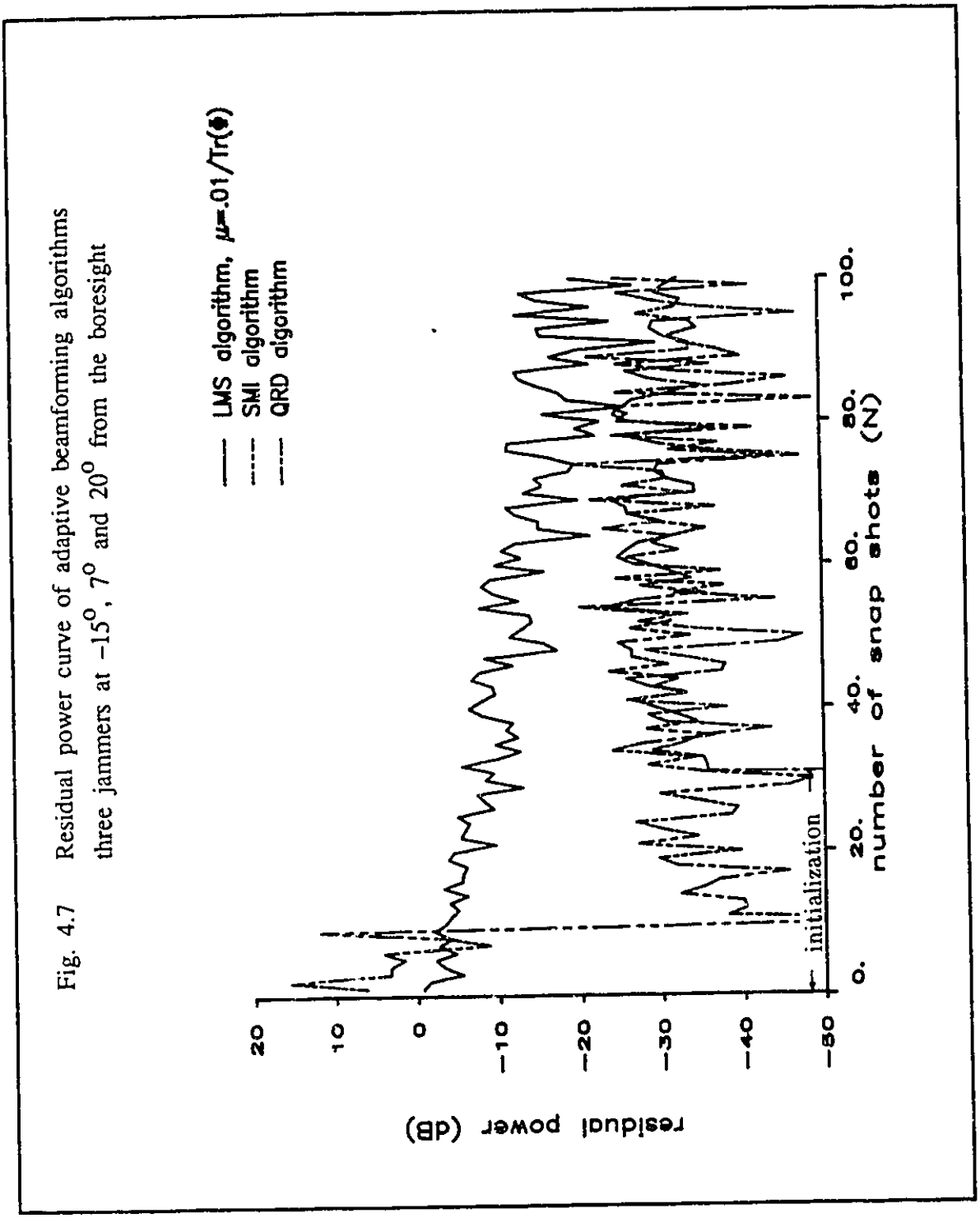


Fig. 4.7 Residual power curve of adaptive beamforming algorithms three jammers at -15^{o}, 7^{o} and 20^{o} from the boresight

iterations after an initialization ($2M$ snapshots). The curves giving the residual power ($\|e(n)\|^2$) in Figs. 4.6 and 4.7 also indicate that, under severe conditions (large number of interferers with low INR and/or constant change of interference scenarios), the QRD–LS algorithm implemented with systolic array is highly stable, whereas the LMS and the SMI algorithms are quite sensitive. In such cases, the signal environment is often referred to as ill–conditioned ($\lambda_{max}/\lambda_{min} >> 1$). It should be noted that the performance of the SMI and the QRD–LS algorithms are very similar under well–conditioned scenarios (Fig. 4.6); however, under ill–conditioned environments (Fig. 4.7), the QRD–LS algorithm seems to be better behaved than the SMI algorithm. As the INRs are relatively high the SMI algorithm converges to within 3–dB of the steady state in less than $2M$ iterations, i.e, its rate of convergence is much faster than expected by [18]. This phenomenon has also been demonstrated by [2].

Figures 4.4a and 4.5a (case 3 and 4) show the results for adaptive beamforming with jammers located in the main antenna beam. Under these circumstances, the antenna beamwidth has been reduced and the main antenna beam is highly distorted as compared to those in Figs. 4.2a and 4.3a. These effects may further degrade the final SINR and limit the performance of the antenna systems. Actually, the degradation of the main antenna beam due to jammers being located within the main beam is a well–understood problem by those workers that employ beamspace adaptive beamforming. The degradation is due to the separation between the jammers and the antenna boresight (direction of the desired signal) being less than a beamwidth ($.886\lambda/Md$) [21].

Simulation results are also provided for the case where the beamforming is carried out with a clamped element. Results are shown in Figs. 4.8a to 4.11a,

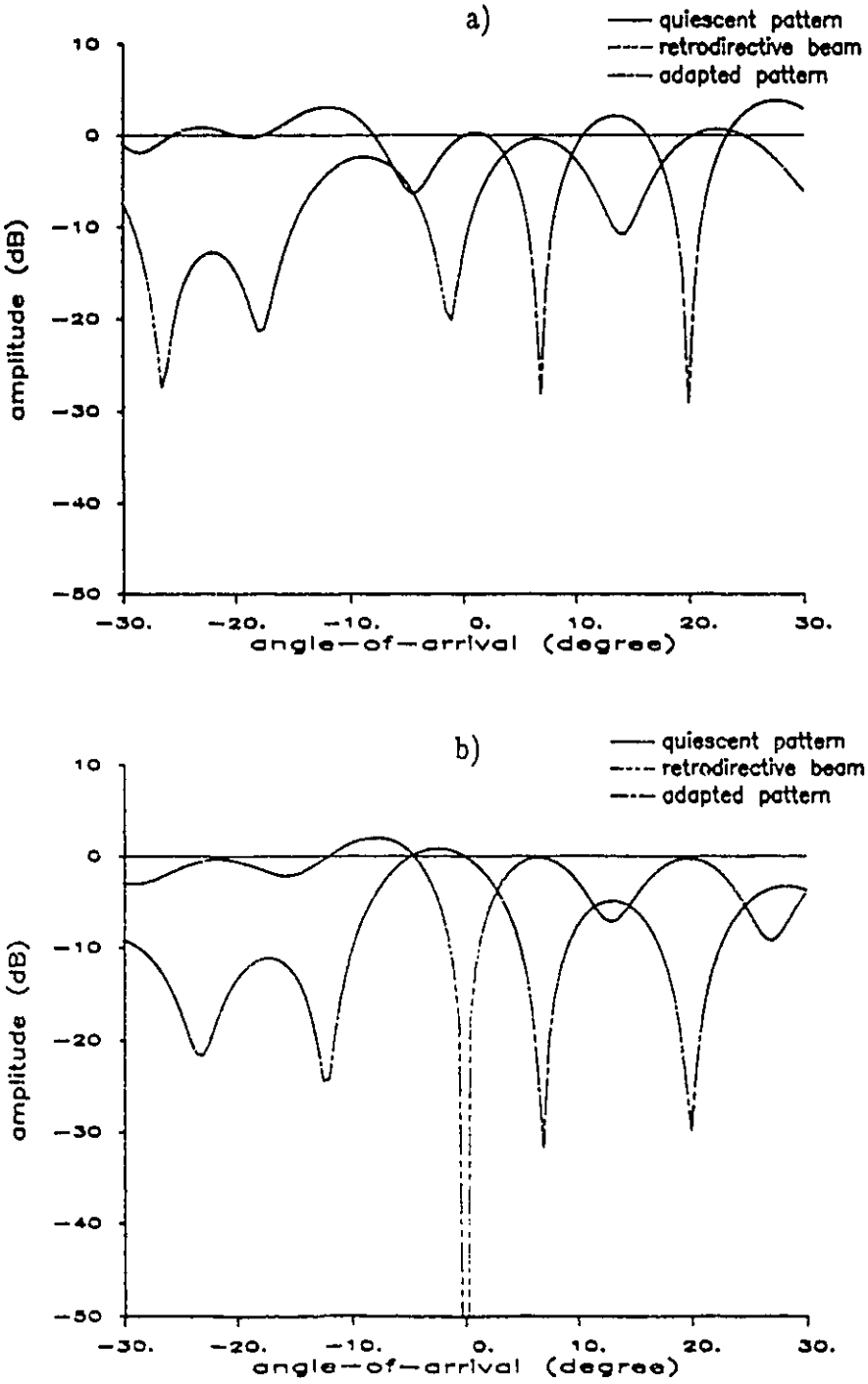


Fig. 4.8 Adapted patterns of a 10-element antenna array using a clamped element as the primary signal, jammers at 7^{O} and 20^{O} after 50 iterations
a) unconstrained b) constrained method

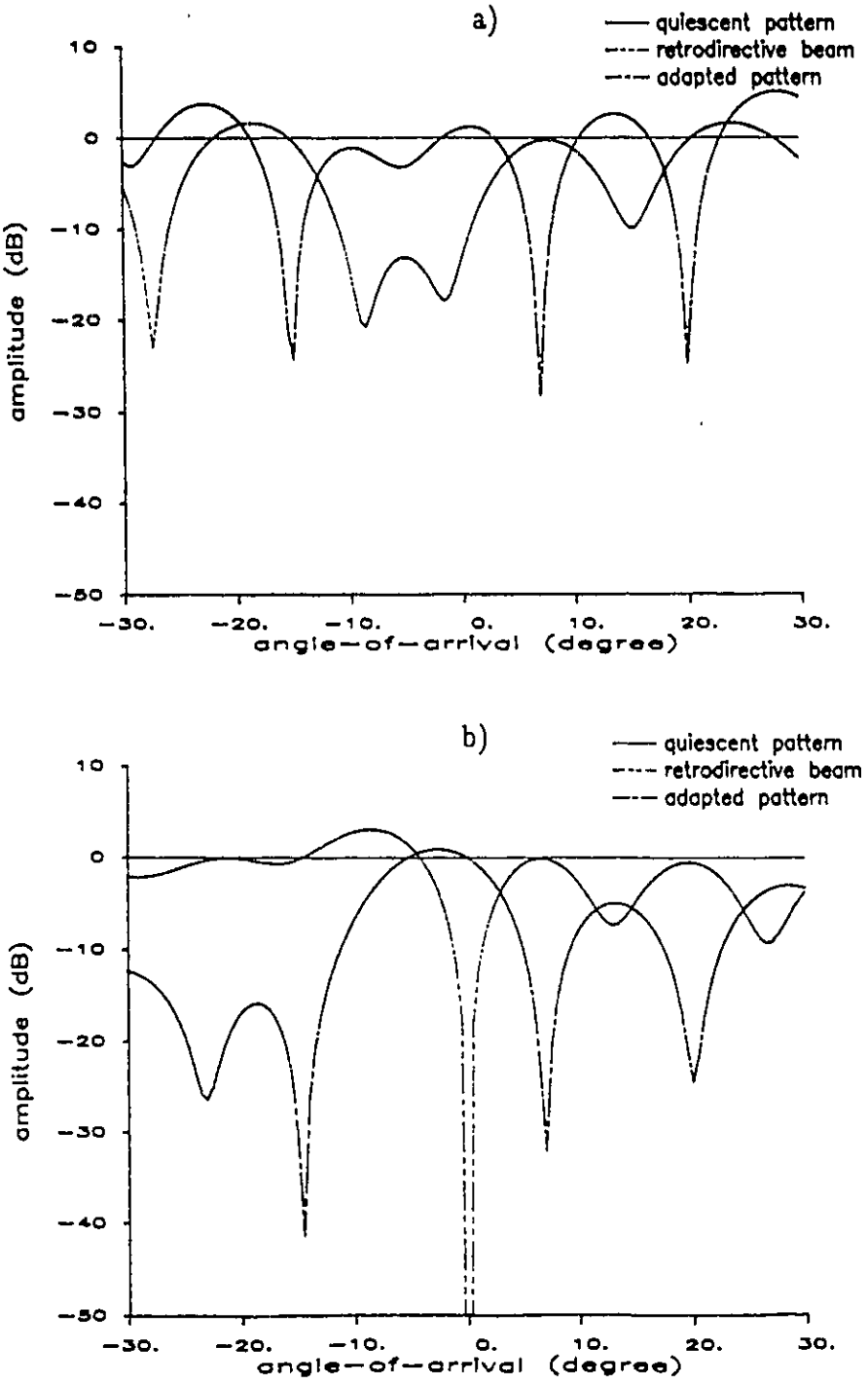


Fig. 4.9 Adapted patterns of a 10-element antenna array using a clamped element as the primary signal, jammers at -15^{0}, 7^{0} and 20^{0}
a) unconstrained b) constrained method

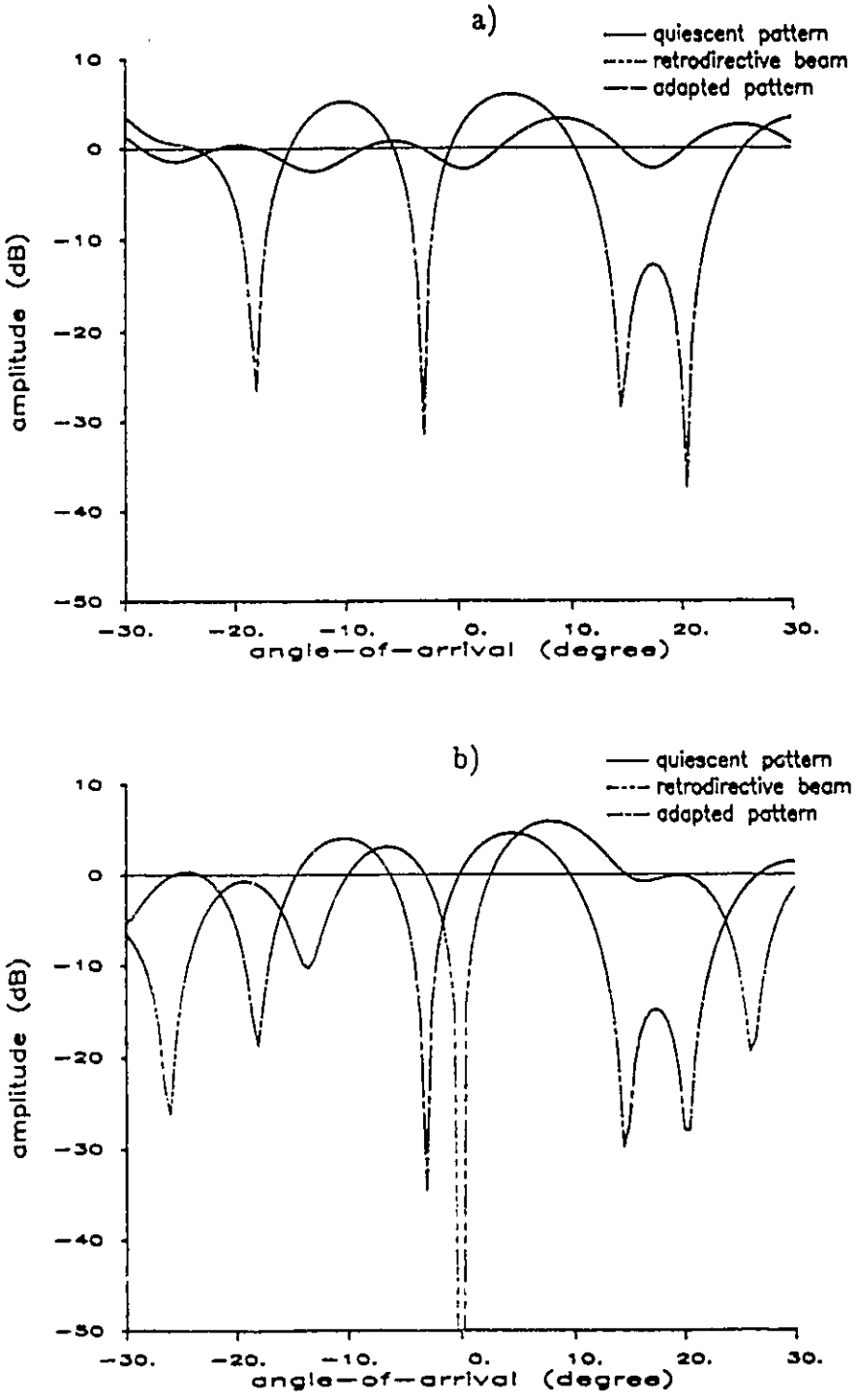


Fig. 4.10 Adapted patterns of a 10–element antenna array using a clamped element as the primary signal, jammers at -18^{o}, -3^{o}, 15^{o} and 20^{o}

a) unconstrained b) constrained method

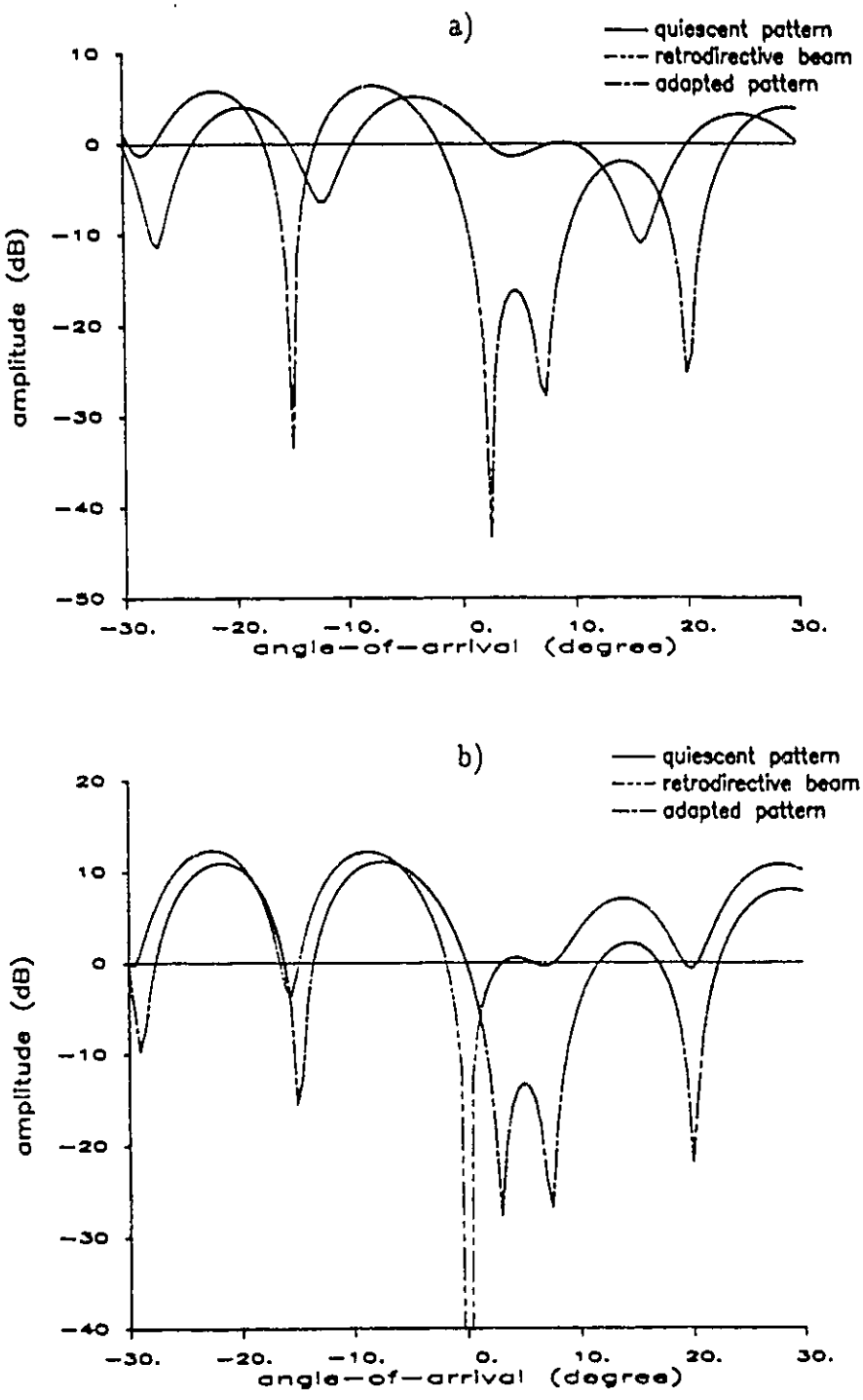


Fig. 4.11 Adapted patterns of a 10-element antenna array using a clamped element as the primary signal, jammers at -15^{o}, 3^{o}, 7^{o} and 20^{o}
a) unconstrained b) constrained method

respectively, using the same simulated data described in Table 4.1. In this instance, the first element of the 10–element array is used as the primary channel and the other nine elements are used as the auxiliary channels. As it can be seen the results are similar to those that were obtained using a high gain antenna and which were given in Figs. 4.2a to 4.5a. Again, deep nulls are generated in the direction of the interfering signals and the antenna main beam is distorted as the jammers enter its 3–dB beamwidth.

4.2.2. *Linearly Constrained Method*

As discussed in section 3.3 and as shown in the last section, adaptive beamforming with an element clamped and used as the primary channel is a special case of the linearly constrained adaptive beamforming. In this case, the constrained vector $\mathbf{c} = [1, 0, 0, ..., 0]^T$, and thus the results given in Figs. 4.8a to 4.11a are readily applied. In this section, a linearly constrained adaptive beamforming or the MVDR beamforming is presented.

Consider a constrained vector $\mathbf{c}(\theta_d)$, which satisfies

$$\mathbf{c}^T(\theta_d)\mathbf{w}(n) = 1 \tag{4.2}$$

where

$$\mathbf{c}(\theta_d) = [1, e^{-j\varphi_d}, e^{-j2\varphi_d}, ..., e^{-j(M-1)\varphi_d}]$$

with

$$\varphi_d = \frac{2\pi d}{\lambda} \sin\theta_d, \quad M = 10.$$

It is noted that the constraint given by Eqn. (4.2) is to maintain a constant gain in the desired look direction, θ_d. The adaptation may be carried out by using the adaptive processor in Fig. 3.4. However, as pointed out, more circuitry is required in the implementation because of the use of the pre–processor network. In addition, the adaptive processor is limited to a single constraint.

The implementation in Fig. 3.3 is well suited for multiple linearly constrained adaptive beamforming. In particular, in the case of the MVDR adaptive beamforming its first row can be used as the constrained pre–processors. Initially, the constrained vector, $\mathbf{c}(\theta_d)$, enters the triarray processor for adaptation and is transformed into vector $\mathbf{c}'(\theta_d)$, which is stored in the PE cells in the first row. The adaptation in this first row is subsequently suppressed by setting $\alpha_1 = 0$. As the input data of the received antennas arrived at the triarray processor they experience a pre–processing performed by the first row (Eqn. (3.39)) before entering the adaptation, which is carried out by the rest of the rows of the systolic array processors.

Figures 4.2b to 4.5b show the results obtained with a linear constraint, in which $\theta_d = 0^{\circ}$, using the same simulated data given in Table 4.1. It is noted that in each of the above figures the retrodirective pattern forms a null at the constrained direction (θ_d=0°), which in effect eliminates any degradation at the wanted direction during the adaptation. As a result, it has been shown that the gain of the antenna pattern remains constant at the desired looking direction. The consequence of applying constraints is that it eliminates one degree of freedom for each constraint, which will further reduce the nulling capability of the adaptive processor as the number of jammers increases. In practice, most antenna arrays used are large and operate in the environments with the number of interferers

much less than the array degrees of freedom, M–1; thus the use of one or two linear constraints will not significantly affect to the performance characteristics of the antenna systems. This can be easily verified by comparing Figs. 4.2b to 4.5b with the unconstrained adaptive beamforming results shown in Figs. 4.2a to 4.5a, respectively.

Adaptive beamforming results with a constant gain in the desired look direction using a clamped element are given in Figs. 4.8b to 4.11b, respectively. Again, as it can be readily seen the antenna gain at the constrained look direction (θ_d=0^o) remains constant whilst interferers are suppressed. In fact, the linearly constrained adaptive beamforming technique is very useful in practice since the adaptation can be carried out with the presence of the desired signals. With the use of constrained vectors, the adaptive processor is capable of preventing the desired look directions from being nulled out or distorted during adaptation process.

4.3. 2–D Adaptive Beamforming

In this section, simulation studies are given to describe the performance of the 2–D adaptive beamforming algorithms that were presented in Chapter 2 and 3. It can be stated that all of the 2–D adaptive beamforming algorithms converge to the optimum solution that is given by the Wiener–Hopf equation, even though the starting points of their derivations were quite different. Where they differ is in their robustness, i.e., the rate at which they converge and the numerical stability they exhibit when processing ill–conditioned data. Their performance relative to one another largely parallels that observed for 1–D beamforming. Since; 1) the 1–D case is widely discussed in the literature, and 2) the performance of the 2–D

algorithms can be inferred from the 1–D case, a comparison of the performance of the 2–D algorithms will not be presented here.

In order to simplify the presentation of the results which follow we choose to change the notation commonly used to denote elevation and azimuth angles. Rather than using $(90^{\circ}-\theta)$ and ϕ we select to use θ_e and θ_a. The simple transformation is given by

$$(\theta_e,\theta_a) \longmapsto (90^{\circ}-\theta,\phi)$$

where

$$\theta_e = \tan^{-1}(\tan\theta \sin\phi).$$

and

$$\theta_a = \tan^{-1}(\tan\theta \cos\phi)$$

Figure 4.12 shows the steady–state adapted beam pattern for a 4×4 antenna array. The desired signal direction coincides with the antenna boresight and a jammer with an interference–to–noise ratio (INR) of 40 dB is located at $\theta_e = 48^{\circ}$, and $\theta_a = -10^{\circ}$. The quiescent pattern for an array with uniformly weighted elements is given in Fig. 4.13. The eigenbeam pattern corresponding to the interferer in Fig. 4.12 is given Fig. 4.14. Figure 4.12 was derived using the 2–D Howells–Applebaum algorithm. The derivation of the 2–D eigenbeam pattern is based on the procedure described in section 2.3.4. In Fig. 4.14 it is observed that the main lobe of the eigenbeam pattern coincides with the AOA of the unwanted signal and its amplitude is equal to the magnitude that the interfering signal would have if it were received by the quiescent pattern in Fig. 4.13. This results in a deep null in the direction of the interferer in Fig. 4.12. The interference null is further illustrated in Figs. 4.15a and 4.15b, which show cross sections through the

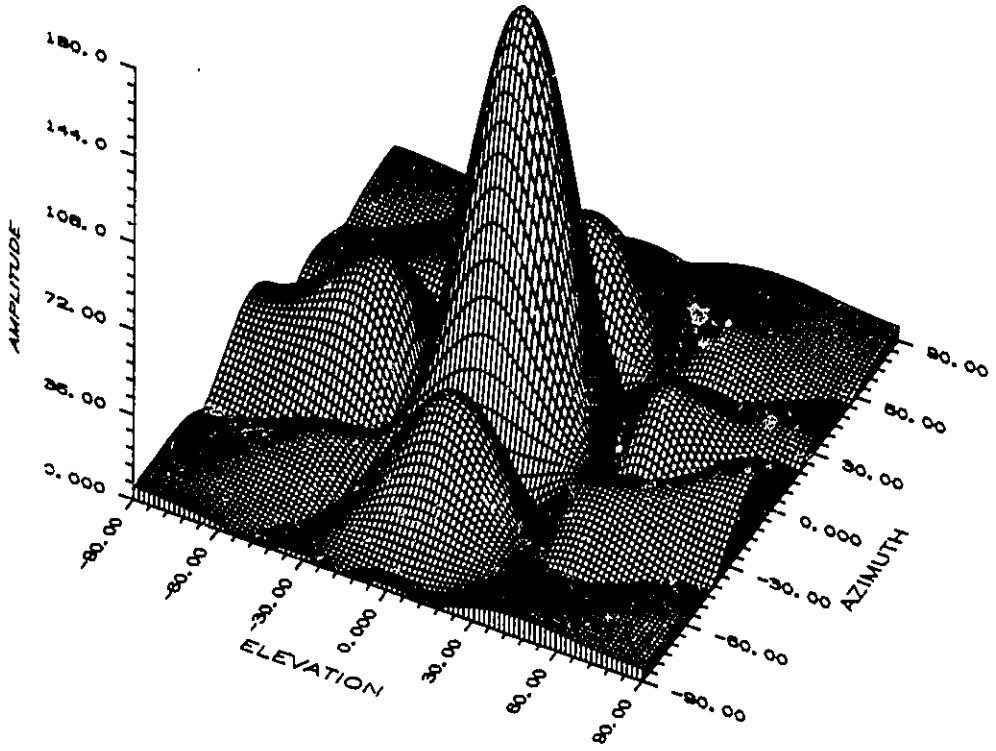


Fig. 4.12 Adapted pattern of a 4×4 array, one jammer at $(48^{O},-10^{O})$ from the boresight

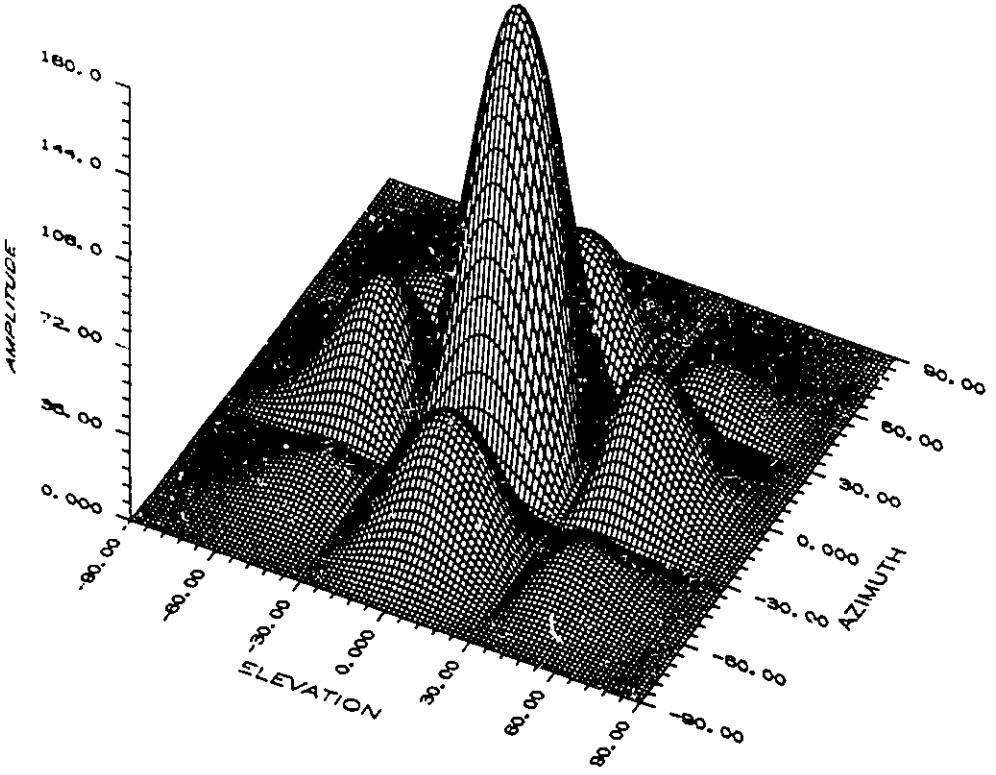


Fig. 4.13 Quiescent beam pattern of a 4×4 array

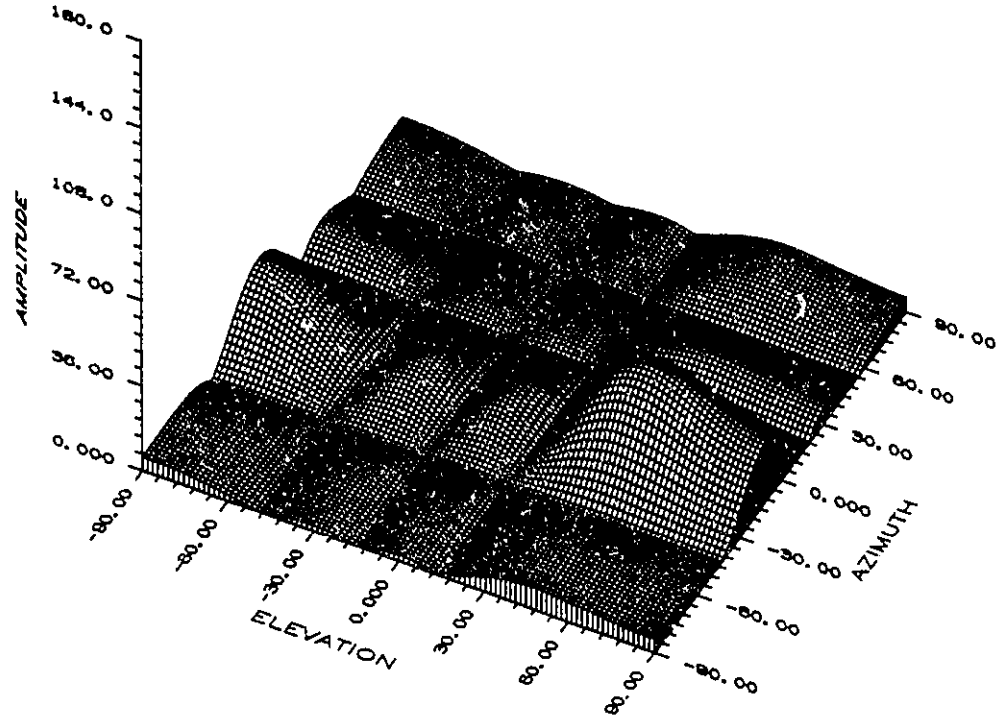


Fig. 4.14 Eigenbeam pattern of the adapted pattern in Fig. 4.12

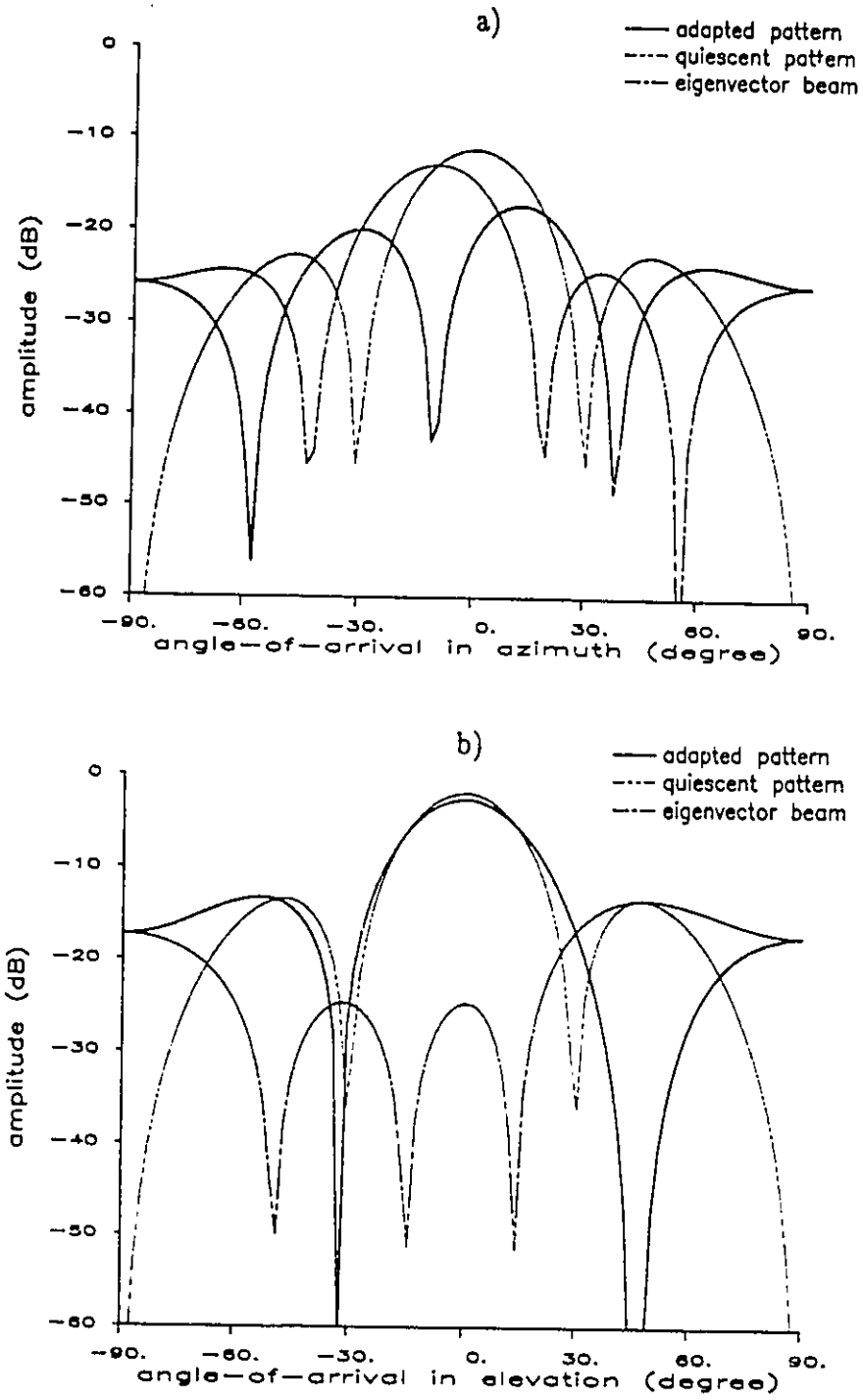


Fig. 4.15 Cross section of the 2-D adapted pattern in Fig. 4.12
a) at 48^{0} in elevation b) at -10^{0} in azimuth

interfering source along the elevation and azimuth planes. Note that a deep null is observed in both figures at the expected jammer location. These results support the contention made earlier that adaptive beamforming can be carried out by operating independently on the rows and columns of the 2–D arrays.

Results from a 3–D systolic array, which is configured for the 2–D QRD–LS algorithm, are now presented. The characteristics of the systolic array are described in Section 3.4. Figures 4.16a and 4.16b give the adapted pattern after 20 and 50 iterations, respectively. The primary vector y(n), which is required for implementing the QRD–LS algorithm was derived from the unweighted 4×4 planar array. That is, the same data were used for the primary vector as used to form the quiescent pattern in Fig. 4.13. It can be seen that, as in the case of the 2–D Howells–Applebaum algorithm given above, null occurrence is observed at the direction of the interferer, (48^{O}, -10^{O}), in both Figs. 4.16 and 4.17. Also, a significant increase of sidelobe levels is observed in the adapted pattern. This effect however is not usually important to the overall performance of an adaptive antenna system, since the SINR is decided solely by the interference–nulling capability of the adaptive processor, i.e., the depth of nulls in the adapted pattern at the unwanted signals.

The curves of the residual beamforming output power, $\|e(n)\|^2$, of the 2–D array are given in Fig. 4.17. These results are derived by using the 2–D LMS, the SMI and the QRD–LS algorithms. It can be seen that the residual curve of the QRD–LS algorithm converges quickly to the noise floor (–40 dB) and becomes quite stable after only a few iterations (snapshots) following initialization. The results with the LMS and SMI algorithms, on the other hand, converge slowly and exhibit large fluctuations. Typically, the convergence time for the QRD–LS

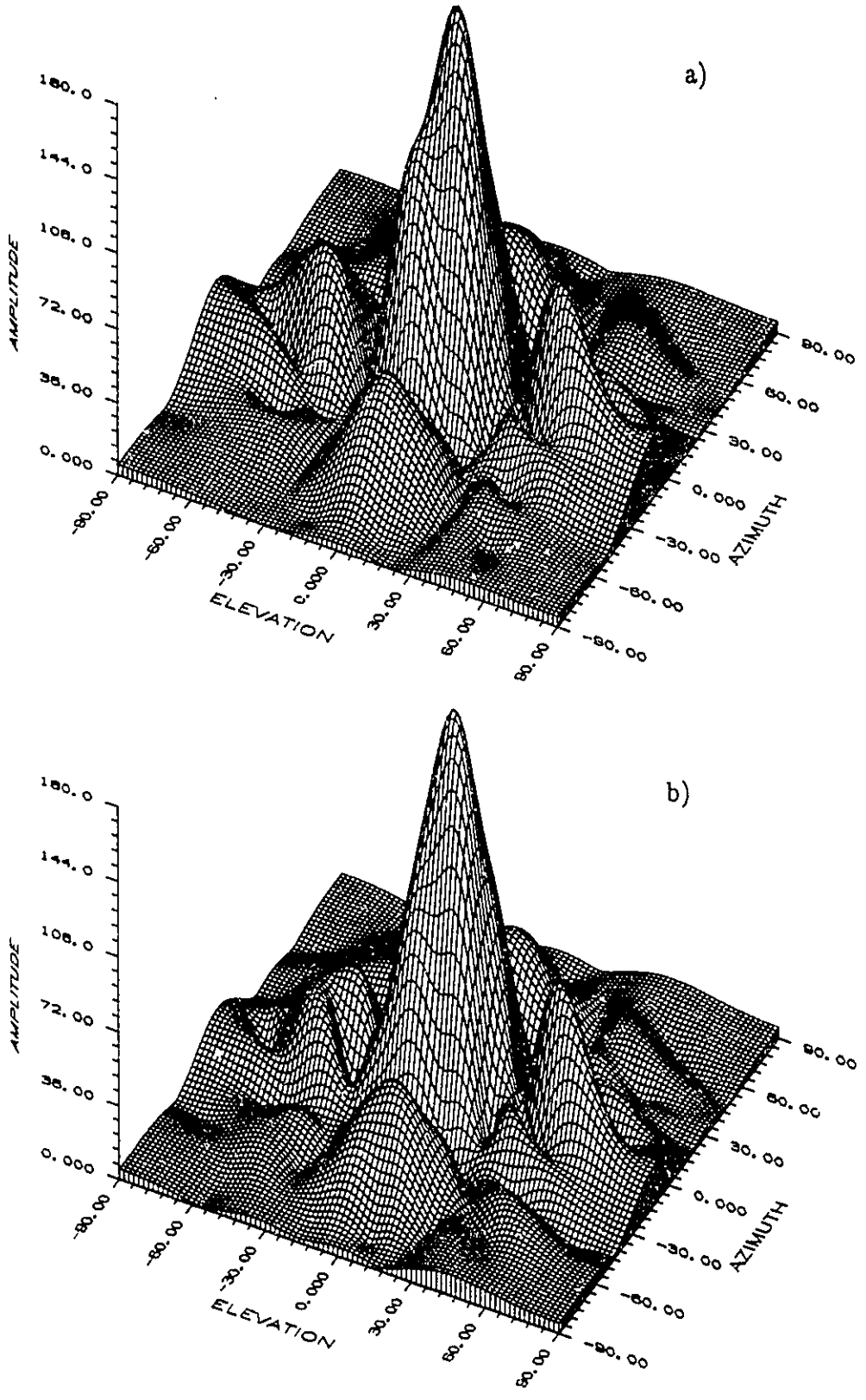


Fig. 4.16 Adapted pattern of a 4×4 array, one jammer at (48^{O},-10^{O})using the QRD--LS algorithm
a) after 20 iterations b) after 50 iterations

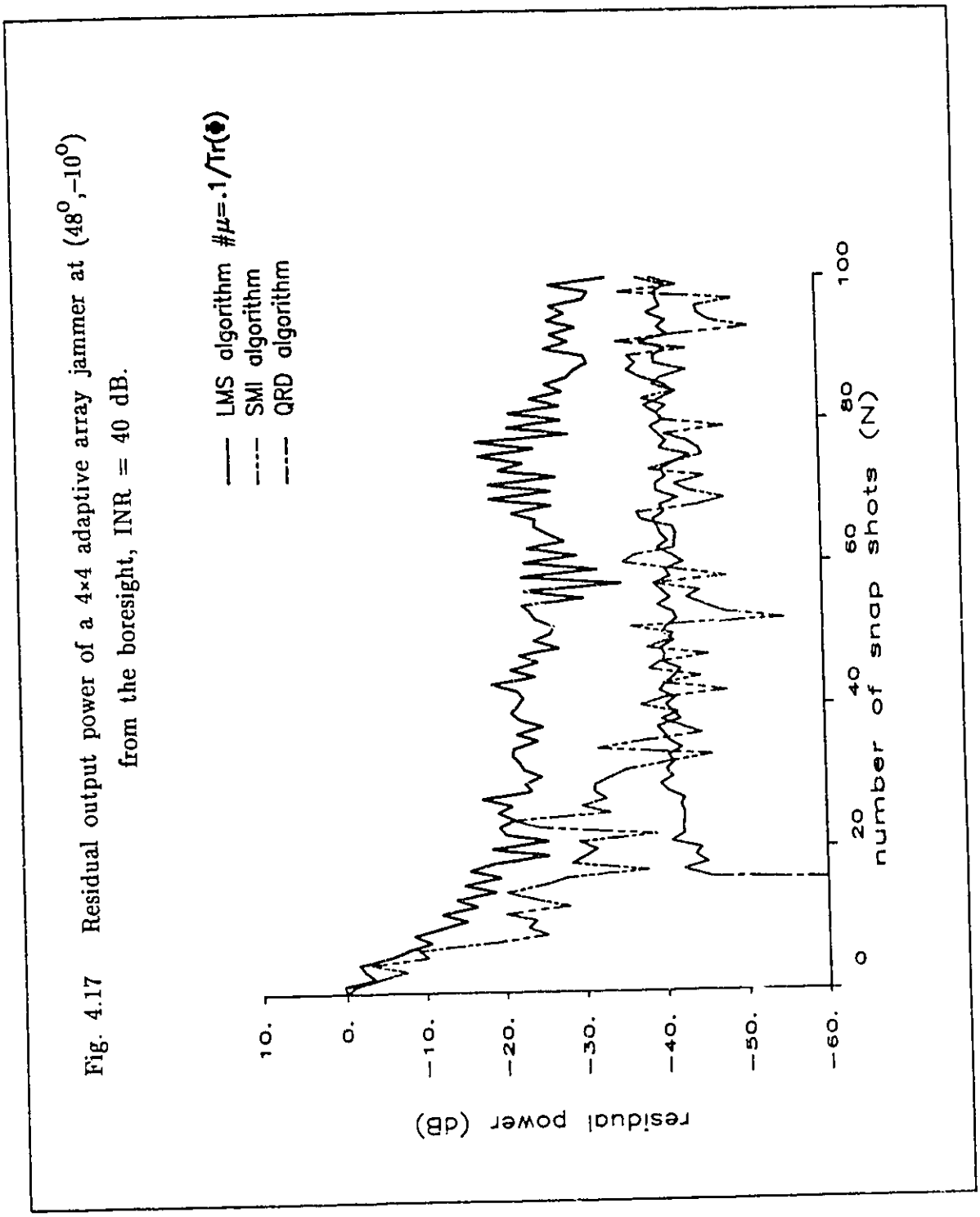


Fig. 4.17 Residual output power of a 4×4 adaptive array jammer at $(48^{o},-10^{o})$ from the boresight, INR = 40 dB.

implemented with systolic array is about $4M$ (=16) clock cycles, which is consistent with the discussion in the last section.

Figure 4.18 gives the steady state adapted pattern for two jammers, one locates at $(48^{o},-10^{o})$ and the other at $(-15^{o},-50^{o})$. Both jammers are 40 dB above the thermal noise level, i.e.,

$$\mathrm{INR}_1 = \mathrm{INR}_2 = 40 \text{ dB}.$$

The retrodirective (eigenbeam) patterns corresponding to the two interference directions are given in Fig. 4.19. As in the last example, the main lobes of the eigenbeam coincide precisely with the jammer directions, thereby producing deep nulls, as illustrated in Figs. 4.18. The results obtained by using the QRD–LS algorithm are given in Figs. 4.20a and 4.20b, respectively. In these figures, it is to be noted that the adapted patterns are degraded with the increase of sidelobe levels; however, locations of interference nulls are very well defined. These results again verified the performance characteristics of the 3–D systolic array processor for 2–D adaptive beamforming. That is to say the adaptive nulling process of a 2–D array can be achieved by operating along the columns and rows of the array. The curves of the residual output power are given in Fig. 4.21. As it can be seen the convergence rate is noticeably slower than that observed in the last example. This is because the rate at which the beamforming processor carries out the adaptive weights becomes increasingly sluggish as the number of interferers increases.

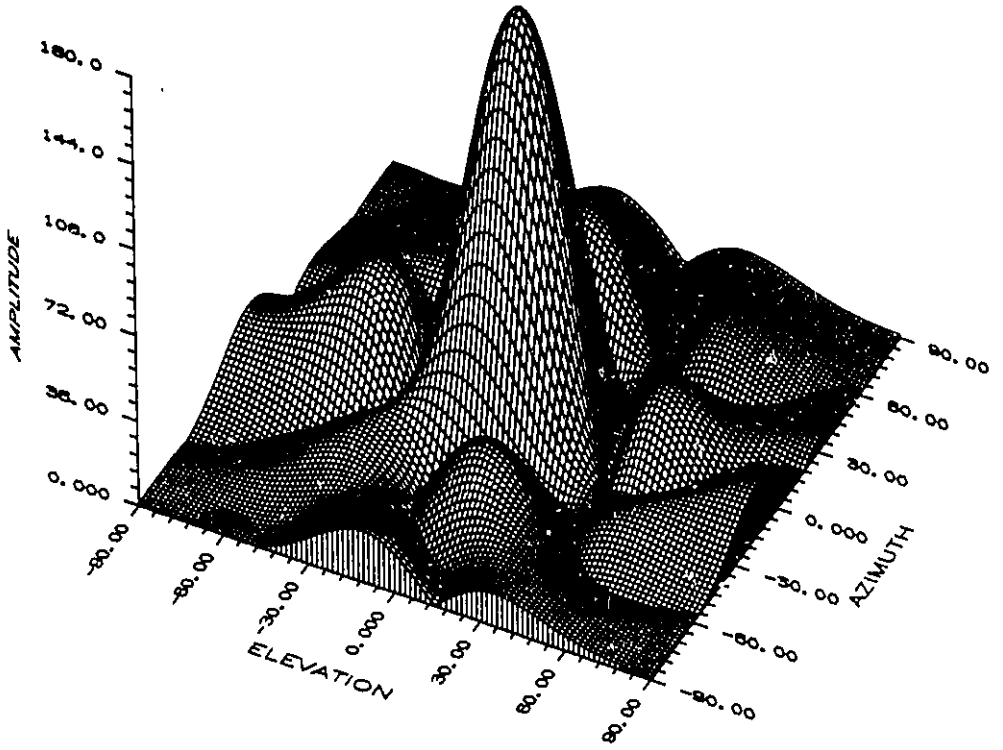


Fig. 4.18 Adapted pattern of a 4×4 array, two jammers at $(48^{O},-10^{O})$ and $(-15^{O},-50^{O})$ from the boresight

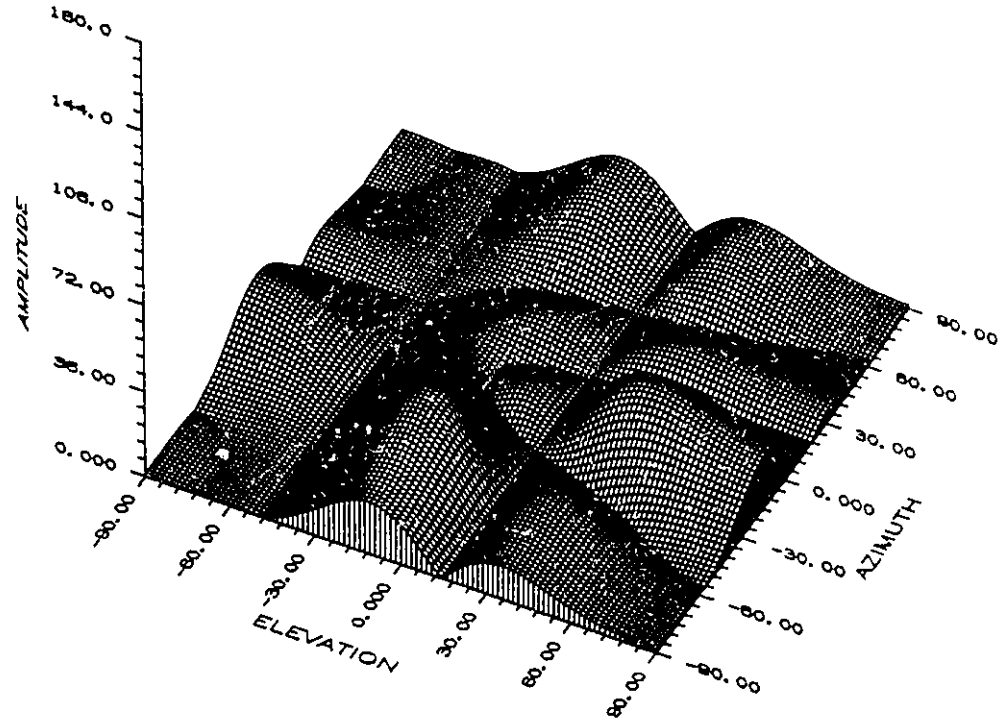


Fig. 4.19 Retrodirective pattern of the adapted pattern in Fig. 4.18

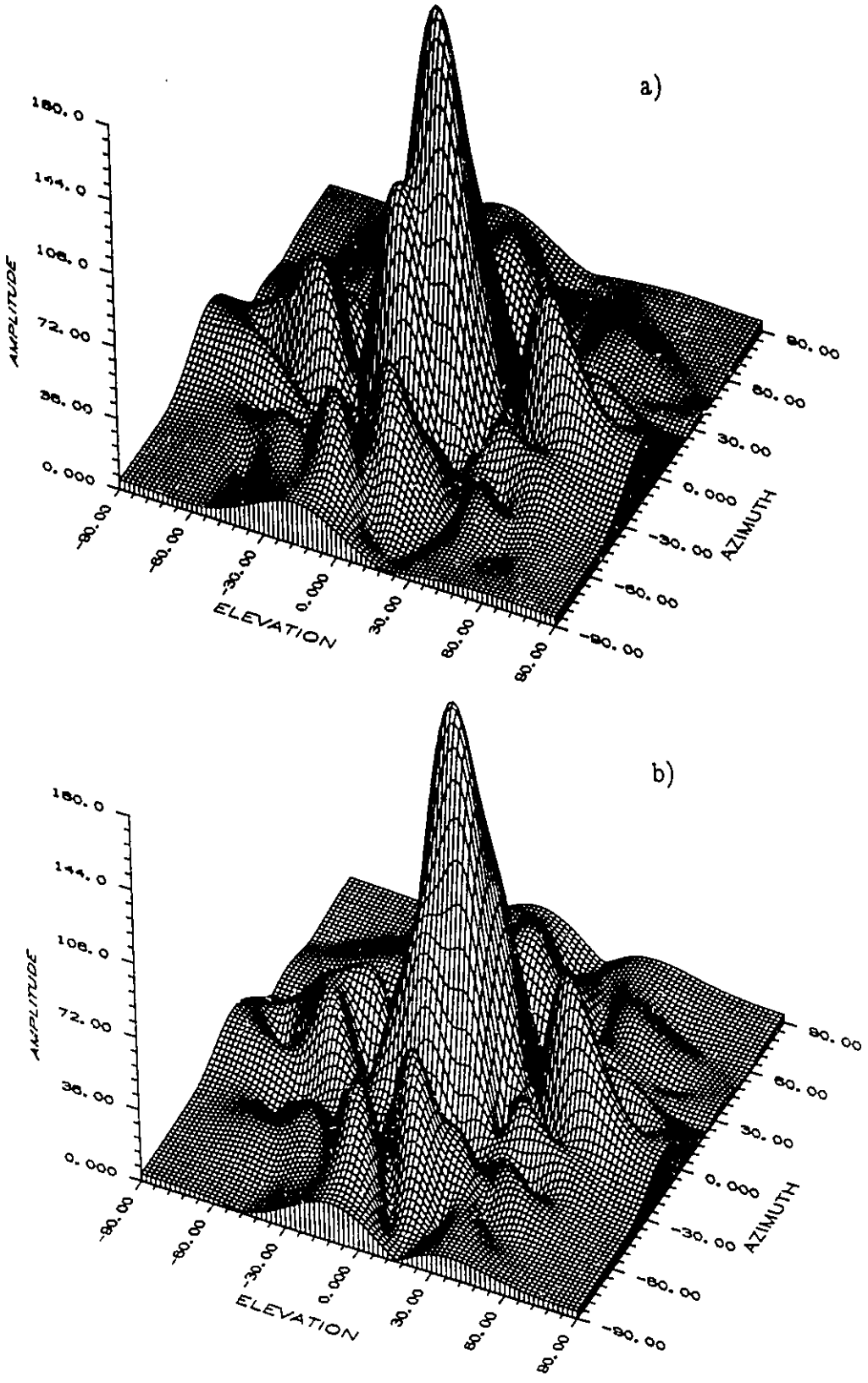


Fig. 4.20 Adapted pattern of a 4×4 array, two jammers at $(48^{0},-10^{0})$ and $(-15^{0},-50^{0})$ using the QRD-LS algorithm
a) after 20 iterations b) after 50 iterations

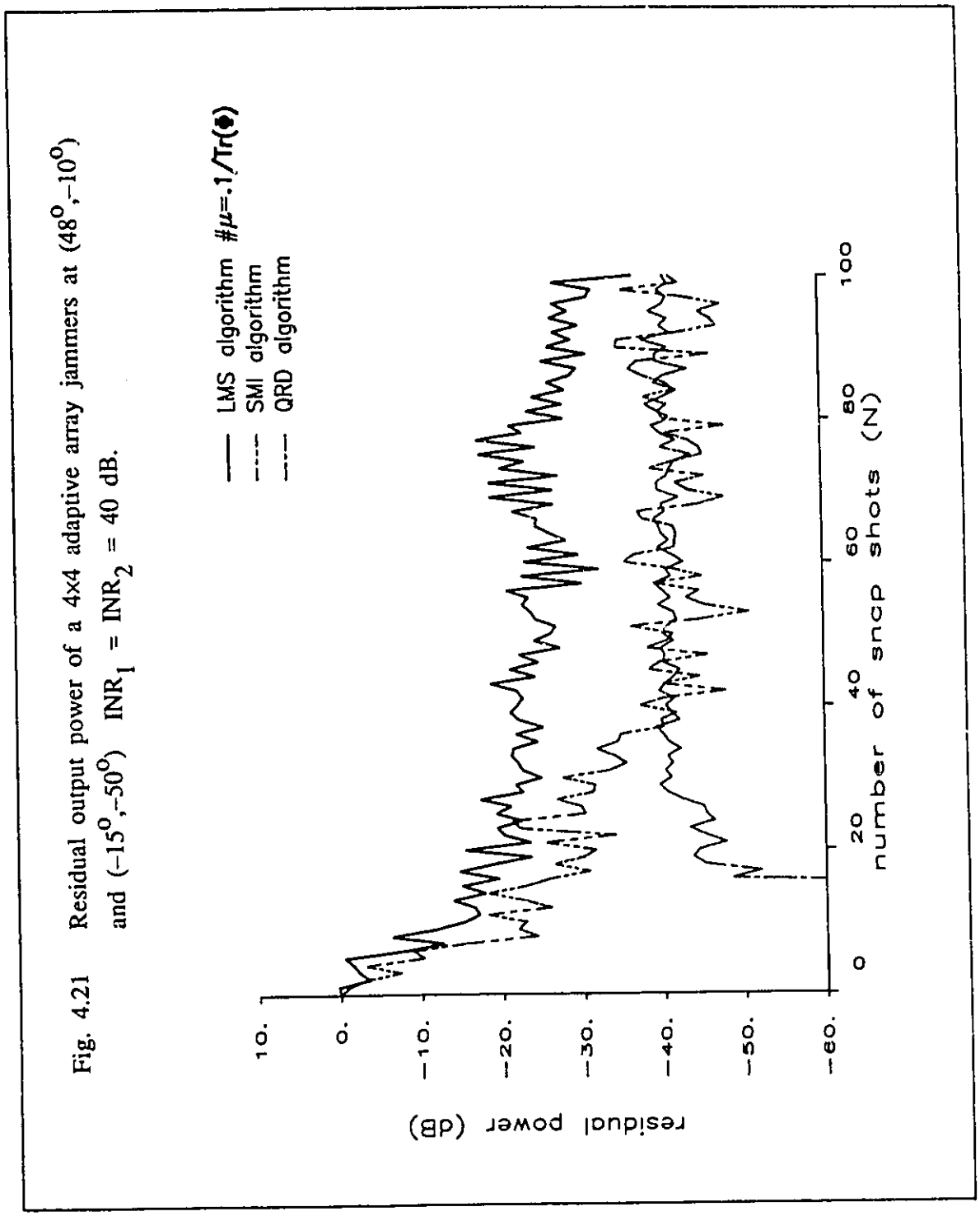


Fig. 4.21 Residual output power of a 4×4 adaptive array jammers at $(48^{o},-10^{o})$ and $(-15^{o},-50^{o})$ $INR_1 = INR_2 = 40$ dB.

4.4. Concluding Remarks

Performance of the adaptive beamforming algorithms has been examined in this chapter. In particular, the use of systolic array for implementing adaptive beamforming processors has been emphasized. Results obtained by using both unconstrained and constrained adaptive beamforming methods have been presented. Both the 1–D and 2–D adaptive beamforming results have been given. In essence, it has been shown that an adaptive processor implemented with the QRD–LS algorithm amenable to systolic array is highly stable and rapidly converges as compared to the others such as the LMS algorithm and the SMI algorithm. As well, the QRD–LS algorithm deals quite well with the ill–conditioned scenarios, while the LMS and the SMI algorithm will become sluggish and highly fluctuated (unstable) as the environments are constantly varied.

Results obtained with 2–D adaptive beamforming algorithms have verified the analysis given in Chapter 2 and 3. Especially, the role of 2–D eigenbeams in the interference nulling process has been illustrated. Also, performance of the 3–D systolic array implementation solving the QRD–LS algorithms for 2–D adaptive beamforming has been described. Throughout the simulation results, it has been demonstrated that adaptive beamforming with a planar array can be achieved by computing the QRD–LS algorithms along the rows and columns of the 2–D array.

CHAPTER 5

AN ALGORITHM FOR SELF–CALIBRATING ANTENNA ARRAYS

5.1. Introduction

Most modern array signal processing techniques such as adaptive beamforming and AOA estimation, which have been introduced in the literature in recent years, have assumed that the processed data are correctly calibrated, and at most are corrupted by the presence of additive noise. This assumption, however, hardly ever holds in practice because of the presence of both gain and phase errors due to misalignment of antenna elements, mutual coupling and drift in the electronics associated with the antenna elements. In fact, array calibration is one of the major practical difficulties that must be overcome before high performance algorithms become robust. The problems associated with the sensitivity of algorithm to gain and phase errors have been extensively studied in the literature [27–29]. These errors can greatly reduce the resolving performance of high performance AOA estimation algorithm, increase sidelobe levels, and limit the depth of the nulls achieved for nulling out interfering sources. In fact, it has been shown in [27] that optimal beamformers are very sensitive to array errors.

Many of the current workers in array processing have raised the problem, but very few had attempted to solve it [30–32]. In [30], it was suggested that pseudo noise should be injected into the processor to reduce the sensitivity of the system

to array errors. However, the method reduces the system's adaptivity against undesired sources, as well as the output SNR. The eigen–based method proposed in [31] relies heavily on the Toeplitz structure of the array covariance matrix; thus it can only be applied to uncorrelated signals and uniformly spaced arrays. The subspace approach based on the MUSIC algorithm presented in [31] shows some interesting properties in array error estimation and antenna system calibration. However, the method assumes that some of the signal AOAs are known in advance. In practice, this assumption usually fails, especially, when interfering signals are from intelligent jammers or consist of multipath signals. As well, it will be shown in the following sections, the set of derived AOAs is not an optimal one. Very recently, a method [33] which only deals with real gain error was proposed. And in [34], the AOAs are computed recursively via the minimum norm least–squares method. However, it has been pointed out that several local minimum norms exist, which correspond to different sets of AOAs. Thus, neither the algorithm proposed by [9] nor that by [10] is robust because of the non–uniqueness of the solution for the optimal AOAs and the array calibration and/or because of the use of unrealistic assumption in the derivation.

In this Chapter, a self–calibration algorithm, which is capable of computing the optimal AOAs and estimating the gain and phase errors in elements of an array, will be developed. The algorithm is an iterative method. Its derivation is based on the eigen–based method and optimal AOAs are estimated via the least–squares norm minimization of a so called *calibration vector*. Results obtained using both simulated and measured data are presented. It will be shown that the proposed self–calibration algorithm is capable of estimating AOAs with minimum mismatch, even when the signals of interests are correlated, i.e. interferers are multipath signals.

5.2. Signal Model

When one takes into account gain and phase errors, the received signal at the mth element of an antenna array at time t_n, which formerly was given by Eqn. (1.9), is rewritten as

$$x_m(n) = g_m(n) \sum_{k=1}^{K} s_k(n) e^{\jmath(m-1)\varphi_k} + \nu_m(n) \tag{5.1}$$

where

$$\varphi_k = \frac{2\pi d}{\lambda} \sin\theta_k$$

in which $g_m(n)$, a complex constant, represents gain and phase errors in the mth element, and $s_k(n)$ is the baseband signal amplitude of the kth source

From Eqn. (5.1), the received antenna data $\mathbf{x}(n)$ is given by

$$\mathbf{x}(n) = [x_1(n),\ x_2(n),\ \ldots,\ x_M(n)]^T$$

$$= \mathbf{GAs}(n) + \nu(n) \tag{5.2}$$

where G is a diagonal matrix of g_m, i.e.,

$$\mathbf{G} = \text{diag}\ \{\ g_m(n)\ \} \tag{5.3}$$

and A is a matrix of steering vectors $\mathbf{a}(\theta_k)$,

$$\mathbf{A} = [\mathbf{a}(\theta_1), \mathbf{a}(\theta_2), \ldots, \mathbf{a}(\theta_K)] \tag{5.4a}$$

with

$$\mathbf{a}(\theta_k) = [1, e^{-j\varphi_k}, \ldots, e^{-j(M-1)\varphi_k}]^T \tag{5.4b}$$

$$\mathbf{s}(n) = [s_1(n), s_2(n), \ldots, s_K(n)]^T \tag{5.5}$$

and

$$\nu(n) = [\nu_1(n), \nu_2(n), \ldots, \nu_M(n)] \tag{5.6}$$

The array covariance matrix of the received signals is thus given by

$$\Phi_{xx} = \mathrm{E}[\mathbf{x}(n)\mathbf{x}^H(n)]$$

$$= \Phi_{ss} + \sigma^2\mathbf{I} \tag{5.7}$$

where I is an $M \times M$ identity matrix, and

$$\Phi_{ss} = \mathbf{GASA}^H\mathbf{G}^H \tag{5.8}$$

in which

$$\mathbf{S} = \mathrm{E}[\mathbf{s}(n)\mathbf{s}^H(n)]$$

When Φ_{ss} is expanded in terms of its components in eigenspace form, we obtain

$$\Phi_{ss} = \sum_{k=1}^{K} \lambda_k \mathbf{e}_k \mathbf{e}_k^H \tag{5.9}$$

where λ_k and $\mathbf{e}_k$ are the eigenvalues and the eigenvectors of the $M \times M$ matrix Φ_{ss} corresponding to the kth source.

Note that with the inclusion of the error matrix G in Eqn. (5.8), the eigenvalues λ_k in Eqn. (5.9) are implicitly composed of two components: the true eigenvector $\mathbf{e}_k$ and a component that is contributed by errors in the array elements. Nevertheless, if the first K eigenvalues of the matrix Φ_{xx} are large enough so that the number of signal sources, K, can be determined (this condition is usually met in radar applications) then the first K eigenvectors and the last $M-K$ eigenvectors of Φ_{xx} can be used as the estimates of the signal subspace and the noise subspace of the array covariance matrix.

Let $\mathbf{E}_s$ and $\mathbf{E}_\nu$ denote the column matrices represent the signal and the noise subspaces of the matrix Φ_{xx}, i.e.,

$$\mathbf{E}_s = [\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_K] \tag{5.10a}$$

and

$$\mathbf{E}_\nu = [\mathbf{e}_{K+1}, \mathbf{e}_{K+2}, \ldots, \mathbf{e}_M] \tag{5.10b}$$

then $\mathbf{E}_s$ and $\mathbf{E}_\nu$ are orthogonal subspaces.

Note that the signal subspace $\mathbf{E}_s$ spans the same subspace spanned by $\mathbf{a}(\theta_k)$'s, where $\mathbf{a}(\theta_k)$ is the steering vector corresponding to the kth source. Thus, if $\mathbf{E}_\nu$ is truly an exact representation of the noise subspace then

$$\mathbf{E}_\nu^H \mathbf{a}(\theta_k) = 0 \tag{5.11}$$

where 0 is a null vector.

Unfortunately, because of the presence of gain and phase errors in the array elements, $\mathbf{E}_\nu$ will be corrupted, and the equality in Eqn. (5.11) will not usually hold. In fact, the noise subspace $\mathbf{E}_\nu$ is orthogonal to the vector $\mathbf{a}(\hat{\theta}_k)$, where $\hat{\theta}_k$ is an estimate of the true angle of arrival θ_k, where

$$\mathbf{a}(\hat{\theta}_k) = \mathbf{G}\mathbf{a}(\theta_k) \tag{5.12}$$

and $\mathbf{G}$ is a $M \times M$ diagonal matrix as defined in Eqn. (5.3). Equation (5.11) becomes

$$\mathbf{b}_k = \mathbf{E}_\nu^H \mathbf{G}\mathbf{a}(\theta_k) = 0 \tag{5.13}$$

5.3. Self–Calibration Algorithm

5.3.1. *Gain and Phase Error Estimation*

It follows from Eqn. (5.13) that a precise calibration of our antenna system can be made when we have a good estimate of the diagonal error matrix G. The system of linear equations that follows from Eqn. (5.13) is

$$\mathbf{b} = \begin{bmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \\ \vdots \\ \mathbf{b}_K \end{bmatrix} = \begin{bmatrix} \mathbf{E}_\nu^H \mathbf{G}\mathbf{a}(\theta_1) \\ \mathbf{E}_\nu^H \mathbf{G}\mathbf{a}(\theta_2) \\ \vdots \\ \mathbf{E}_\nu^H \mathbf{G}\mathbf{a}(\theta_K) \end{bmatrix} \tag{5.14}$$

where $\mathbf{b}$ is a $K(M-K)\times 1$ vector and is defined as the *calibration vector.*

Now, by utilizing the equivalent notation

$$\mathbf{G}\mathbf{a}(\theta_k) = \Omega(\theta_k)\mathbf{g} \tag{5.15}$$

where $\Omega(\theta_k)$ is a $M\times M$ diagonal matrix whose elements are those of steering vector $\mathbf{a}(\theta_k)$, and $\mathbf{g}$ is a vector composed of the elemental array errors, i.e.,

$$\Omega(\theta_k) = \text{diag}\{\ a_m(\theta_k)\ \} \tag{5.16}$$

and

$$\mathbf{g} = [g_1(n),\ g_2(n),\ \ldots,\ g_M(n)] \tag{5.17}$$

Equation (5.14) then can be expressed as

$$\mathbf{b} = \mathbf{T}\,\mathbf{g} \tag{5.18}$$

where

$$\mathbf{T} = \begin{bmatrix} \mathbf{T}_1 \\ \mathbf{T}_2 \\ \vdots \\ \mathbf{T}_K \end{bmatrix} = \begin{bmatrix} \mathbf{E}_\nu^H\Omega(\theta_1) \\ \mathbf{E}_\nu^H\Omega(\theta_2) \\ \vdots \\ \mathbf{E}_\nu^H\Omega(\theta_K) \end{bmatrix} \tag{5.19}$$

The matrix $\mathbf{T}$ in Eqn. (5.19) of dimension $K(M-K)\times M$ is defined as the *projection calibration matrix.*

It should be noted from Eqn. (5.18) that in the case of error free array

elements the vector g is a unit vector (all elements are equal to 1) and the calibration vector b is a null vector. In this case, the matrix T must be a singular matrix of rank M–1, i.e., its Mth singular value $\beta_M = 0$. On the other hand, when gain and phase errors are present in array elements, and if $K(M–K) \geq M$, and furthermore, if the rank of the matrix T is M–1, then the right singular vector $\mathbf{v}_M$ of the matrix T is the least–squares norm estimate of the error vector g. As well, if these conditions hold, the angles θ_k's, which are used to form the matrices $\Omega(\theta_k)$'s, are the minimum norm least–squares estimates of the AOAs.

By utilizing the SVD, the matrix T is decomposed into the form

$$\mathbf{T} = \mathbf{U}\Sigma\mathbf{V}^H \tag{5.19a}$$

where U is a $K(M–K)\times K(M–K)$ column matrix of the left singular vectors, V is a $M\times M$ column matrix of the right singular vectors, i.e.,

$$\mathbf{U} = [\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_{K(M-K)}] \tag{5.19b}$$

and

$$\mathbf{V} = [\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_M] \tag{5.19c}$$

and Σ is a diagonal matrix of the singular values β_m, in which

$$\beta_1 \geq \beta_2 \geq \cdots \geq \beta_{M-1} > \beta_M$$

Then, if $\beta_M \simeq 0$,

$$\hat{\mathbf{g}} = \mathbf{v}_M \tag{5.20}$$

from which it follows that the least–squares norm

$$\|\mathbf{T}\,\hat{\mathbf{g}}\|^2 \text{ is minimized.} \tag{5.21}$$

It should be noted that the vector $\hat{\mathbf{g}}$ has been normalized, i.e., its leading element $g_0 = G_{00} = v_{M,1} = 1$; thus it is within an arbitrary complex scaling factor of the true vector $\hat{\mathbf{g}}$. However, as it will be seen in what follows that this scaling factor will not affect the calibration procedure because of the errors that it introduces into the array covariance matrix.

The calibration of the antenna system can be carried out by first correcting the array covariance matrix given by Eqn. (5.7) as follows

$$\hat{\Phi}_{xx} = \hat{\Phi}_{ss} + \sigma^2\mathbf{I} \tag{5.22}$$

where

$$\begin{aligned}\hat{\Phi}_{ss} &= \hat{\mathbf{G}}^{-1}\Phi_{ss}(\hat{\mathbf{G}}^H)^{-1} \\ &= \hat{\mathbf{G}}^{-1}\mathbf{GASA}^H\mathbf{G}^H(\hat{\mathbf{G}}^H)^{-1}\end{aligned} \tag{5.23}$$

where $(.)^{-1}$ denotes the inverse of a matrix.

It follows from Eqn. (5.23) that if $\hat{\mathbf{G}}$ is an exact estimate of the error matrix G then

$$\hat{\Phi}_{ss} = \Phi_{ss} = \mathbf{ASA}^H \tag{5.24}$$

i.e., the calibration errors are totally removed from the system covariance matrix Φ_{xx}.

Unfortunately, the rank of the matrix T in general will be equal to M. That is, its Mth singular value does not equal zero, and thus no solution for the vector g will exist. Obviously, in this case the only solution for g that satisfies the condition $\mathbf{b} = 0$ is that $\mathbf{g} = 0$, which is not a non–trivial solution for the array error vector. In [32], it was proposed that $\mathbf{v}_M$ be used as an approximate solution for g. This means that the calibration process is carried under the assumption that some of the AOAs are known in advance. This assumption however does not often hold in practice, especially when the signals of interest are intelligent jammers or when the interferers are multipath signals. As well, the set of AOA estimates is not truly an optimal set. The *self–calibration algorithm*, which is described in the following, will be shown to be capable of automatic searching for an optimal set of AOAs. The development of the algorithm is based on the eigen–based structure method and a recursive minimum norm least–squares optimization of the vector b.

5.3.2. *Self–Calibration Algorithm*

A criterion for an optimal estimate of the calibration error vector g can be expressed as follows. Given a set of AOAs, $\hat{\theta}_k$'s, defined by steering vectors $\mathbf{a}(\hat{\theta}_k)$'s, and the noise subspace $\mathbf{E}_\nu$, find an optimal estimate of the vector g, denoted by $\hat{\mathbf{g}}$, to minimize the residual norm $\|\hat{\mathbf{T}}\hat{\mathbf{g}}\|^2$, where

$$\hat{\mathbf{T}} = \begin{bmatrix} \hat{\mathbf{T}}_1 \\ \hat{\mathbf{T}}_2 \\ \vdots \\ \hat{\mathbf{T}}_K \end{bmatrix} = \begin{bmatrix} \mathbf{E}_\nu^H \Omega(\hat{\theta}_1) \\ \mathbf{E}_\nu^H \Omega(\hat{\theta}_2) \\ \vdots \\ \mathbf{E}_\nu^H \Omega(\hat{\theta}_K) \end{bmatrix}. \tag{5.25}$$

This procedure is a least squares minimization problem.

It is followed from Eqns. (5.20) and (5.21) that the optimal estimate of $\hat{\mathbf{g}}$ for a given set of $\hat{\theta}_k$'s is $\hat{\mathbf{g}} = \mathbf{v}_M$, and the minimum norm estimate of $\epsilon(\hat{\theta}_k$'s) is given by

$$\epsilon(\hat{\theta}_k\text{'s}) = \|\hat{\mathbf{T}}\mathbf{v}_M\|^2. \tag{5.26}$$

Furthermore, the set of the minimum norms, denoted by $C(\epsilon)$, is

$$C(\epsilon) = \{\epsilon(\hat{\theta}_k\text{'s})\} \tag{5.27}$$

where the angles $\hat{\theta}_k$ are recursively computed using the MUSIC algorithm [52], i.e., the K highest peaks of the spatial spectrum

$$H(\theta) = \frac{1}{\|\mathbf{E}_\nu^H \mathbf{G}\ \mathbf{a}(\theta)\|^2} \tag{5.28}$$

where

$$\mathbf{a}(\theta) = [a_1(\theta),\ a_2(\theta),\ \ldots,\ a_m(\theta),\ \ldots,\ a_M(\theta)] \tag{5.29a}$$

with

$$a_m(\theta) = e^{-j(m-1)\varphi}. \tag{5.29b}$$

The overall optimal estimate of the vector $\hat{\mathbf{g}}^o$ is defined by

$$\hat{\mathbf{g}}^{O} = \{ \hat{\mathbf{g}} / \epsilon(\hat{\theta}_k\text{'s}) \in C(\epsilon) : \text{minimum} \} \tag{5.30}$$

and the set of $\hat{\theta}_k$'s corresponding to the optimal vector $\hat{\mathbf{g}}^{O}$ is the optimal estimates of the AOAs, $\hat{\theta}_k^{O}$'s.

In practice, however, the search for an optimal set of the AOAs is very time–consuming. As well, there often exists several local minimums of $\epsilon(\hat{\theta}_k\text{'s})$ that correspond to different sets of $\hat{\theta}_k$'s [34]. From this it appears that if a reference look direction is provided, then a unique set of the AOAs can be specified. Furthermore, the use of a reference signal will significantly reduce the computations required to carry out an iterative process.

In summary, the algorithm for self–calibrating antenna arrays is illustrated in Fig. 5.1, which can be summarized as follows.

1. Form the array covariance matrix Φ_{xx}= $E[\mathbf{x}(n)\mathbf{x}^{H}(n)]$, and estimate the noise subspace $\mathbf{E}_{\nu}$.

2. Utilize the subspace method (MUSIC) defined by Eqn. (5.28) to estimate the AOAs, $\hat{\theta}_k$'s, with $\mathbf{G}_{initial}$ = I, an identity matrix.

3. For each set of $\hat{\theta}_k$'s, form the estimate of the projection covariance matrix $\hat{\mathbf{T}}$ defined by Eqn. (5.25), where $\hat{\theta}_1 = \theta_{reference}$, is assumed to known in advance. Compute the least–squares norm

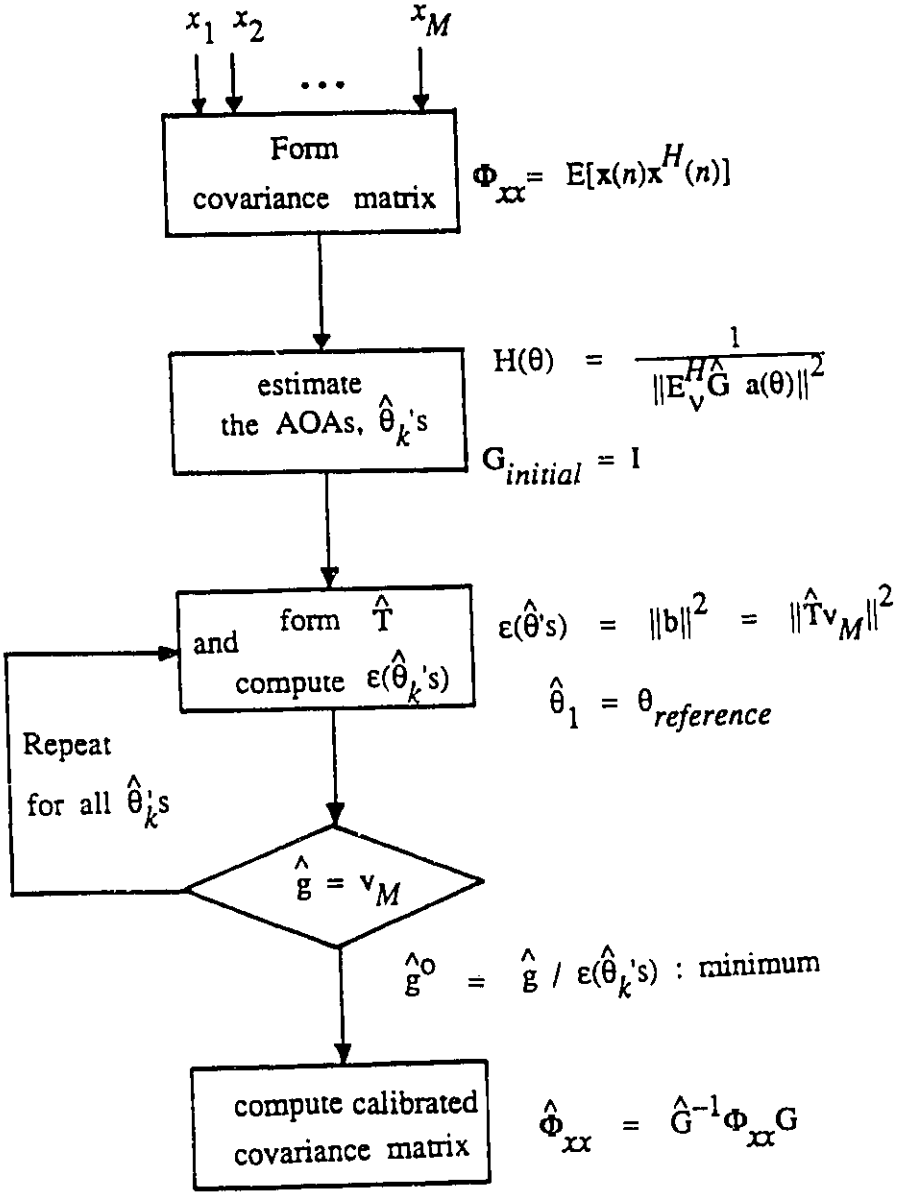


Fig. 5.1 Flow chart of the self-calibration algorithm

$$\epsilon(\hat{\theta}'s) = \|\mathbf{b}\|^2 = \|\hat{\mathbf{T}}\mathbf{v}_M\|^2 \tag{5.31}$$

where $\mathbf{v}_M$, which is the estimate of the gain and phase vector $\hat{\mathbf{g}}$, is the Mth right singular vector of matrix $\hat{\mathbf{T}}$ decomposed using the SVD.

4. Repeat Step 3 for all $\hat{\theta}_k$'s. The set of $\hat{\theta}_k$'s at which $\|\mathbf{b}\|^2$ is minimum will correspond to the optimal estimate of the AOAs, and so also the error vector $\hat{\mathbf{g}}^o$.

5. Finally, using Eqn. (5.22) to compute the calibrated covariance matrix, i.e.,

$$\hat{\Phi}_{xx} = \hat{\mathbf{G}}^{-1}\Phi_{xx}\mathbf{G}. \tag{5.32}$$

The range of the angle $\hat{\theta}_k$ at which the minimum norms $\epsilon(\hat{\theta}_k's)$ is computed is defined as follows

$$\hat{\theta}_{k_{max}} = \hat{\theta}_k + \Delta\theta_k/2 \tag{5.33a}$$

$$\hat{\theta}_{k_{min}} = \hat{\theta}_k - \Delta\theta_k/2 \tag{5.33b}$$

where

$$\Delta\theta_k = \hat{\theta}_{k_{max}} - \hat{\theta}_{k_{min}} \tag{5.33c}$$

is chosen as the minimum separation that can be resolved between two nearest sources using conventional estimation methods. The criterion that was proposed by Gabriel [21] can be employed here. That is, the minimum separation of the two signals, $\Delta\theta_k$, is defined as a function of the SNR, i.e. $\Delta\theta_k$ is smaller for high SNRs than for low SNRs.

5.4. Results

5.4.1. *Simulation data*

The effect of array calibration on the AOA estimates is shown in Fig. 5.2. In this result, the gain and phase errors at the mth element is denoted by

$$g_m = (1 + \delta_{am}) e^{j\delta_{pm}} \tag{5.34}$$

where δ_{am}, δ_{pm} respectively are the relative errors in amplitude and phase of the mth element. It should be noted that σ_{gain}, σ_{phase} in Fig. 5.2 are respectively defined by

$$\sigma_{gain} = |\delta_{am}|,$$

and

$$\sigma_{phase} = |\delta_{pm}|.$$

The antenna array used in the simulation is a 32–element array with two uncorrelated signals arrived at 0^o and 1^o with SNR_1 =SNR_2 = 30 dB. It can be observed that as the calibration errors increase the AOAs become more and more difficult to distinguish and/or tend to shift away from the original direction. This phenomenon is further illustrated in Figs. 5.3a to 5.3d, in which the eigenbeams, which correspond to the eigenvectors that are formed in the noise subspace, are increasingly degraded as the array errors increase. The effect is typified by the distortion in the nulls of the patterns: the nulls correspond to the signal AOAs. Fig. 5.4 gives a plot the minimum norm of the calibration vector as a function of the AOA estimates for the case $\sigma_{gain} = \sigma_{phase} = .04$ by using the self–calibration algorithm presented above. The reference signal is chosen to be at $\theta = 0^o$, and

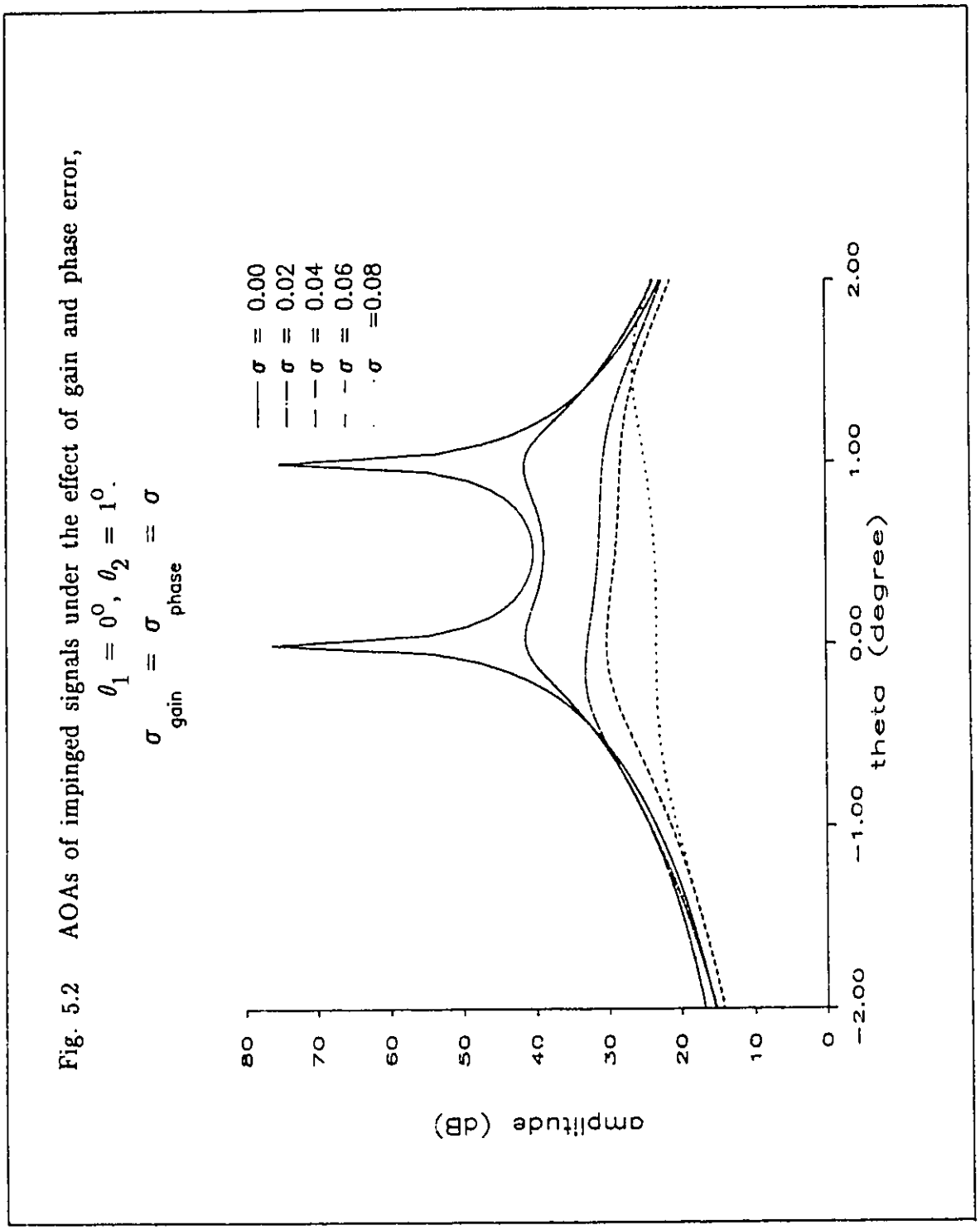


Fig. 5.2 AOAs of impinged signals under the effect of gain and phase error, $\theta_1 = 0^o$, $\theta_2 = 1^o$.

$\sigma_{gain} = \sigma_{phase} = \sigma$

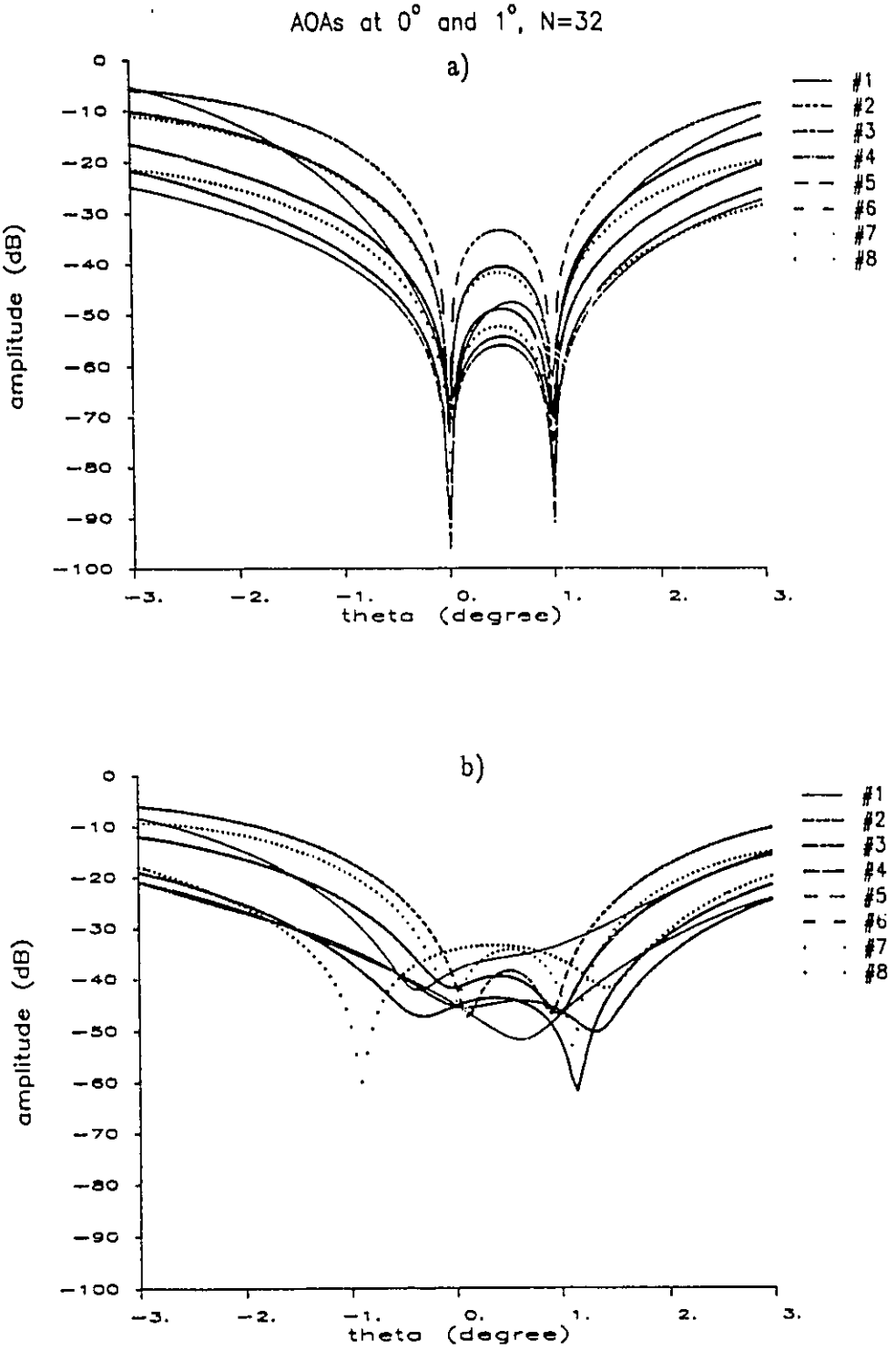


Fig. 5.3 Typical eigenbeam patterns of the noise subspace a) $\sigma_g = \sigma_p = 0.0$ b)$\sigma_g = \sigma_p = 0.02$,

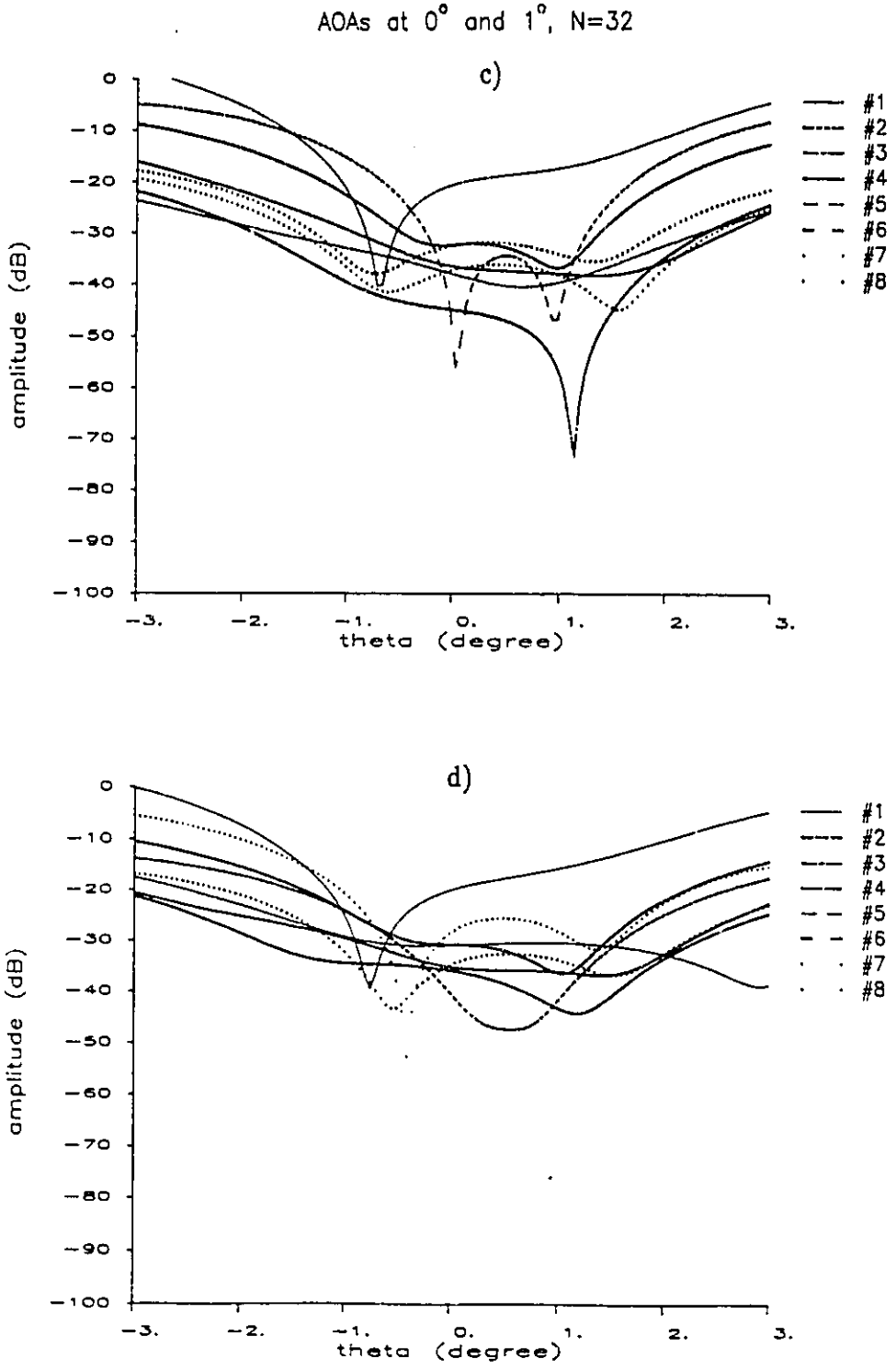


Fig. 5.3 Typical eigenbeam patterns of the noise subspace
c) $\sigma_g = \sigma_p = 0.04$ d) $\sigma_g = \sigma_p = 0.06$

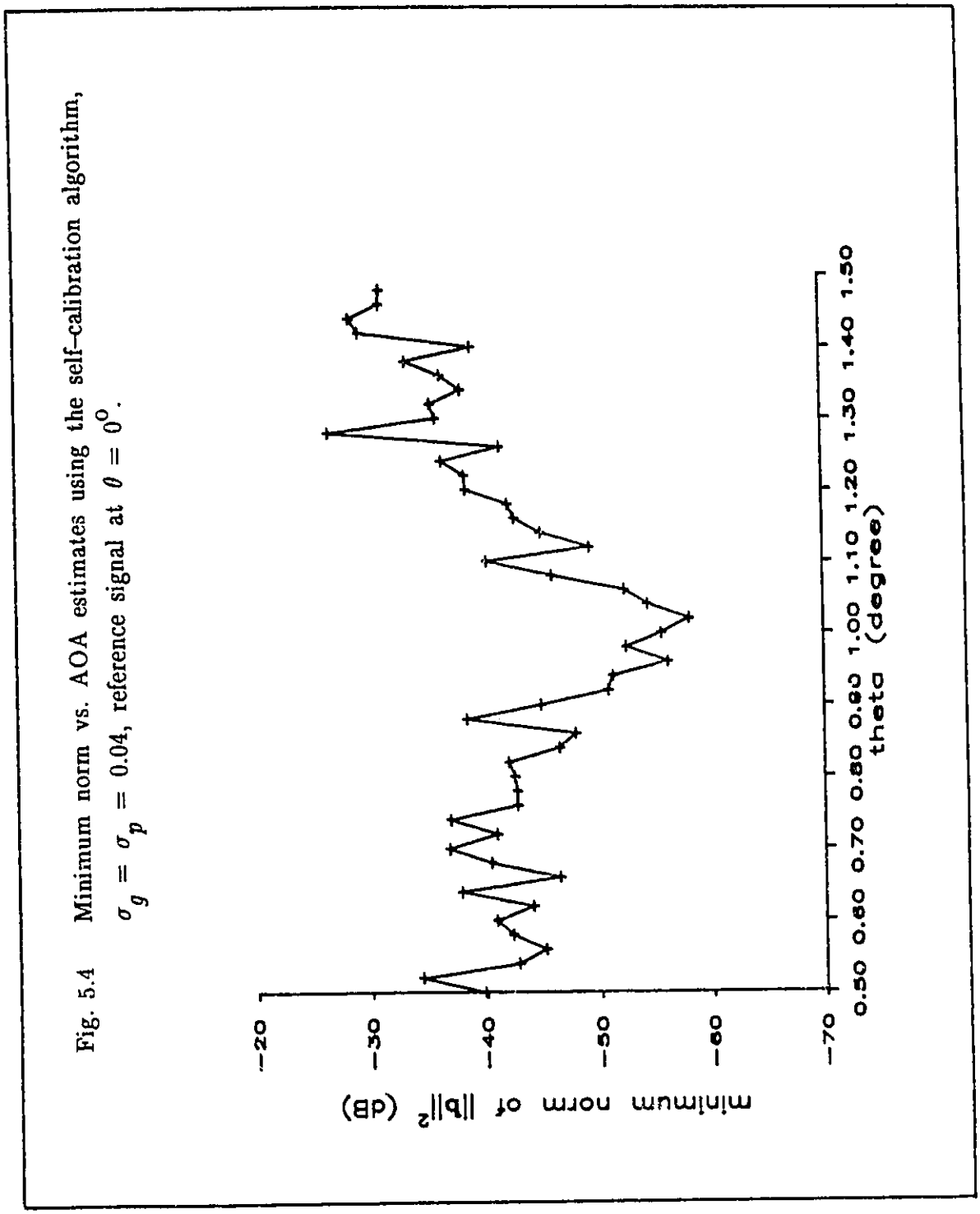


Fig. 5.4 Minimum norm vs. AOA estimates using the self-calibration algorithm, $\sigma_g = \sigma_p = 0.04$, reference signal at $\theta = 0^o$.

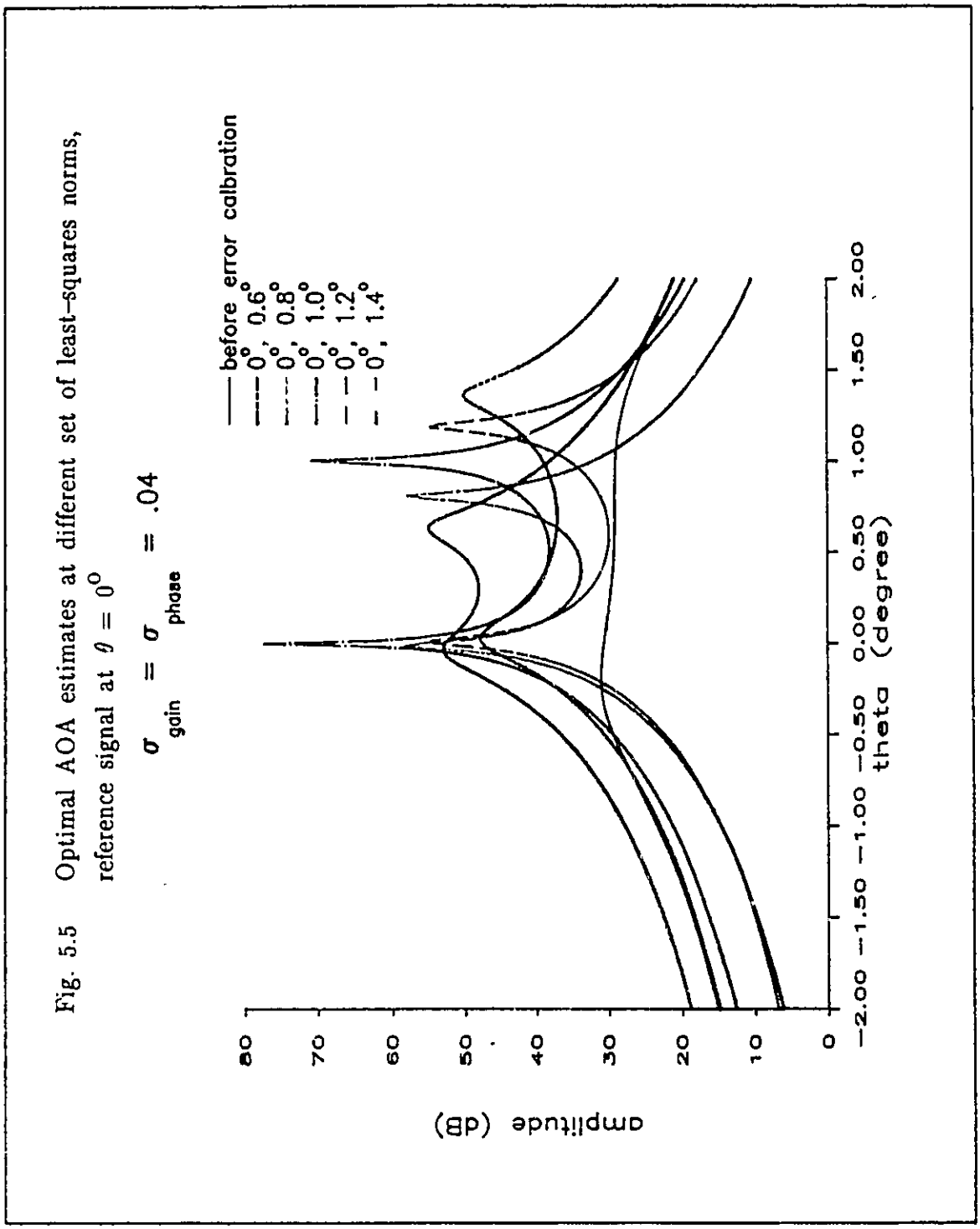


Fig. 5.5 Optimal AOA estimates at different set of least-squares norms, reference signal at $\theta = 0^{o}$

$$\sigma_{gain} = \sigma_{phase} = .04$$

$\Delta\theta_2 = .2$ beamwidth. It should be noted that in this example the gain error is very small but the phase error is about $\pm 2.3^{\circ}$, in terms of the required phase accuracy is relatively large. As it can be readily seen in Fig. 5.4, the minimum of the minimum least–squares norm is uniquely defined with a value for the AOA of the second signal of 1°. The optimal AOA value of the AOA is found to be 1.02°. Its relative error is given by

$$\frac{|1^{\circ} - 1.02^{\circ}|}{1^{\circ}} \times 100 = 2\%,$$

which is largely due to round–off errors during computation. Further results are illustrated in Fig. 5.5, in which the peaks are highest at the desired AOAs, i.e., $\theta_1 = 0^{\circ}$, $\theta_2 = 1^{\circ}$.

It should be noted that similar method was proposed by [32], however, the set of AOAs obtained is not an optimal one since the corresponding minimum norm (ϵ) is not the least minimum one. In fact, the AOAs are evaluated via the calibration vector g by computing the minimum least–squares norm of vector b where some of the AOAs are assumed to be known in advance. Meanwhile, the self–calibration technique presented here is capable of searching for the AOAs with the least minimum norm by using an iteration method. The algorithm is thus more robust due to the fact that the eigenstructure based method and the SVD are iteratively utilized to compute the minimum norm least–squares solutions for the AOAs, θ_k's, and the array calibration error vector g simultaneously.

5.3.2. *Measurement data*

In this section, we present results that were derived using measured data. These data was recorded on Lake Huron, Ontario using a measurement facility developed at the Communications Research Laboratory (CRL). The equipment used in the experiment is consisted of a C–band beacon transmitter and a receiving antenna consisting of a 32–element sampled aperture array. The transmitted frequency was 8.32 GHz, and the inter–element spacing for the array is 0.02 m, i.e., $d/\lambda = 1.585$. Details of both the measurement setup and the measurement site are presented in Appendix B. The setup is one in which multipath signals, i.e. surface reflected signals, are recorded along with the direct signal from the transmitter to the receiver. It is to be expected that the signals received at the antenna array are highly correlated. In order to decorrelate the multipath signals, the array is decomposed into subarrays of 27 elements and the formation of the array covariance matrix is carried out by means of the forward–backward spatial smoothing method [10], i.e.,

$$\Phi_{xx} = \frac{1}{2}(\Phi_{xx}^{f} + \Phi_{xx}^{b}) \tag{5.35}$$

where

$$\Phi_{xx}^{f} = \frac{1}{6}\sum_{i=1}^{6} \mathrm{E}[\mathbf{x}_i(n)\mathbf{x}_i^{H}(n)] \tag{5.36a}$$

and

$$\Phi_{xx}^{b} = \frac{1}{6}\sum_{j=1}^{6} \mathrm{E}[\mathbf{x}_j(n)\mathbf{x}_j^{H}(n)] \tag{5.36b}$$

in which

$$\mathbf{x}_i(n) = [x_i(n),\ x_{i+1}(n),\ \ldots,\ x_{i+26}(n)] \tag{5.36c}$$

$$\mathbf{x}_j(n) = [x_{33-j}(n),\ x_{32-j}(n),\ \ldots,\ x_{7-j}(n)] \tag{5.36d}$$

Table 5.1. Measurement parameters and conditions of the data set used in the analysis

Time:	12:14:53, 4–Nov–87
Temperature:	12^0
Wind:	westward, 16 km/hr
Precipitation condition:	drizzle
Pressure:	944 millibars
Average water elevation:	176.8 m above sea level
Sea surface condition:	1.5–m waves in height
Transmitter horn elevation:	192.37 m
Transmitter polarization:	horizontal
number of samples:	128

Table 5.2. Comparison of theoretical and measured AOAs

angle of arrival	theoretical	measured‡
θ_{dir}	$.107^o$	$.240^o$
θ_{ind}	$-.338^o$	$-.315^o$
$\Delta\theta$	$.445^o$	$.555^o$

‡ average values
$\frac{dN}{dh} = -79$

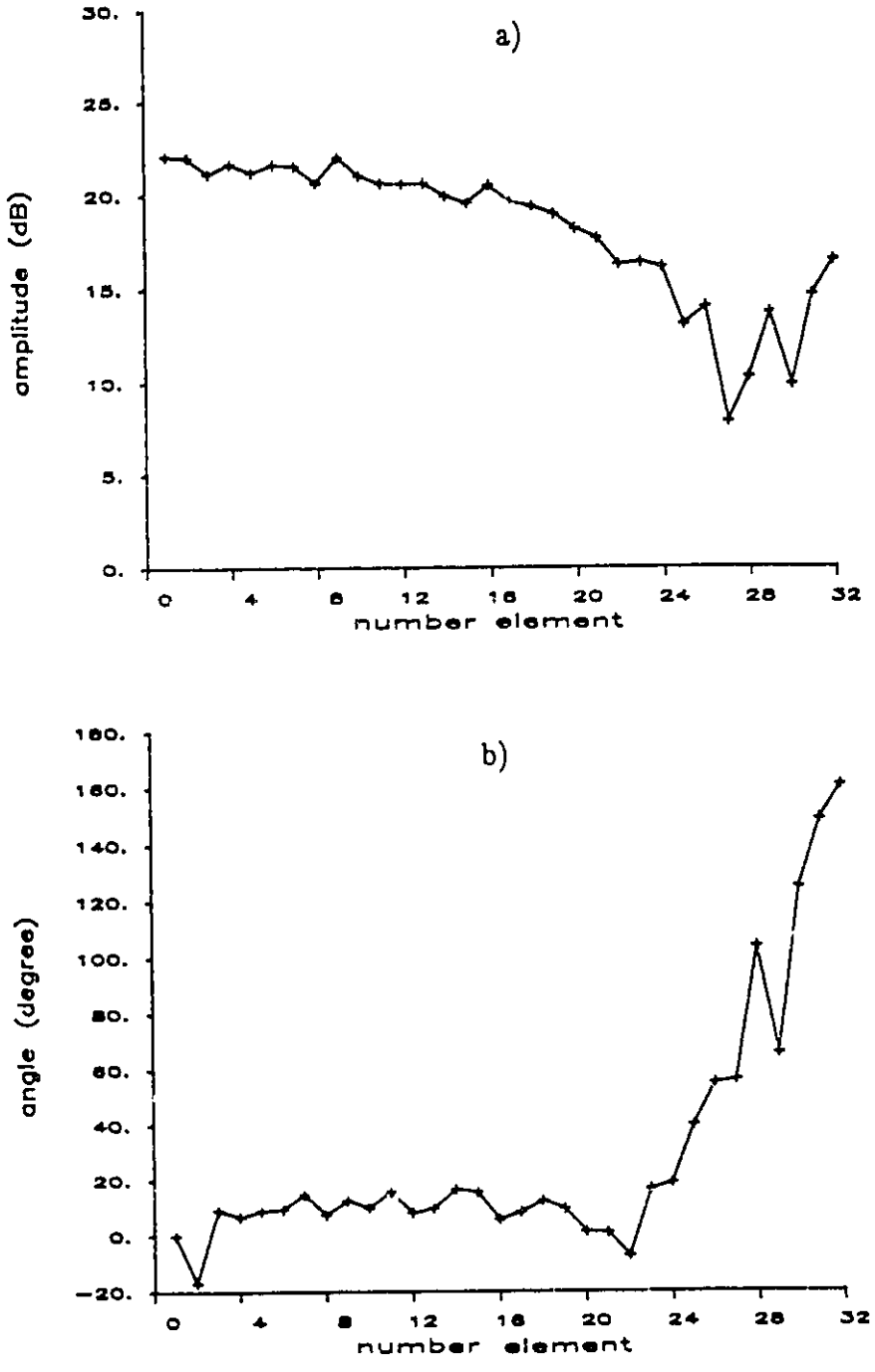


Fig. 5.6 Typical measurement of the 32–element antennas
a) amplitude b) phase

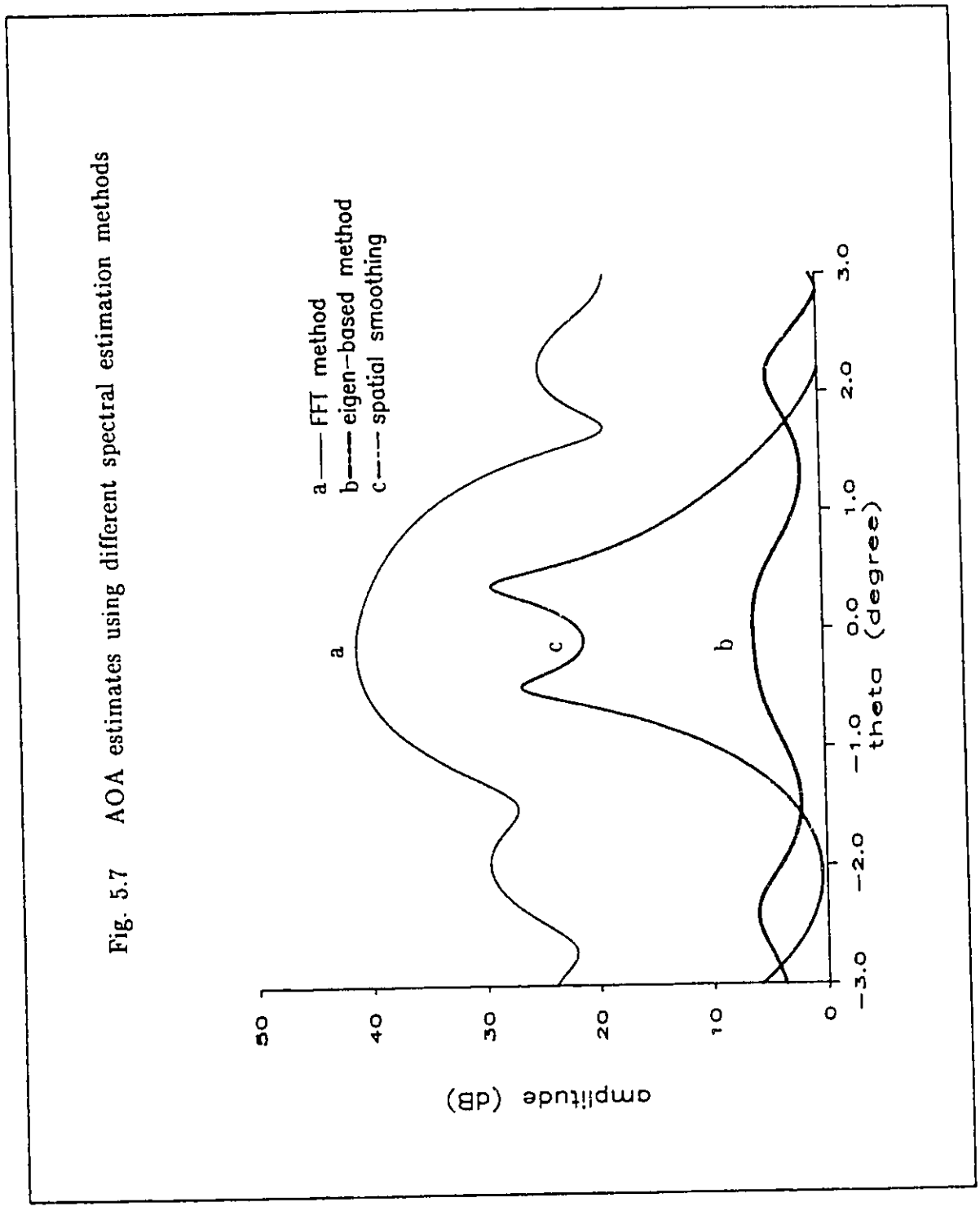


Fig. 5.7 AOA estimates using different spectral estimation methods

Typical measurements of the amplitude and phase aperture distributions of the antenna array are shown in Figs. 5.6a and 5.6b, respectively, with the measurement parameters and conditions given in Table 5.1. In this example, a test signal is located at $.107^{o}$ (i.e., θ_{dir}) from the boresight. Let us recall that the ratio d/λ is equal to 1.585. It can be seen that the array amplitude decreases from the left to right and then decreases near the end, whereas the phase increases monotonically (expected to be linearly increasing), slowly at first, then quickly increases when the amplitude becomes increasing rather than decreasing. These variations in amplitude and phase are due to errors in the array elements. The angles of arrival, which are estimated by using different methods, are shown in Fig. 5.7. As expected the FFT method failed to resolve the AOAs since the signal separation is less than one beamwidth. In fact

$$\Delta\theta \;=\; \theta_{di} - \theta_{ind} = .445^{o} \text{ (see Table 5.2)}$$

while the beamwidth

$$\theta_{3\text{dB}} = \frac{.88\lambda}{Md} \;\simeq\; 1.0^{o}.$$

As well, the eigen–based structure method (MUSIC) also fails to resolve the signals because of the coherent characteristics of the multipath signals. However, after applying spatial smoothing, the AOAs are well separated and clearly distinguished from each other. The measured signal directions, however, do not coincide with the true look directions due to the effect of calibration errors. The true and measured AOAs are given in Table 5.2. Note that the true values for the direct and indirect angles, denoted by θ_{dir} and θ_{ind}, respectively, were derived using the multipath model given in the Appendix B. To further illustrate the errors

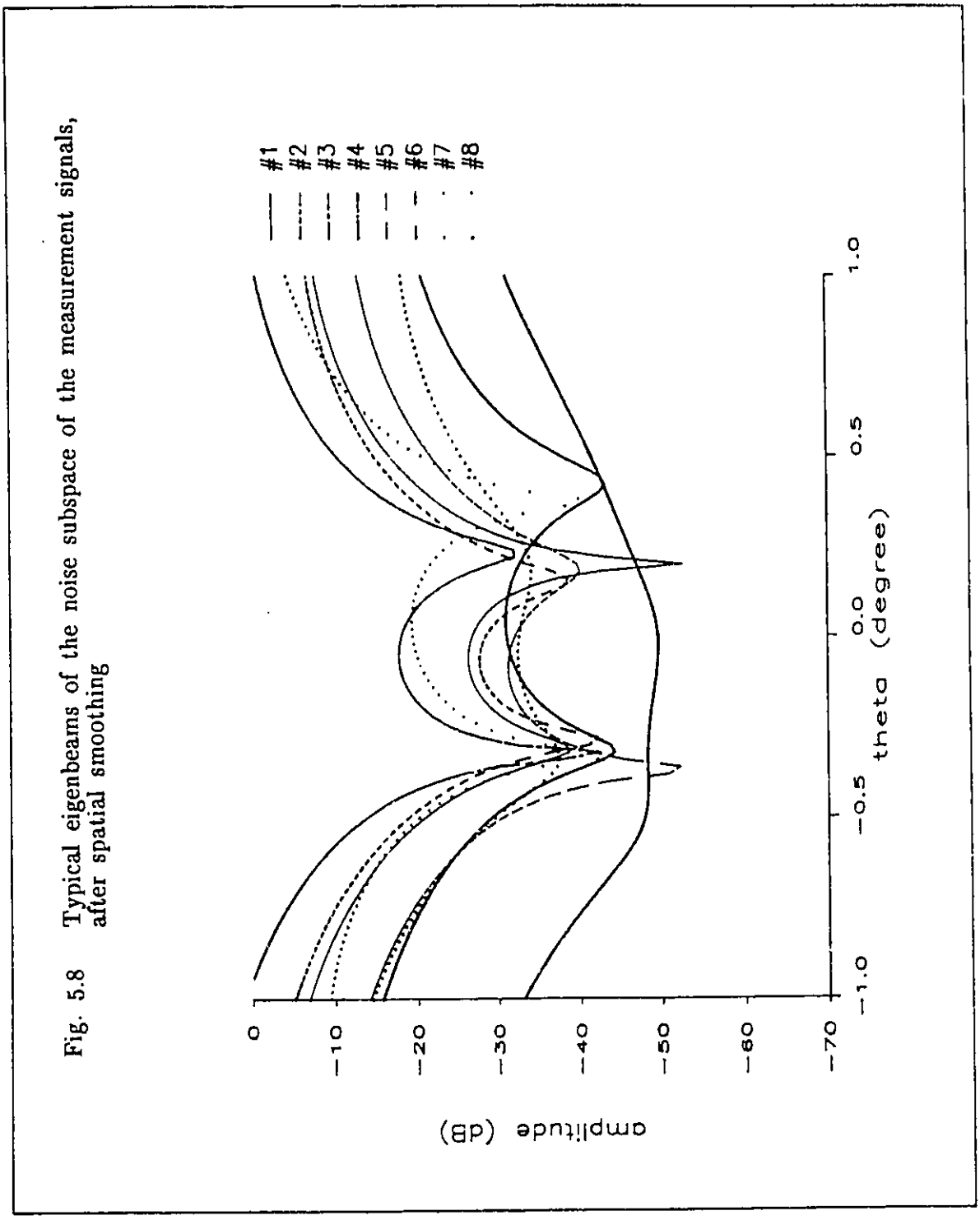


Fig. 5.8 Typical eigenbeams of the noise subspace of the measurement signals, after spatial smoothing

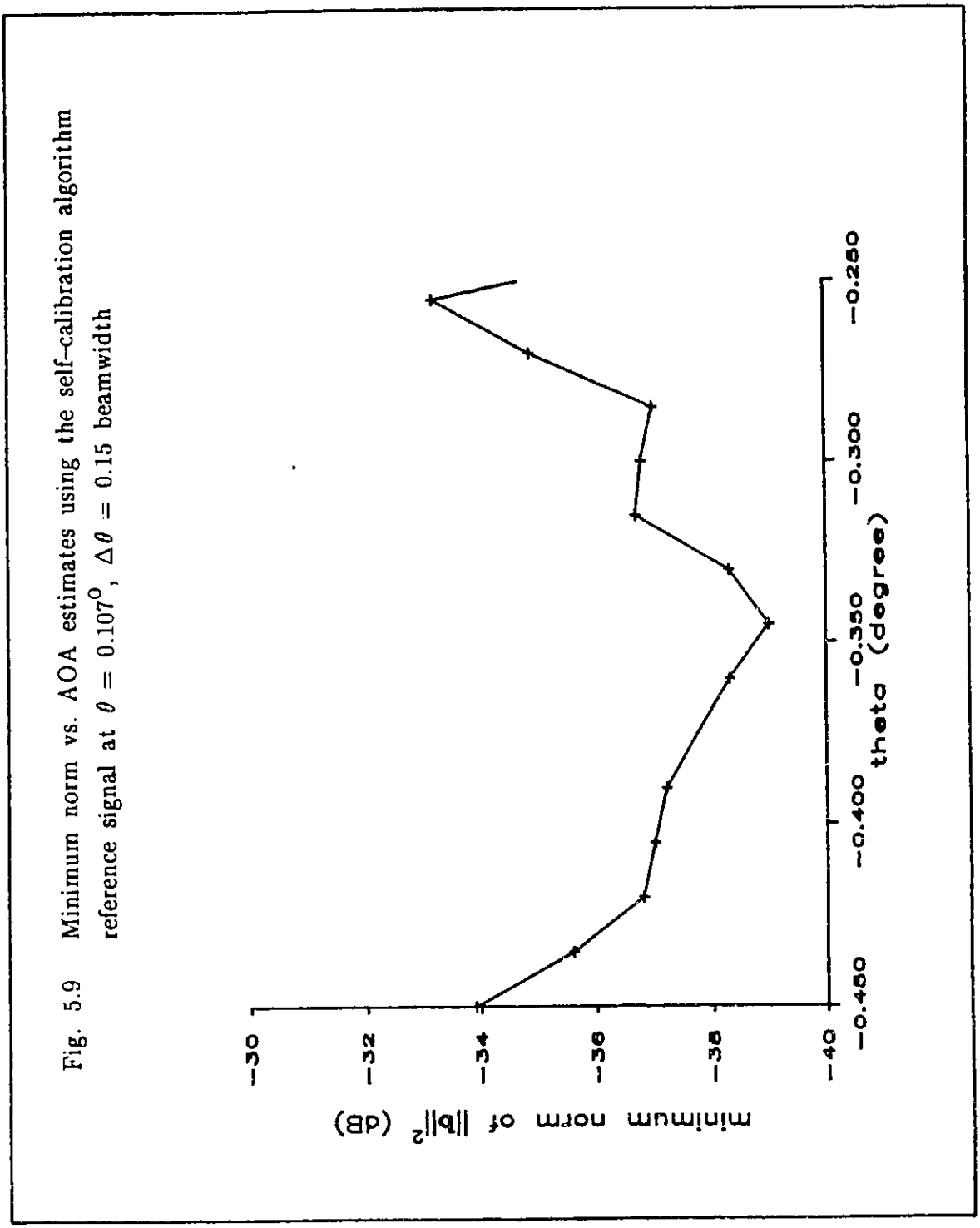


Fig. 5.9 Minimum norm vs. AOA estimates using the self-calibration algorithm reference signal at $\theta = 0.107^0$, $\Delta\theta = 0.15$ beamwidth

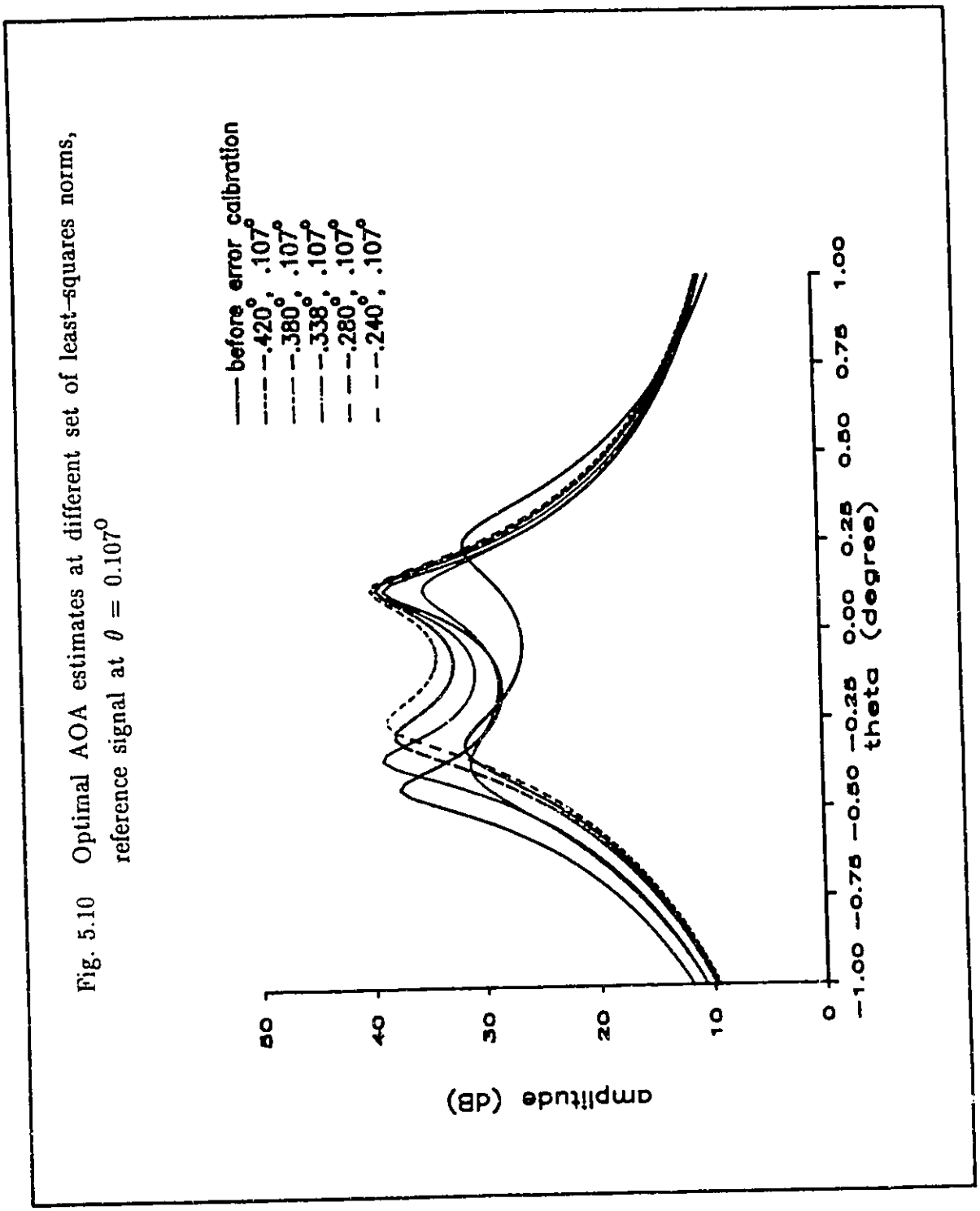


Fig. 5.10 Optimal AOA estimates at different set of least–squares norms, reference signal at $\theta = 0.107^{o}$

associated with the AOA measurement in Fig. 5.7, some typical eigenbeams of the corresponding eigenvectors, which is used to form the null subspace (after spatial smoothing), are shown in Fig. 5.8. It is can be seen via the eigenbeams that directional nulls (i.e signal directions) are degraded due to the calibration errors in both gain and phase of the array elements.

Figure 5.9 gives the results obtained by applying our self–calibration algorithm with the reference signal chosen as

$$\theta_{reference} \doteq \theta_{dir} = .107^{\circ}.$$

and the separation between the reference and auxiliary (indirect) signals

$$\Delta\theta_2 = .15 \text{ beamwidth.}$$

The curve of the minimum norm values, i.e. $C(\epsilon)$, is shown in Fig. 5.10. As it can be seen the highest peaks of the pattern occur at the directions of the optimal AOAs. It follows from Fig. 5.9 that the indirect signal is located uniquely, and that its optimal value, $\hat{\theta}^{\circ}_{ind}$, is found to be $-.348^{\circ}$. Its relative error is found from

$$\frac{|\theta_{ind} - \hat{\theta}^{\circ}_{ind}|}{|\theta_{ind}|} \times 100 = 3\%,$$

where θ_{ind} is the true value.

Adaptive Nulling Results

Now that we have successfully demonstrated AOA estimation using uncalibrated data, we can go on to implementing adaptive beamforming for nulling out unwanted signals and for maintaining the desired signal at a prescribed level. The adaptive process can be carried out by using the LS triarray processor given in Fig. 4.4. Figs. 5.11a and 5.11b show the results of adaptive nulling using the measurement data before and after applying the self–calibration algorithm. In these figures, a constrained vector $\mathbf{c}(\theta_d)$ corresponding to the angle $\theta_d = \theta_{dir} = .107^{\circ}$ has been employed. As it can be seen this results in a deep null at $\theta = .107^{\circ}$ in the retrodirective beam pattern, which leads to a constant gain at the desired look direction in the adapted pattern. It is worth noting that the null formed before applying the calibration algorithm (Fig. 5.11a) is located at an angle $\theta = -.315^{\circ}$, which is in variance with the AOA of the indirect signal. Also, this null is much shallower than that which is formed after utilizing the self–calibration algorithm (Fig. 5.11b).

5.5. Concluding Remarks

An self–calibration algorithm for antenna array systems has been presented. The development of the algorithm was based on the minimum norm least–squares estimate of the calibration vector **b**. The gain and phase error vector is computed via the SVD of the projection calibration matrix T. The AOAs of the test signal and an auxiliary signal are simultaneously computed via the spatial spectrum using the MUSIC algorithm. It has been shown using both simulated and measured data that the proposed algorithm is capable of estimating the optimal error vector and

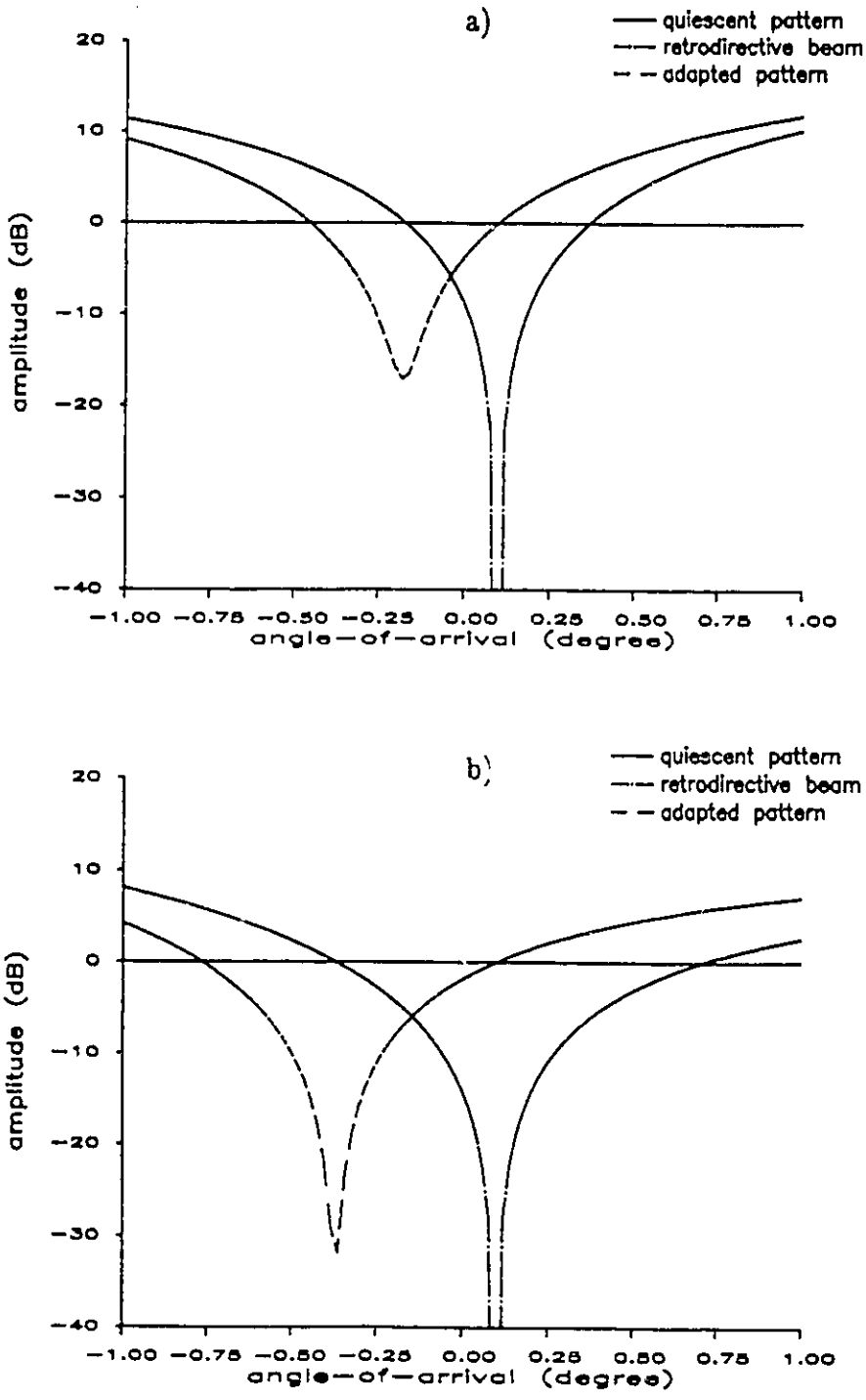


Fig. 5.11 Adapted patterns with a constraint at $\theta = 0.107^{o}$.
a) before calibration b) after calibration

the optimal AOAs with minimum mismatch. The self–calibration algorithm was demonstrated to be robust. It is expected that this algorithm will lead to the development of practical self–calibration subsystems in the not too distant future.

An outstanding problem associated with the self–calibration algorithm presented above is the validation range of the AOA ($\Delta\theta$), which is used in the computation of the minimum least–squares norm. Under normal conditions, for example, as was the case for the measured data, the AOA estimates can be found by using an eigen–based technique and the validation range can be estimated. However, as the calibration errors become critical, for example, in the case of simulation data with $\sigma_{gain} = \sigma_{phase} = .06$, or more (Fig. 5.1), it is impossible to estimate the validation range of the AOAs, therefore the calibration will be more difficult to carry out. In fact, this difficulty suggests that there exists a limitation in the range of errors that the self–calibration process will be able to successfully cope with, i.e. there is an upper bound beyond which the technique fails. It is believed that this bound varies as a function of the signal separation and the signal–to–noise ratio.

CHAPTER 6

CONCLUSIONS

6.1. Conclusions and Discussion

The objective outlined in the beginning of the thesis has been achieved. Novel techniques and architectures for adaptive beamforming, which are typified by the 2–D adaptive beamforming algorithms and the self–calibration algorithm for antenna arrays, have been derived and presented.

Two–dimensional Adaptive Beamforming

Two–dimensional adaptive beamforming algorithms, both the classical and modern methods, have been developed. The derivations that were used in presenting these algorithms followed closely from the theory of 1–D beamforming algorithms. In large measure the 2–D algorithms are natural extensions of corresponding 1–D algorithms. The 2–D LMS and 2–D Applebaum algorithms have been shown to be cost–effective, i.e. they are bounded. In particular, their discrete forms are within reach of current digital signal processors. The relationship between the 2–D and 1–D adaptive algorithms was derived. It has been shown that by ordering the 2–D array into a lexicographic format the two algorithms are mathematically equivalent to each other. The nulling capabilities of the 2–D adaptive algorithms were further illustrated by the introduction of 2–D eigenbeams.

It has been shown that a 2–D retrodirective beam pattern can be formed by two independent 1–D eigenbeams, which respectively, are formed along a row and a column of the 2–D array.

It is believed that the concept of 2–D eigenbeams was first presented in this thesis, and also the true forms of the 2–D adaptive beamforming algorithms. Most of work in the past has treated the 2–D problem as an extension of the 1–D cases. Although partially adaptive beamforming with large arrays was proposed by Chapman [19] the adaptation was carried out along the rows of the 2–D array and results are combined along the columns or vice versa. This approach is thus not robust since interference nulling applies along a plane containing the interferer instead of locally nulled process as provided by the 2–D methods.

The degree of freedom for a 2–D array is usually greater than that for a 1–D array, simply because a 2–D array usually has more antenna elements. There are up to $(LM–J)$ degrees of freedom associated with a 2–D array, where J denotes the number of desired directional beams. That is, a 2–D antenna array equipped with an adaptive processor is capable of nulling out up to $(LM–J)$ jammers. The inherent versatility of 2–D arrays, which arises from the large number of independent antenna elements, provides much promise for the future development of surveillance radar systems. Indeed, 2–D adaptive antennas provide such systems with the capability to conduct both track–while–scan and adaptive beamforming operations at the same time.

A 2–D adaptive beamforming algorithm based on the QRD–LS algorithm was also presented. It was shown that this algorithm could be implemented by using triarrays for performing the least–squares minimization. The adaptive beamformer

architecture is arranged in the form of a 3–D systolic array in which the PEs are highly pipelined and interconnected with their nearest neighbours. With the proper data flow in time–skewed format the adaptation in both the rows and columns of the 2–D array can be processed simultaneously and concurrently. The use of the 3–D systolic array for implementing 2–D adaptive beamformer has resulted in a significant improvement in memory storage and overall system time delay. It has been shown that the computation is improved approximately by a factor of L over the conventional method, with the assumption that $L \leq M$, where L and M are the dimensions of the planar array. In fact, it requires $2(L \times M)$ delay cycles to initialize the triarray systolic adaptive beamforming processor for a planar array, whereas only $4M$ delay cycles are required by the 3–D systolic array implementation. The tradeoff however is in the complexity of the implementation. In particular, a larger number of PE cells is required, which means that more memory and interconnections are involved. As a consequence, this will reduce the performance of the adaptive processor.

The computational requirements for a fully adaptive 2–D array are very demanding and therefore are expensive to implement in large arrays. In the past, adaptive beamforming has been implemented using closed–loop algorithms and analog processors. Because of their rates of convergence, closed–loop algorithms are much less attractive than they have been in the past due to the rapid changes that can take place in present day signal environments. This suggests that in the future open–loop algorithms and digital processing will become increasingly prevalent. In order to reduce the computational overhead for digital adaptive beamforming, in the near term, the dimension of 2–D arrays will be reduced by means of subarraying. Each subarray is [illegible] that its beam points in the wanted direction. The adaptive beamforming is then carried out on the outputs of

subarrays using multiple parallel digital processors. At the present time, systolic arrays, which are massively parallel and pipelined processors, show considerable promise for being the vehicle which will provide radar systems with a real–time signal processing capability.

Self–Calibration Algorithm for Antenna Arrays

Adaptive beamforming process cannot be done satisfactorily without the use of a proper calibration of the signals received at antenna elements. In this thesis, calibration errors in both gain and phase of an antenna array have been systematically modelled. As well, a self–calibration algorithm has been developed. The algorithm is an iterative one, whose derivation is based on the subspace technique and the least–squares norm minimization. It has been shown with the use of a reference signal, which is assumed to be known in advance, that the self–calibration technique is capable of locating the optimal AOA for an auxiliary signal and correcting the calibration errors with minimum mismatch. The algorithm has been successfully demonstrated using both simulated and measured data. In particular, in the case of measured data, the auxiliary signal was a multipath signal and highly correlated with the test signal. The interference–nulling capability of an adaptive beamformer has been shown to be significantly improved with the incorporation of the self–calibration technique.

Although, several workers have claimed to solve the self–calibration problem [32,34] either by using an algorithm to compute the recursive least–squares norm minimization of the calibration vector or by assuming that some of the signal sources are known in advance, it is believed that their solutions are ambiguous, non–unique, and/or not optimal. The iterative scheme presented in this thesis has

solved the problem of uncertainty and non–uniqueness associated with the calibration errors. As well, with the technique presented here optimal solutions for the signal AOAs can be obtained. The technique has been shown to be robust and superior to the other methods presented in the literature.

The problems associated with the self–calibration algorithm are the estimate of initial AOAs and the choice of a validation range. It may be readily observed that the level of calibration errors may degrade the initial estimate of the AOAs. As a consequence, this may further become a critical matter in the choice of a validation range of the AOA, which is used in the computation of the minimum least–squares norm. It has been shown that self calibration of an antenna array can be achieved as long as an estimate of the AOAs can be made and the signal separations are well defined.

6.2. Recommendations

The following recommendations are made with regard to future work. It is believed that there are still a great number of improvements to be made to bring the described algorithms a step closer to a practical realization and to further improve the performance of the algorithms and methods presented.

Two–Dimensional Adaptive Beamforming

1) The performance of the 2–D adaptive beamforming algorithms should be investigated further from an analytical point of view; in particular, it is still necessary to derive the SINR expression for the beamforming output. The

performance of the 2–D LMS and 2–D Applebaum algorithms as the signal environment is varied should be extensively studied. As well, implementation of 2–D adaptive beamforming with linear constraints requires further work. Intuitively, it is thought that linearly constrained adaptive beamforming with planar arrays can be conveniently incorporated into the 3–D systolic array structure, which was used to implement 2–D adaptive beamforming using the QRD–LS algorithm.

2) The effect of mutual coupling effect on the performance of 2–D adaptive beamforming algorithms should be investigated, especially when the antenna main beam is steered away from boresight. In this instance, the sidelobe levels will increase, which in turn may degrade the SINR, to negate some of the improvement achieved by the beamforming algorithm. By correcting mismatch errors due to mutual coupling and applying weighting such as the Chebyshev weighting, it may be possible to reduce the sidelobe levels, thereby improving the adaptive antenna's performance.

3) An experimental antenna system should be developed to test the performance of the 2–D adaptive beamforming algorithms in practical situations. The self–calibration algorithm could be employed to correct any gain and phase errors. Ultimately a 2–D adaptive beamforming processor should be developed and tested with a planar antenna array.

Self–Calibration Algorithm for Antenna Arrays

4) Even though the algorithms presented here work well for the case of two signals or under multipath scenarios, the requirement for uniqueness in the

AOA estimates is not thoroughly defined. It is not clear that if the characteristics of the calibration problem is solely represented by the local minimum of the least–squares norm. The exact bound of the AOAs, which must be used in the searching procedure so that a unique solution can be obtained, remains undefined. Further investigations should be carried out.

5) Self–calibration with more than two signals should be investigated. Tests using both simulation and experimental data should be carried out. The iterative optimization process is expected to be more complex as the number of signals increases, however, it is worthwhile to investigate the computational aspects of the method.

Recently, a Warp computer, a linear systolic machine, became part of the CRL research facilities. A systolic adaptive beamformer test–bed model using the Warp has been proposed [53]. Further investigations into the following are also recommended

6) A systolic adaptive beamformer test–bed using the Warp processor should be developed. To evaluate a real–time adaptive beamforming, it is desirable that the Warp computer will be used incorporating a 32–element C–band antenna system which is being developed at the Communications Research Laboratory. The success of this system relies heavily on proper data acquisition and transfer between the antenna system and the host of the Warp computer. Further work needs to be carried out in developing the hardware interface between the antenna and the Warp.

7) The self–calibration algorithm should be implemented on the host of the

Warp computer for calibrating the antenna data prior to processing by the Warp.

8) Finally, the 3–D systolic array architecture for 2–D adaptive beamforming should be implemented using the Warp computer. As well, implementation complexity and the time required for processing the antenna data using the Warp computer should be compared with the time required by the VAX computer.

REFERENCES

[1] R.A. Monzingo and T.W. Miller, *Introduction to Adaptive Arrays*, Willey–Interscience, 1980.

[2] J.E. Hudson, *Adaptive Array Principles*, Peter Peregrinus, UK 1981.

[3] B. Widrow and S.D. Stearns, *Adaptive Signal Processing*, Prentice Hall, 1985.

[4] R.T. Compton, Jr., *Adaptive Antennas: Concepts and Performance*, Prentice Hall, 1988.

[5] S. Haykin, Ed., *Array Signal Processing*. Prentice Hall, 1985.

[6] C.R. Ward *et al.*, "Application of systolic arrays to adaptive beamforming", *IEE Proceedings*, Vol. 1311, Pt. F, No. 6, Oct. 1984.

[7] C.R. Ward *et al.*, "A novel algorithm and architecture for adaptive digital beamforming", *IEEE Trans. Antennas Propagat.*, Vol. AP–34, No. 3, March 1986.

[8] J.V. McCanny and J.G. McWhirter, "Some systolic array developments in United Kingdom", *IEEE Computer*, Vol. 20, No. 7, July 1987.

[9] D.J. Chapman and S.P. Applebaum, "Adaptive arrays with mainbeam constraints", *IEEE Trans. Antennas Propagat.*, Vol. AP–24, Sept 1976.

[10] W.F. Gabriel, "Using spectral estimation techniques in adaptive processing antenna systems", NRL Report 8920, October, 1985.

[11] T.J. Shan and T. Kailath, "Adaptive beamforming for coherent signals and interferences", *IEEE Trans. Acoust., Speech, Signal Processing*, Vol. ASSP–33, No. 3, June 1985.

[12] Y. Bresler *et al.*, "Optimum adaptive beamforming for coherent signals and interferences", *IEEE Trans. Acoust., Speech, Signal Processing*, Vol. ASSP–36,

No. 6, June 1988.

[13] V.U. Reddy *et al.*, "Performance analysis of the optimum beamformer in the presence of correlated sources and its behaviour under spatial smoothing", *IEEE Trans. Acoust., Speech, Signal Processing*, Vol. ASSP–35, No. 7, July 1987.

[14] S.C. Pei *et al.*, "Modified spatial smoothing for coherent jammer suppression without signal cancellation", *IEEE Trans. Acoust., Speech, Signal Processing*, Vol. ASSP–36, No. 3, March 1988.

[15] A.P. Howells, "Intermediate frequency side–lobe canceller", *U.S. Patent* 3202 990, Aug. 24, 1965.

[16] S.P. Applebaum, "Adaptive arrays", *Syracuse Univ. Res. Corp.*, SPL TR 66–1, Aug. 1966.

[17] B. Widrow *et al.*, "Adaptive antennas systems", *Proc. IEEE*, Vol. 55, pp. 2143–2159, Dec. 1967.

[18] I.S. Reed, *et al.*, "Rapid convergence rate in adaptive arrays", *IEEE Trans. Aerosp. Electron. Syst.*, Vol. AES–10, pp. 853–863, Nov. 1974.

[19] D.J. Chapman, "Partial adaptivity for large array", *Trans. IEEE Antennas Propagat.*, Vol. AP–24, pp. 685–695, Sept. 1976.

[20] E. Brookner and J.M. Howell, "Adaptive–adaptive array processing", *Proc. IEEE*, pp. 602–607, 1986.

[21] W.F. Gabriel, "Spectral analysis and adaptive array superresolution techniques", *Proc. IEEE*, Vol. 68, pp. 654–666, June 1980.

[22] J. G. McWhirter, "Recursive least–squares minimization using a systolic array", *Proc. SPIE*, p. 431, Real–Time Signal Processing VI, 1983.

[23] S.T. Alexander and S.A. Rajala, "Image compression results using the LMS adaptive algorithm", *IEEE Trans. Acoust., Speech, Signal Processing*, Vol. ASSP–33, June 1985.

[24] Y.J. Hong, *et al.*, "The effect of a finite–distance signal source on a far–field steering Applebaum array – two dimensional array case", *IEEE Trans. Antennas Propagat.*, Vol. AP–36, No. 4, April 1988.

[25] T.V. Ho and J. Litva, "Systolic array for 2–D adaptive beamforming", *Int'l Conf. on Systolic Arrays*, May 1988.

[26] M.M. Hadhoud and D.W. Thomas, "The two–dimensional adaptive LMS algorithm", *IEEE Trans. Circuits Syst.*, Vol. CAS–35, No. 5, May 1988.

[27] H. Cox, "Resolving power and sensitivity to mismatch of optimum array processors", *J. Acoust. Soc. Am.*, Vol. 54(3), pp. 771–785, 1973.

[28] P.M. Schultheiss and J.P. Ianniello, "Optimal range and bearing estimation with randomly perturbed arrays", *J. Acoust. Soc. Am.*, Vol. 68(1), pp. 167–173, 1980.

[29] A.H. Quazi, "Array beam response in the presence of amplitude and phase fluctuations", *J. Acoust. Soc. Am.*, Vol. 72(1), pp. 171–180, 1982.

[30] C.L. Zahm, "Effects of errors in direction of incidence on the performance of an adaptive array", *Proc, IEEE*, Vol. 60(8), pp. 1008–1009.

[31] A. Pualraj and T. Kailath, "Direction of arrival estimation by eigenstructure methods with unknown gain and phase", *Proc. IEEE ICASSP'85*, pp. 17.7.1–17.7.4.

[32] A. Pualraj, *et al.* "A subspace approach to determining sensor gain and phase with applications to array processing", *Proc. SPIE*, Vol. 696, pp. 102–107, 1986.

[33] A.J. Weiss, *et al.* "Eigenstructure approach for array processing with unknown intensity coefficients", *IEEE Trans. Acoust. Speech, Signal Processing*, Vol. ASSP–36, pp. 1613–1617, Oct. 1988.

[34] B. Friedlander and A.J. Weiss, "Eigenstructure methods for direction finding with sensor gain and phase uncertainties", *Proc. IEEE ICASSP'88*, pp.

2681–2684.

[35] W.F. Gabriel, "Adaptive arrays – an introduction", *Proc. IEEE*, Vol. 64, No. 2, Feb. 1976.

[36] L.J. Griffiths, "A simple adaptive algorithm for real–time processing in antennas arrays", *Proc. IEEE*, Vol. 57, pp. 1696–1704, Oct. 1969.

[37] O.L. Frost III, "An algorithm for linearly constrained adaptive signal processing", *Proc. IEEE*, Vol. 60, pp.926–935, Aug. 1972.

[38] R.T. Compton, Jr., et al., "Improved feedback loop for adaptive arrays", *IEEE Trans. Aerosp. Electron. Syst.*, Vol. AES–16, No. 2, March 1980.

[39] J. Capon, "Probability distributions for estimators of the frequency wavenumber spectrum", *Proc. IEEE*, Vol. 58, pp. 1785–1786, 1970.

[40] D.M. Boroson, "Sample size considerations for adaptive arrays", *IEEE Trans. Aerosp. Electron. Syst.*, Vol. AES–16, pp. 842–845, 1980.

[41] H. Steyskal and J.F. Rose, "Digital beamforming for radar systems", *Microwave Journal*, Jan. 1989.

[42] Special Issue on Adaptive Arrays, *IEEE Trans. Antennas Propagat.*, Vol. AP–24, Sept. 1976.

[43] Special Issue on Adaptive Processing Antenna Systems, *IEEE Trans. Antennas Propagat.*, Vol. AP–34, March 1986.

[44] R.T. Compton, "An adaptive array in a spread–spectrum communication system", *Proc. IEEE*, Vol. 66, No. 3, MArch 1978

[45] J.H. Winters, "Spread spectrum in a four–phase communication system employing adaptive antennas", *IEEE Trans. Commun.*, Vol. COM–30, No. 5, May 1982.

[46] M.P. Ekstrom, "Realizable Wiener filtering in two–dimensions", *IEEE Trans. Acoust., Speech, Signal Processing*, Vol. ASSP–30, pp. 31–40, Feb. 1982.

[47] G.H. Golub, "Numerical methods for solving linear least–squares problems",

Num. Math., No. 7, pp. 206–216, 1965.

[48] H.T. Kung and W.M. Gentleman, "Matrix triangularization by systolic arrays", *Proc. SPIE*, p. 298, Real–Time Signal Processing IV, 1981.

[49] W. Givens, "Computation of plane unitary rotations transforming a general matrix to triangular form", *J. Soc. Ind. Appl. Math.*, No. 6, pp. 26–50, 1958.

[50] W.M. Gentleman, "Least–squares computations by Givens transformations without square–roots", *J. Ins. Math. Appl.*, Vol. 12, pp. 329–336, 1974.

[51] J.G. McWhirter and T.J. Shepherd, "An efficient systolic array for MVDR beamforming", *Intn'l Conf. on Systolic Arrays*, May 1988.

[52] R.O. Schmidt, "A signal subspace approach to multiple source location and spectral estimation", Ph.D dissertation, Stanford University.

[53] T.V. Ho, M.K. Law, and J. Litva, "Application of the Warp processor in adaptive beamforming", *Intn'l Conf. on Systolic Arrays*, May 1989.

[54] T.J. Shepherd and J.G. McWhirter, "A pineline array for linearly constrained least squares optimization", *Proc. IMA Conf. on Mathematics and Signal Processing*, Academic Press, Sept. 1985.

APPENDIX A

BASIC TERMINOLOGY AND CONCEPTS

A.1. Basic Theory of Array Antennas

Linear Arrays

Consider the linear array in Fig. A.1, consisting of M elements with inter–element spacing d. For convenience the array is treated as a receiving antenna but because of the reciprocity principle, the results obtained apply equally if the array is considered to be a transmitting antenna. The elements are assumed to be isotropic point sources. The radiation pattern in the plane of the array in direction θ denoted by $E_a(\theta)$ is given by

$$E_a(\theta) = E_1 + E_2 e^{-j\varphi} + \ldots + E_M e^{-j(M-1)\varphi}$$

$$= \sum_{m=1}^{M} E_m e^{-j(m-1)\varphi} \qquad (A.1)$$

where φ is the phase difference of signals in adjacent elements

$$\varphi = \frac{2\pi d}{\lambda} (\sin\theta - \sin\theta_0) \qquad (A.2)$$

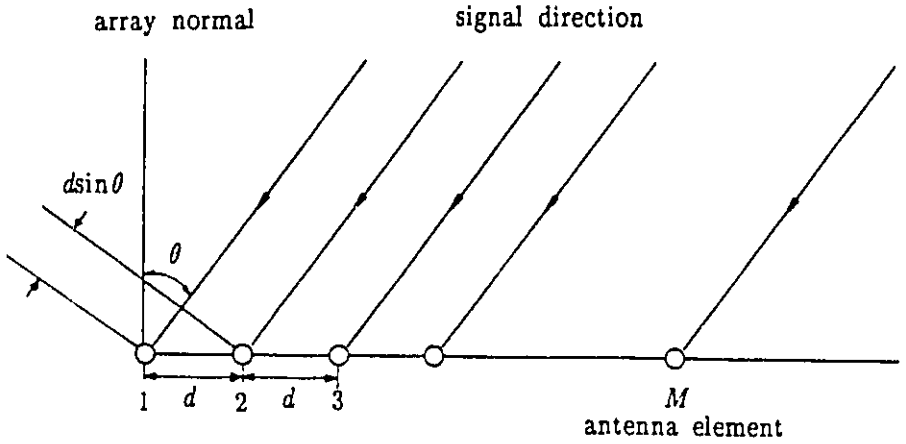


Fig. A.1 Configuration of a linear antenna array

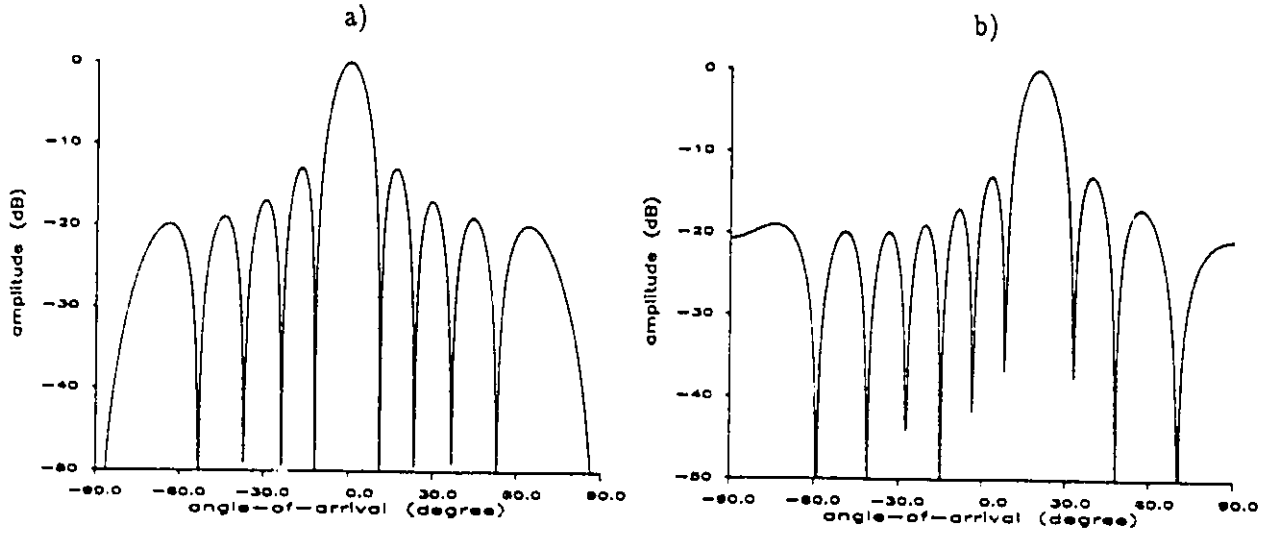


Fig. A.2 Beam pattern of a 10-element antenna array
a) main beam at 0^{o} b) main beam at 20^{o}.

where θ_0 is the main antenna beam direction with respect to boresight (Fig. A.2).

For uniformly excited arrays with all signals in phase, the E_m are identical, then

$$E_a(\theta) = E_0 \frac{\sin\frac{M}{2}\varphi}{M\sin\frac{\varphi}{2}} \tag{A.3}$$

When the individual elements are not isotropic (omnidirectional) but have a radiation pattern $E_e(\theta)$ themselves, this can be included to give the overall array pattern

$$E(\theta) = E_a(\theta)E_e(\theta) \tag{A.4}$$

$E_a(\theta)$ is known as *array factor*, and $E_e(\theta)$ the *element factor*. In many cases, such as those of dipoles, slots, and waveguide apertures, $E_e(\theta)$ may be approximately by a simple $\sqrt{\cos\theta}$ function [A1].

The grating lobes for a one–dimensional array are located at

$$\sin\theta_g - \sin\theta_0 = \pm n\frac{\lambda}{d}, \quad n = 0, 1, 2, \ldots \tag{A.5}$$

The lobe at $n = 0$ is the main beam. The array will have only one single major lobe, and grating–lobe maxima will not occur at values of $-90^\circ < \theta \leq 90^\circ$ as long as $\lambda/d < 1/2$. When the scanning is limited, the value of λ/d may be increased. The maximum scanning angle, θ_{max} [A2], is related to d and λ by

$$d < \lambda/(1 + |\sin\theta_{max}|) \tag{A.6}$$

The key parameters of interest in the array pattern are the width of the main beam and the sidelobe levels. For the uniformly excited array (see Eqn. (A.3)), the half-power beamwidth θ_B is given by,

$$\theta_B = 0.886\lambda/D \text{ or } \theta_B^0 \approx 51\lambda/D \tag{A.7}$$

where $D = (M-1)d$ is the length of the array, and the first side lobe level is at −13.2 dB relative to the main beam [A2]. It can be seen that (see Eqns. (A.3) and (A.7)) as the number of elements increases the main lobe beamwidth decreases, and the number of sidelobes and pattern nulls increases.

Planar Arrays

Consider a rectangular planar array (Fig. A.3) consisting of $L \times M$ elements. The radiation pattern is given by

$$E_a(\theta,\phi) = \sum_{l=1}^{L} \sum_{m=1}^{M} E_{lm} e^{-j\{(l-1)\varphi_x + (m-1)\varphi_y\}} \tag{A.8}$$

where the phase differences φ_x, φ_y are

$$\varphi_x = \frac{2\pi d_x}{\lambda} (\sin\theta\cos\phi - \sin\theta_0\cos\theta_0) \tag{A.9a}$$

$$\varphi_y = \frac{2\pi d_y}{\lambda} (\sin\theta\sin\phi - \sin\theta_0\sin\theta_0) \tag{A.9b}$$

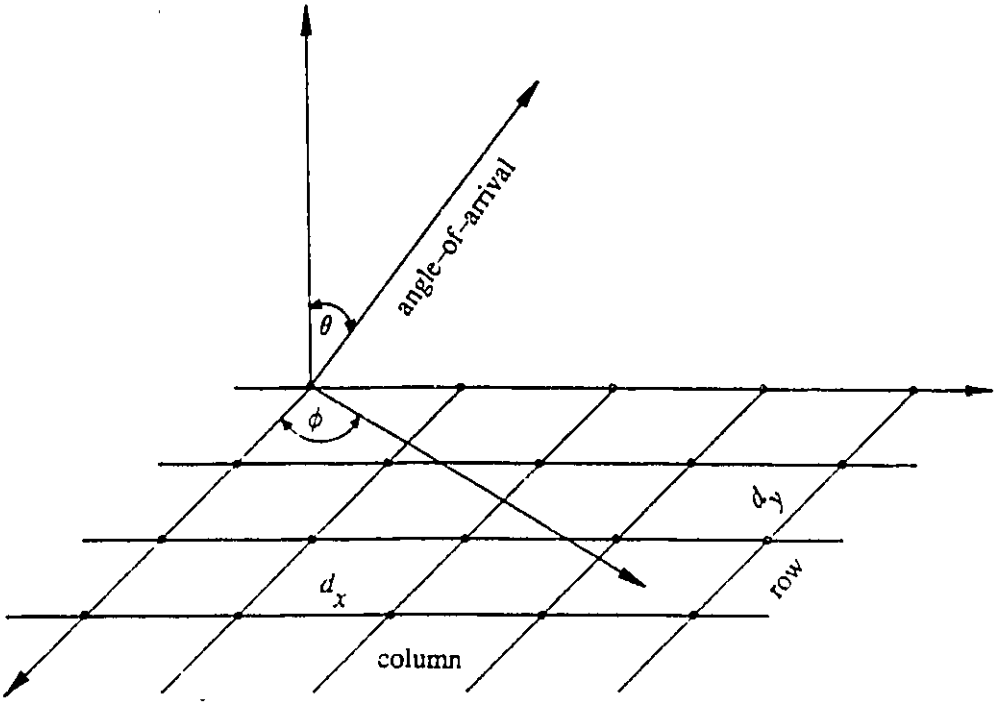


Fig. A.3 Configuration of a planar antenna array

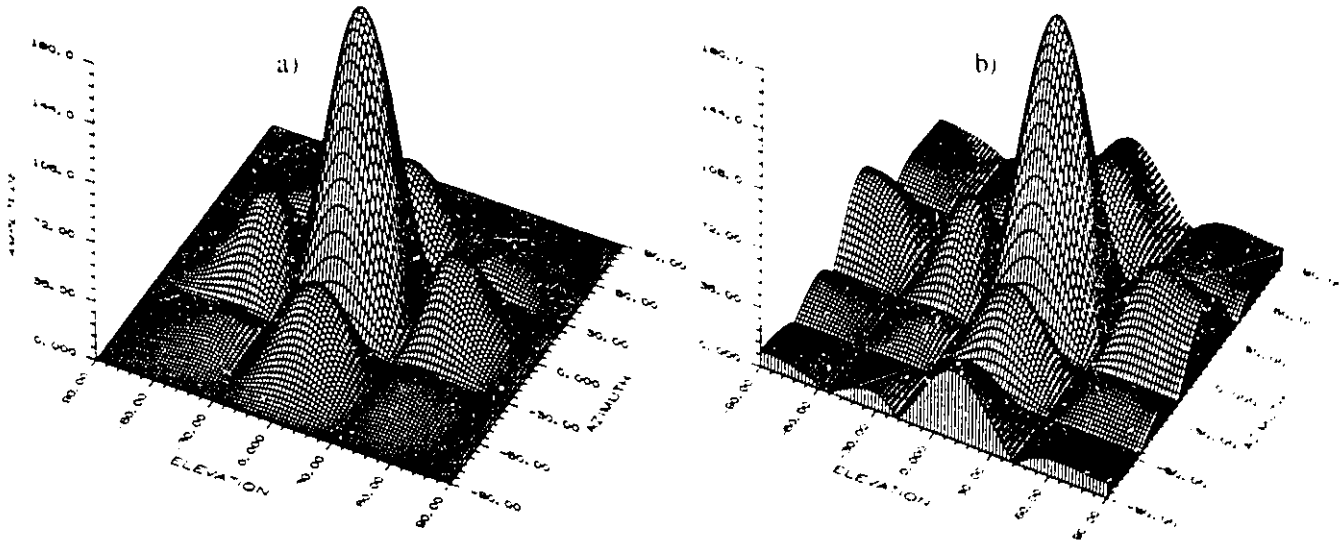


Fig. A.4 Beam pattern of a 4×4 antenna array

a) main beam at 0^{O} b) main beam at $(10^{O},10^{O})$

and d_x, d_y are the element spacings along the x and y directions respectively, and θ_0, ϕ_0 , are the beam pointing angles. It is assumed d_x and d_y are constant, i.e. the spacing along the x-axis and the y-axis are uniform.

The radiation pattern for a $L \times M$ rectangular array of radiators may be written as a product of the radiation pattern in the two planes which contains the principle axes of the antennas, i.e.,

$$\begin{aligned} E_a(\theta,\phi) &= E_a(\varphi_x,\varphi_y) \\ &= E_a(\varphi_x)E_a(\varphi_y) \end{aligned} \tag{A.10}$$

Then, for uniformly excited arrays

$$E_a(\theta,\phi) = \frac{E_{0x}E_{0y}}{LM} \frac{\sin\frac{L}{2}\varphi_x}{\sin\frac{1}{2}\varphi_x} \frac{\sin\frac{M}{2}\varphi_y}{\sin\frac{1}{2}\varphi_y} \tag{A.11}$$

Consequently, the constraint on the element spacings d_x and d_y is the same as that for linear arrays in order to avoid grating lobes. Also, it is clear that the beamwidth, sidelobe levels, and main beam directions for a planar array can be analyses in the same manner employed for linear array.

As with linear arrays, the row and column array factors $E_x(\theta,\phi)$ and $E_y(\theta,\phi)$ of the rectangular array can each be arbitrarily steered by introduction of the appropriate phase shift sequences for the row and column elements (Fig. A.4).

In addition to the rectangular arrays, there are circular arrays and elliptical

arrays, which are useful when angular symmetry is desired in a 2–D operation.

A.2. VLSI Array Processors

Real–time array signal processing requires the use of parallel processors because of the speed offered by computing machines [A3]. Current general–purpose computers suffer severe system overhead, and therefore are not suitable for real–time signal processing. Recently, low–cost, fast VLSI devices have become available. Their use in the future design of massively parallel processors suggests major improvement in the speeds of signal processors. In fact, at present one solution is to use special–purpose array processors, and to maximize the processing concurrency by means of either pineline processing and/or parallel processing.

Two special–purpose VLSI array architectures, which have been extensively discussed in the literature, are the *systolic* and the *wavefront array*. The concurrency in these arrays is derived from pineline processing and/or parallel processing. These types of processing are illustrated in Fig. A.5.

Systolic Array

A systolic system is a network of processors which rhythmically compute and pass data through the system [A4]. In a systolic computing system, every processor regularly pumps data in and out at a rate which is controlled by a global clock. During each cycle, computations are carried out at each processor so that a regular flow of data is maintained within the network. The data

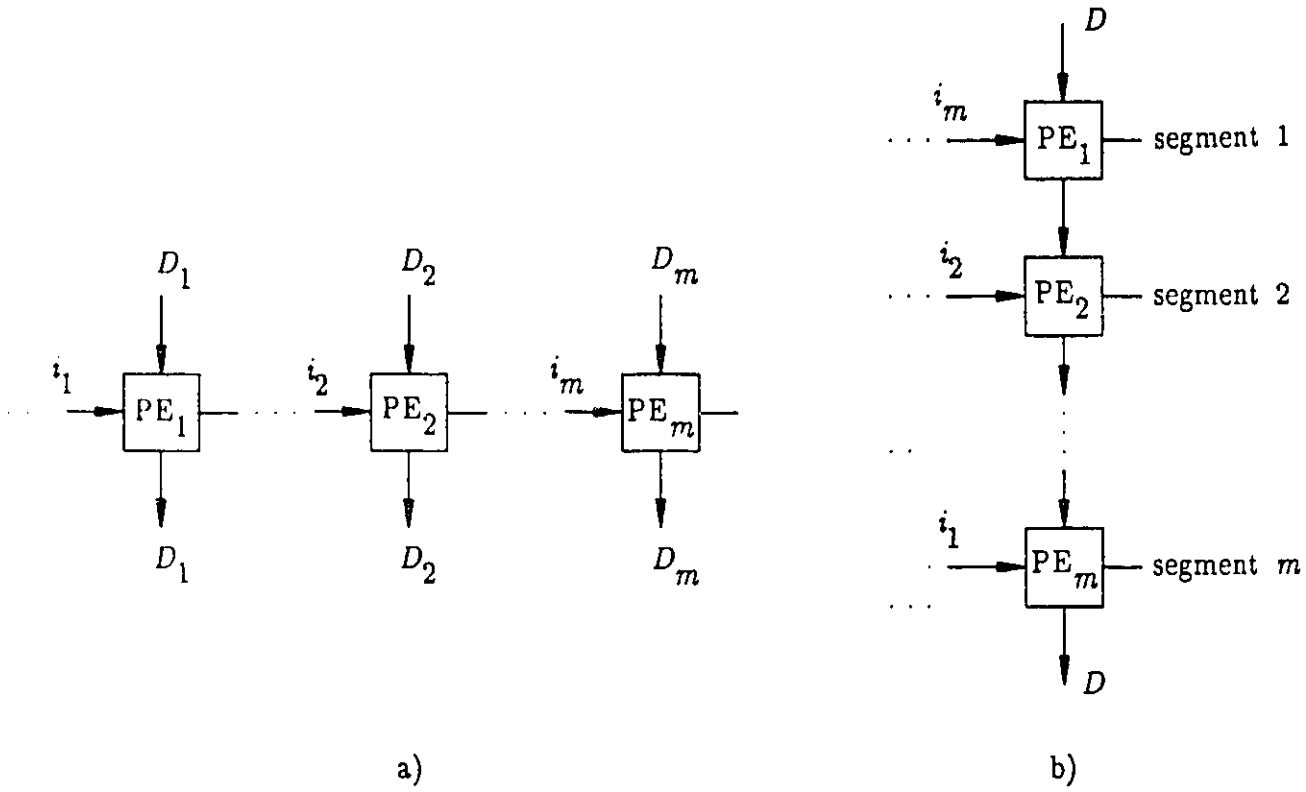


Fig. A.5 Array processor architectures
a) parallel processor b) pineline processor

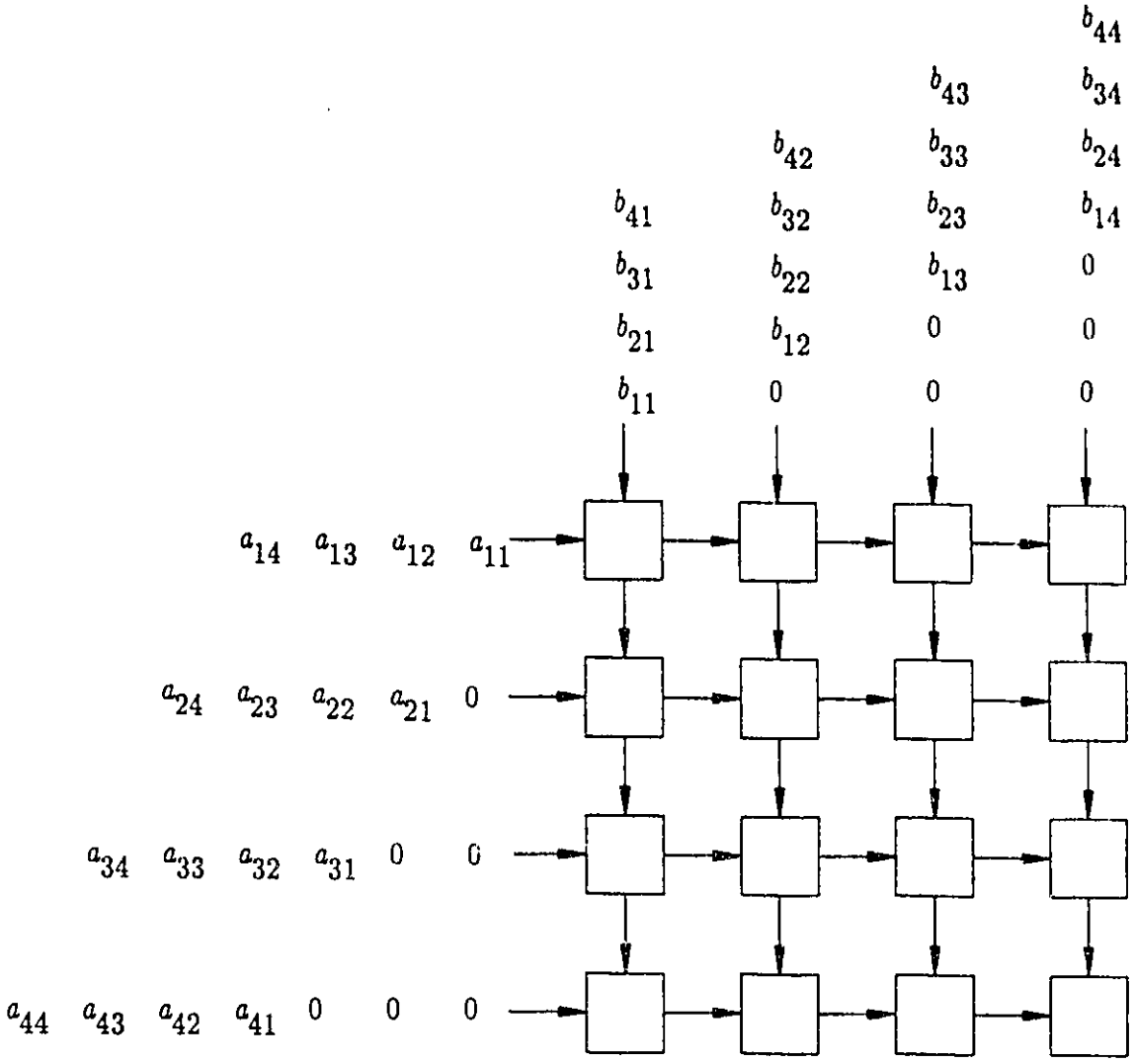


Fig. A.6 A systolic array of matrix multiplication. The figure shows a systolic array consisting of a 4×4 array of PEs (represented by black boxes). All the PEs uniformly consume and produce data during each time unit. The input data, matrices A and B, are pre–arranged in an ordered sequence. The C data (C = A×B) stay temporary within the PEs and will be pumped out from one side of the array. Due to the systolic nature of the array the time of arrival of the input data is adjusted by introducing delays. This is why the zeros are shown in the figure.

movements in a systolic array are described in terms of snapshots of the activities of the processors and the interconnecting lines.

The systolic array is especially suited for implementing those algorithms [A5], which can be structured to take advantage of the parallelism offered by the systolic array. The processing elements (PEs) in a systolic array can be locally connected to perform digital filtering, matrix multiplication, and other related operations. An example of a systolic array structure for matrix multiplication is shown in Fig. A.6. The input data are pre–arranged in an orderly sequence and enter the computation in time skew format because of the systolic nature of the structure. Also, it is to be noted that the data reside temporarily within the PEs and will be pumped out from one side of the array.

The physical realization of systolic arrays is closely tied to VLSI technology. The affinity of VLSI comes about because of the important properties of modularity, regularity, local interconnection, as well as high degree of pinelining and highly synchronized multiprocessing. It is also architecturally scalable, i.e., the size of the array may be indefinitely extended as long as the system synchronization can be maintained. There is extensive literature on the subject of systolic arrays and their applications. The interested reader is referred to [A6,A7].

Wavefront Array

The limitations of systolic arrays are closely tied to the fact that their activities are controlled by a global clock. From a hardware perspective, this global synchronization is of primary concern due to its effect to the clock skew, fault tolerance, and peak power of an array processor [A8]. A simple solution to

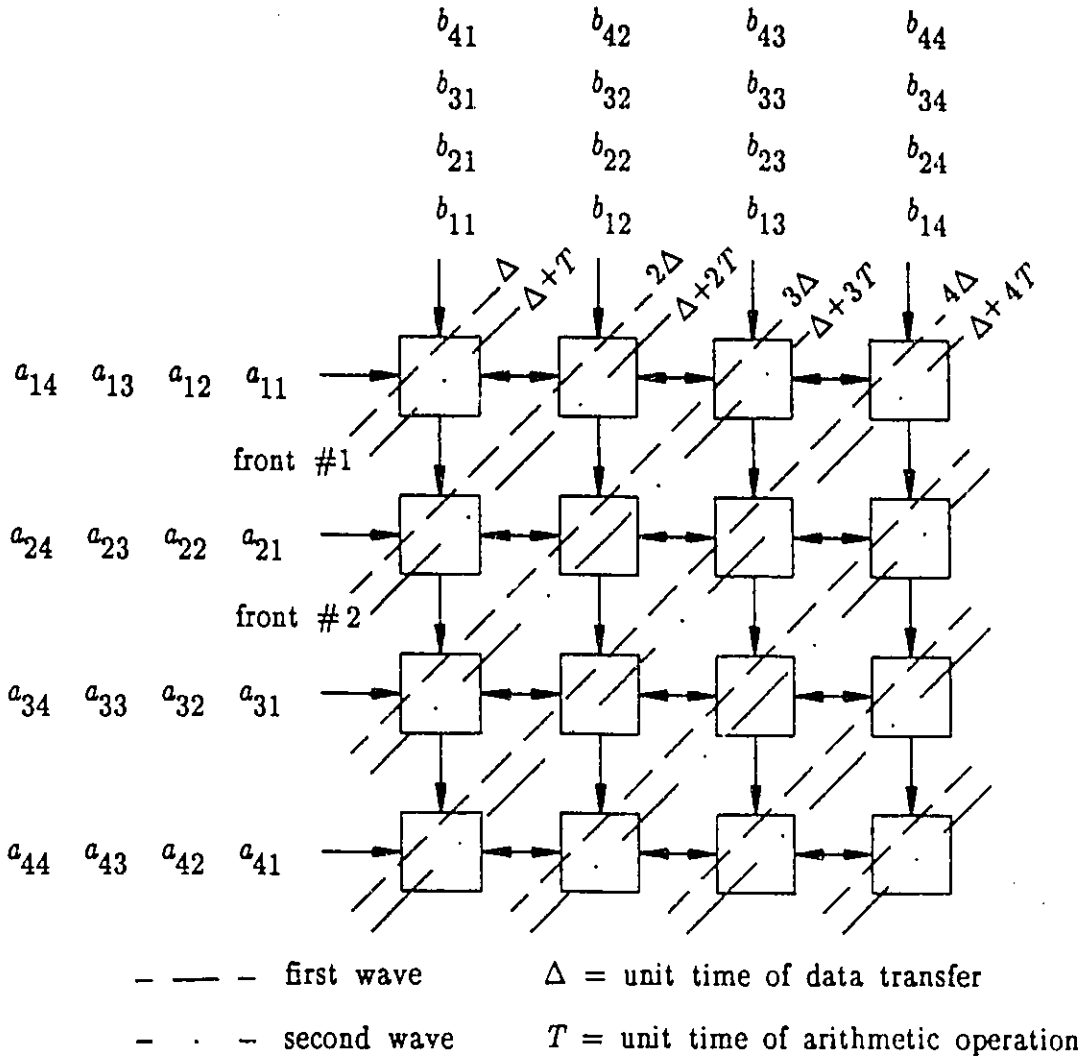


Fig. A.7 A wavefront for carrying out matrix multiplications. In this figure, the computing network serves as a (data) wave propagation medium. In this case, the time interval (T) between separate wavefronts is equal to the time needed for the arithmetic operations: multiple–and–add. The speed of propagation of the wavefront is determined by the time interval Δ, which is equal to the data transfer time. In general, the major advantages of the wavefront arrays over systolic arrays are seif–timing and data driven computation, which means that the global clock can be dispensed with.

these problems is to adopt the principle of data flow computing in array processors. The array architecture is so called the wavefront array. It takes advantages of control–flow locality in addition to the data–flow locality inherently possessed by most algorithms.

The execution of instructions in a wavefront array processor is dependent only only upon the availability of operands and resources required. Unrelated instructions can be executed concurrently without interference. Conceptually, this permits a data–driven, self–timing approach to array processing which substitutes the requirement of correct "timing" by correct "sequencing" [A8]. An example of matrix multiplication using a wavefront array is illustrated in Fig. A.7. The data matrices are stored in the memory modules to the left (in columns) and on the top (in rows). The process starts with the processing element at the far left corner. The computational activities then propagate to the nearest neighbors, thereby creating a computational wavefront which travels down the processor array. The computational wavefront is similar to an electromagnetic wavefront, since each processor acts as a secondary source and contributes to the propagation of the wavefront.

The principle advantages of wavefront array processors are simple representation of concurrent activity, relative independence of PEs, greater use of pinelining, and reduced use of global control and memory. Since there is no need for synchronizing the entire wavefront array is truly architecturally scalable. In fact, it may be stated that a wavefront array is a systolic array in combination with the data flow principle.

References

[A1] J.R. Forrest, "Phased Arrays", *Modern Radar Techniques*, Scanlan 1988.

[A2] M.I. Skolnik, *Radar Handbook*, McGraw–Hill, Chapter 11, 1970.

[A3] S.Y. Kung, "VLSI array processors", *IEEE ASSP Magazine*, Vol. 2, No. 3, July 1985.

[A4] H.T. Kung and C.E. Leiserson, "Systolic arrays for VLSI", *Sparse Matrix Symposium*, SIAM, pp. 256–282, 1978.

[A5] H.T.Kung, "Why systolic architectures", *IEEE Computer*, Vol. 15, No. 1, 1982.

[A6] A.L. Fisher and H.T. Kung "Special–purpose VLSI architectures: general discussions and a case study", *VLSI and Modern Signal Processing*, S.Y. Kung, H.J. Whitehouse, and T.K. Kailath (eds.), 1985.

[A7] S.Y. Kung, "On supercomputing with systolic/wavefront array processors", *Proc. IEEE*, Vol. 72, No. 7, July 1984.

[A8] S.Y.Kung *et al.*, "Wavefront array processor: language, architecture, and applications", *IEEE Trans. on Computer*, C–31, No. 11, pp. 1054–1066, Nov. 1982.

APPENDIX B

EXPERIMENTAL SETUP AND MULTIPATH PHENOMENON

B.1. Experimental Setup

A set of experiments were carried out to collect data with a sampled aperture antenna. One of the objective was to evaluate the performance of adaptive beamforming algorithm. The experimental setup is shown in Fig. B.1. It used a multi-parameter aperture radar system (MARS), a bistatic radar, which was developed at the Communications Research Laboratory, McMaster University [B1] (Fig. B.2). The MARS consists of a transmitter, a 32-element sampled aperture antenna and a data acquisition network. The specifications for the system are summarized in Table B.1.

The measured data that were presented in the thesis were collected during a field trip in late October 1987, on the west coast of Bruce Peninsula, close to Tobermory, Ontario, at a site overlooking Lake Huron (Fig. B.3). The distance between the receiver and the transmitter was greater that the far field region for the MARS array, which is defined by

$$r_f = \frac{2D^2}{\lambda}$$

Table B.1. Specifications of the measurement system used in the Lake Huron experiments.

Transmitter

- 100 mW CW sources feeding two 22 dB horn antennas
- simultaneous dual frequencies
- horizontal and vertical polarization
- adjustable transmitter height

Antenna Array

- 32–element linear array
- 1.82–meter aperture, .05715–meter inter–element spacing
- 10 dB H–polarized horns
- matched ± .1 mm machine tolerance
- multi–frequencies; 8.0 to 12.4 GHz with 30 MHz steps.

Receivers

- coherent modulation, with frequency stability to 10^{-12}
- accuracy of .1 dB and 1^0
- .1 Hz Doppler resolution
- 1 Hz to 2 KHz sampling rate

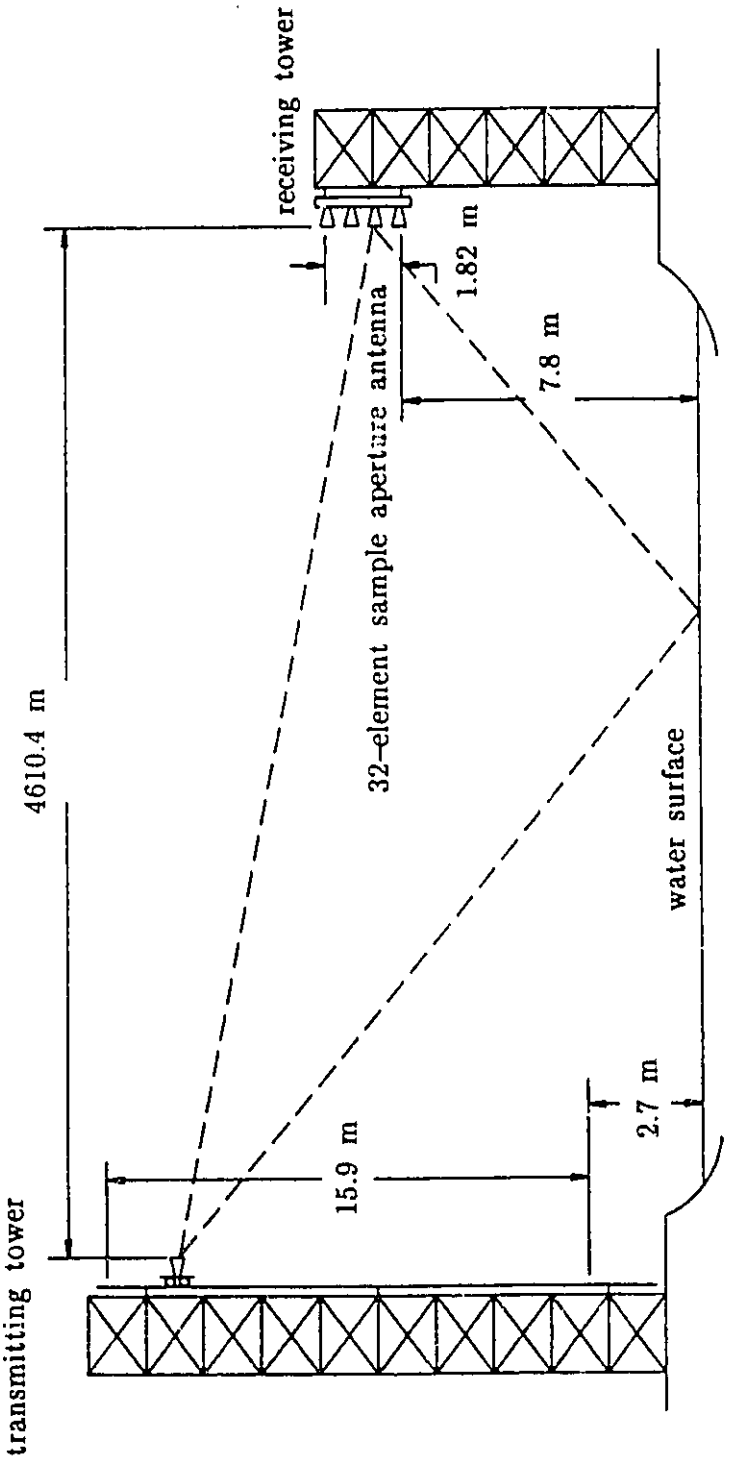


Fig. B.1 Geometry of the experiment setup

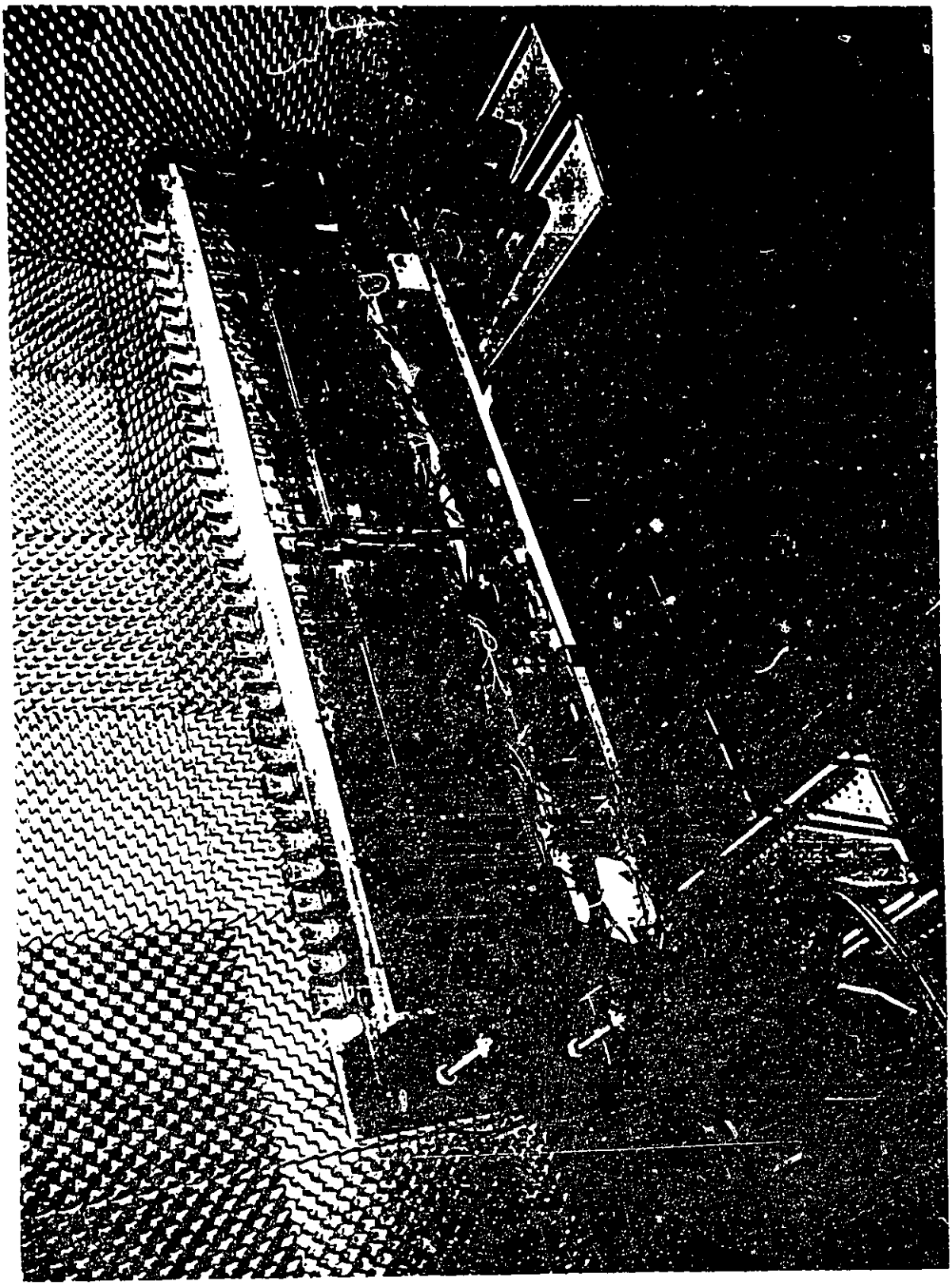


Fig. B.2 Physical view of the sampled aperture array

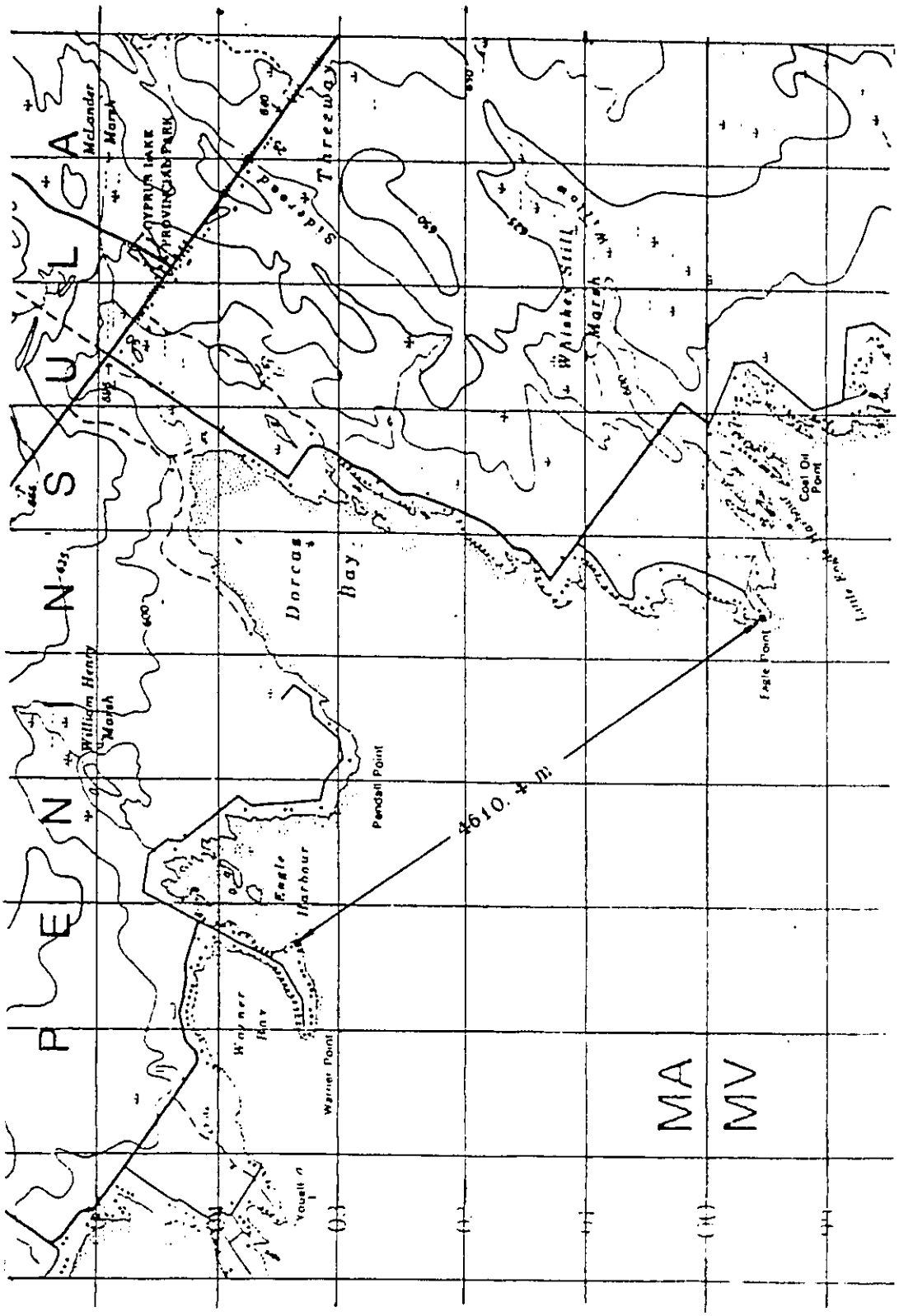


Fig. B.3 Plan view of the measurement site in Dorcus Bay, Ontario

where D is the maximum dimension of the antenna, and λ is the wavelength of the radar signal. For the geometry in Fig. B.1

$$r_f = \frac{2(1.82)^2}{.02941} = 225.24 \text{ m},$$

which is less that 4610 meters. Therefore, the receiver site is the in far field region, and the received signals can be approximately assumed to be plane waves.

B.2. Multipath Phenomenon

In Fig. B.1, it is seen that the received signal consists of, not only the direct signal, but also a surface reflected or multipath signal. That is, it consists of two components: 1) the direct component from the transmitter, and 2) the specular (indirect) component reflected from the surface. The direct component constitutes the desired signal and the latter the undesired signal. In the measurements that were carried out the multipath signal was very close to the desired one; their separation is being a fraction of a beamwidth. The over–sea multipath problem should not be underestimated since it may corrupt the antenna's performance, especially when the antenna is operating under rough sea surface conditions. In this case, the indirect signal usually consists of two components; (1) a coherent component (specular) and (2) a noise–like component (diffuse).

The current earth geometry which describes the characteristics of the signals when they are reflected by a curved earth is given in Fig. B.4. The source and the receiving antenna are separated by the range d. The point of reflection,

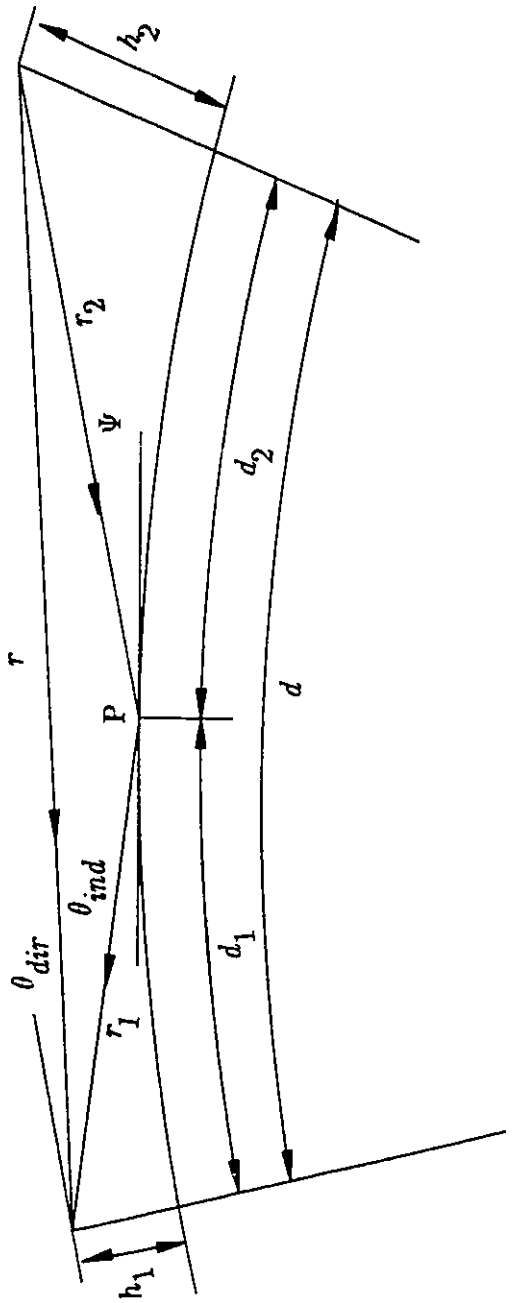


Fig. B.4 Geometry of direct and indirect signals on a curved surface

indicated by P, determines the distances d_1, d_2, and the grazing angle ψ. Also, in Fig. A.4, h_1 is the height of the centre of the antenna aperture and h_2 is the height of the source. The angle–of–arrival for the direct and indirect signals are θ_{dir} and θ_{ind}, respectively, and they are given by [B2]

$$\theta_{dir} = \tan^{-1}\{\frac{h_2 - h_1 - d^2/R_e}{d}\} \tag{B.1}$$

$$\theta_{ind} = \tan^{-1}\{\frac{h_1 + d_1^2/R_e}{d_1}\} \tag{B.2}$$

where

$$d_1 = \frac{d}{2} + p\cos(\frac{\psi + \pi}{2}),$$

$$p = 2\{\frac{1}{3}[R_e(h_1 + h_2) + (\frac{d}{2})^2]\}^{\frac{1}{2}},$$

$$\psi = \cos^{-1}\{\frac{1}{p^3}[2\mathrm{Re}(h_1 - h_2)d]\}, \text{ for } h_1 \leq h_2,$$

R_e = effective radius of the earth = τR, and

R = radius of the earth = 6.38×10^6 m.

The parameter τ is given in terms of refractivity $\frac{\mathrm{d}N}{\mathrm{d}h}$ of the troposphere by

$$\tau \simeq \frac{1}{1 + .00637\frac{\mathrm{d}N}{\mathrm{d}h}} \tag{B.3}$$

The multipath signal model received at the element m of the antenna array can be expressed as

$$x_m = x_{m_{dir}} + x_{m_{ind}}$$

$$= Ae^{j(m-1)\varphi_{dir}} + |\rho|Ae^{j(m-1)(\varphi_{ind} + \Delta\varphi)} \tag{B.4}$$

where A is the amplitude of the measured signal, and ρ is the complex reflection coefficient with $\Delta\varphi$ is its phase. Since the difference between the direct and indirect path, given by

$$\Delta r = r_1 + r_2 - r, \tag{B.5}$$

is virtually constant, the amplitude and phase differences between $x_{m_{dir}}$ and $x_{m_{ind}}$ remain unchanged from one data sample to another, provided that the source elevation does not vary appreciably over the observational time interval. Thus the expected value of the cross–correlation of the two signals at the output of the mth element is

$$E[x_{m_{dir}} x^{*}_{m_{ind}}] = |\rho| . |A|^2 \tag{B.6}$$

which is a constant. This implies that the direct and indirect signals can be highly correlated. The coherence between these signals will degrade the performance of most superresolution techniques, in terms of their ability to resolve the signals.

B.3. Array Calibration Method

It is necessary to calibrate the received data because of the inevitable presence of measurement errors, which are due to: 1) direct current (dc) offsets in the I and Q channels, 2) differences in the gain and phase characteristics of

individual receivers, and 3) discrepancies in the test and local oscillator signals supplied to the receivers [A1]. The dc offset, and partially errors in gain and phase of the antenna array as well as other discrepancies can be removes by using IQ calibration and far field calibration procedures [B3]. The errors, which remain after these processes, are termed as *calibration errors.*

In baseband, the received signal will then can be modeled as

$$x_m(n) = g_m(n) \sum_{k=1}^{K} s_k(n) e^{j(m-1)\varphi_k} + \nu_m(n) \tag{B.7}$$

in which

$$g_m(n) = (1 + \delta_{am}) e^{j\delta_{pm}} \tag{B.8}$$

where K is the number of signals, which is equal to 2 in the case of the experimental setup above, and $g_m(n)$ represents the calibration errors. In Eqn. (B.8), δ_{am}, δ_{pm} respectively denote the relative errors in amplitude and phase of the mth element. Note that these errors can not be totally removed by the IQ calibration and the far field calibration processes due to drift in the individual array element, which results in different forms from one snap shot to another. Thus, an algorithm, which is capable of automatical correcting these errors in real–time basis, is necessary. This algorithm, so called self–calibration algorithm, is developed and presented in Chapter 5.

References

[B1] S. Haykin and V. Kezys, "Multi–parameter adaptive radar system (MARS)", 30*th Midwest Symposium on Circuits and Systems*, 1988.

[B2] J. Litva and P. Bauman, "Studies of correlation between meteorological and low–level radar angle–of–arrival measurement", *CRL Report* No. 174, Nov. 1986.

[B3] T. Lo, "Adaptive beam–space nulling of multipath signals", M.Eng. Thesis, Dept. of Electrical and Computer Eng., McMaster Univ., 1989.