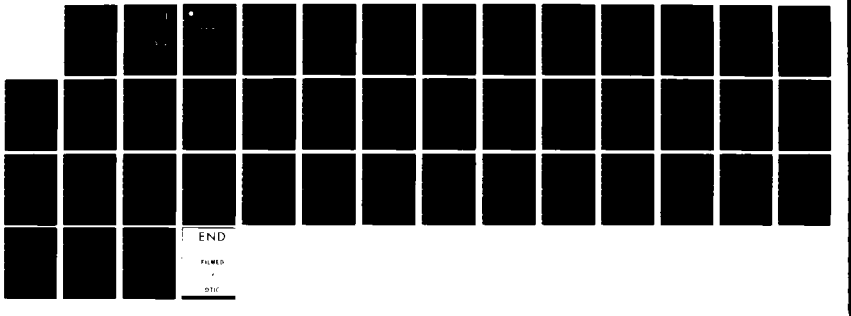
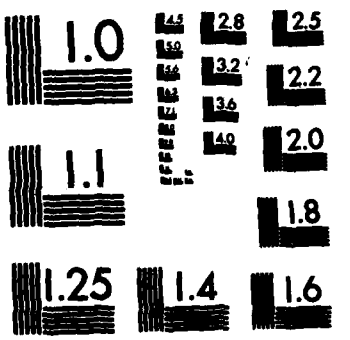
AD-A124 439 A GAIN OPTIMIZING ALGORITHM FOR ADAPTIVE ARRAYS(U) OHIO 1/1
STATE UNIV COLUMBUS ELECTROSCIENCE LAB
H H AL-KHATIB ET AL. SEP 75 ESL-4063-2 N00019-75-C-0179
UNCLASSIFIED F/G 12/1 NL





PHOTOGRAPH THIS SHEET

AD A124439

DTIC ACCESSION NUMBER

II

LEVEL

I

INVENTORY

Rept. No. ESL-4063-2

DOCUMENT IDENTIFICATION

Contract N00019-75-C-0179 Sept. 75

DISTRIBUTION STATEMENT A
Approved for public release;
Distribution Unlimited

DISTRIBUTION STATEMENT

ACCESSION FOR	
NTIS	GRA&I
DTIC	TAB
UNANNOUNCED	
JUSTIFICATION	
BY	
DISTRIBUTION /	
AVAILABILITY CODES	
DIST	AVAIL AND/OR SPECIAL
A	

DISTRIBUTION STAMP

DTIC
ELECTE
S FEB 16 1983 D
D

DATE ACCESSIONED

DTIC COPY INSPECTED

43

83 02 010 042

DATE RECEIVED IN DTIC

PHOTOGRAPH THIS SHEET AND RETURN TO DTIC-DDA-2

DTIC FORM OCT 79 70A

DOCUMENT PROCESSING SHEET

A GAIN OPTIMIZING ALGORITHM FOR ADAPTIVE ARRAYS

H. H. Al-Khatib and R. T. Compton, Jr.

The Ohio State University

ElectroScience Laboratory

Department of Electrical Engineering
Columbus, Ohio 43212

Quarterly Technical Report 4063-2

September 1975

Contract N00019-75-C-0179

Department of the Navy
Naval Air Systems Command
Washington, D.C. 20361

NOTICES

When Government drawings, specifications, or other data are used for any purpose other than in connection with a definitely related Government procurement operation, the United States Government thereby incurs no responsibility nor any obligation whatsoever, and the fact that the Government may have formulated, furnished, or in any way supplied the said drawings, specifications, or other data, is not to be regarded by implication or otherwise as in any manner licensing the holder or any other person or corporation, or conveying any rights or permission to manufacture, use, or sell any patented invention that may in any way be related thereto.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER	2. GOVT ACCESSION NO. / 3. RECIPIENT'S CATALOG NUMBER
4. TITLE *(and Subtitle)*: A GAIN OPTIMIZING ALGORITHM FOR ADAPTIVE ARRAYS	5. TYPE OF REPORT & PERIOD COVERED: Second Quarterly Technical Report
	6. PERFORMING ORG. REPORT NUMBER: ESL 4053-2 6
7. AUTHOR(s): H. H. Al-Khatib and R. T. Compton, Jr.	8. CONTRACT OR GRANT NUMBER(s): Contract N00019-75-C-0179
9. PERFORMING ORGANIZATION NAME AND ADDRESS: The Ohio State University ElectroScience Laboratory, Department of Electrical Engineering Columbus, Ohio	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS: Department of the Navy, Naval Air Systems Command, Washington, D.C. 20361	12. REPORT DATE: September 1975 13. NUMBER OF PAGES: 38
14. MONITORING AGENCY NAME & ADDRESS *(if different from Controlling Office)*	15. SECURITY CLASS. *(of this report)*: Unclassified 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT *(of this Report)*

APPROVED FOR PUBLIC RELEASE: DISTRIBUTION UNLIMITED

17. DISTRIBUTION STATEMENT *(of the abstract entered in Block 20, if different from Report)*

18. SUPPLEMENTARY NOTES

19. KEY WORDS *(Continue on reverse side if necessary and identify by block number)*

Gain Optimizimg algorithm
Adaptive Arrays
Adaptive Antennas

20. ABSTRACT *(Continue on reverse side if necessary and identify by block number)*

An algorithm for optimizing the gain of an adaptive antenna system is discussed in this report. The gain optimization is accomplished by a constrained gradient-search technique. The algorithm is presented and some simple examples showing how the gain is optimized in a two-element array are given. The convergence and stability of the algorithm are also discussed.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

CONTENTS

I. INTRODUCTION

This report discusses an adaptive algorithm for optimizing the gain of an antenna array on an incoming signal. The algorithm is based on a steepest-ascent maximization of the array output power, subject to a constraint on the array weights.

This algorithm is under investigation for use in conjunction with a power inversion adaptive array [1-5], a modified version of the LMS adaptive array [6,7]. A power inversion adaptive array can provide significant protection from interference in a spread spectrum communication system, but it does not provide any form of beam tracking on the desired signal. Beam tracking can be obtained, however, by first arraying antenna elements in pairs with power inversion feedback, and then combining the element-pair outputs with the gain optimizing algorithm described here. In this report, we discuss only the gain optimizing algorithm. The combined system will be the subject of a future report.

This report discusses

#1473

II. A GAIN OPTIMIZING ALGORITHM

Consider an array of N antenna elements as shown in Fig. 1. For simplicity, the elements are assumed to be isotropic and non-interacting. The signal from each element, $Y_i(t)$, is passed through a processor P_i that generates K outputs labeled $X_i(t)$ on Fig. 1. Each output $X_i(t)$ is multiplied by a real weighting coefficient W_i and then is summed to produce the array output $S(t)$. The processor P_i may be either a quadrature hybrid with two outputs as shown in Fig. 2, or a tapped delay-line with K outputs, as shown in Fig. 3. A quadrature hybrid processor (Fig. 2) provides a simple magnitude and phase adjustment of the signal $Y_i(t)$ and is the appropriate form of processing when the signals are narrowband. A tapped delay-line processor (Fig. 3) provides a frequency dependent transfer function behind each element and is appropriate for wider bandwidth signals* The gain optimizing algorithm to be developed in this report may be used with either type of processing, and an example of each will be given below.

In this report we develop an iterative algorithm for adjusting the weights W_i such that the array gain is maximized on an incoming signal. The algorithm is based on a steepest-ascent maximization of the array output power, subject to a constraint to prevent the weights from going to infinity. The algorithm discussed here is in the spirit

---

*Reference [8] contains a quantitative comparison of the bandwidth performance of quadrature hyprid and tapped delay-line processors for a two-element array.

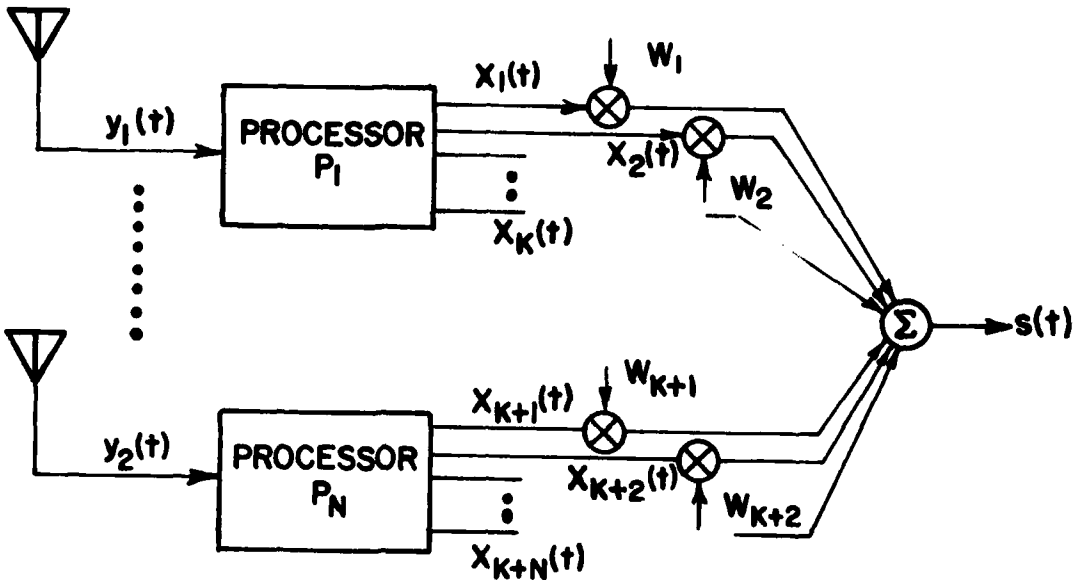


Fig. 1. An adaptive array system.

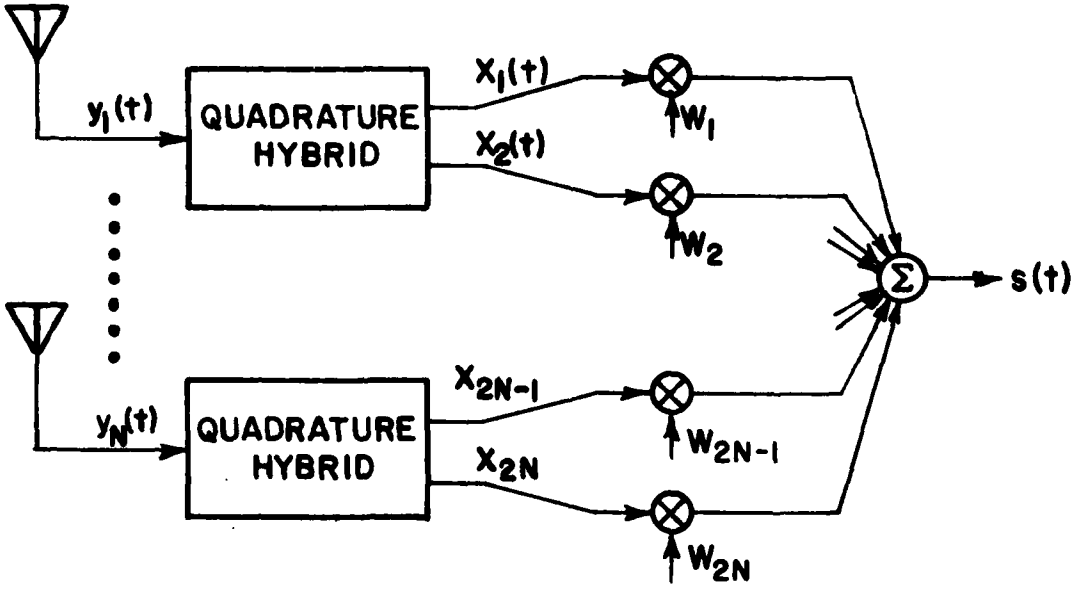


Fig. 2. Array with quadrature hybrid processing.

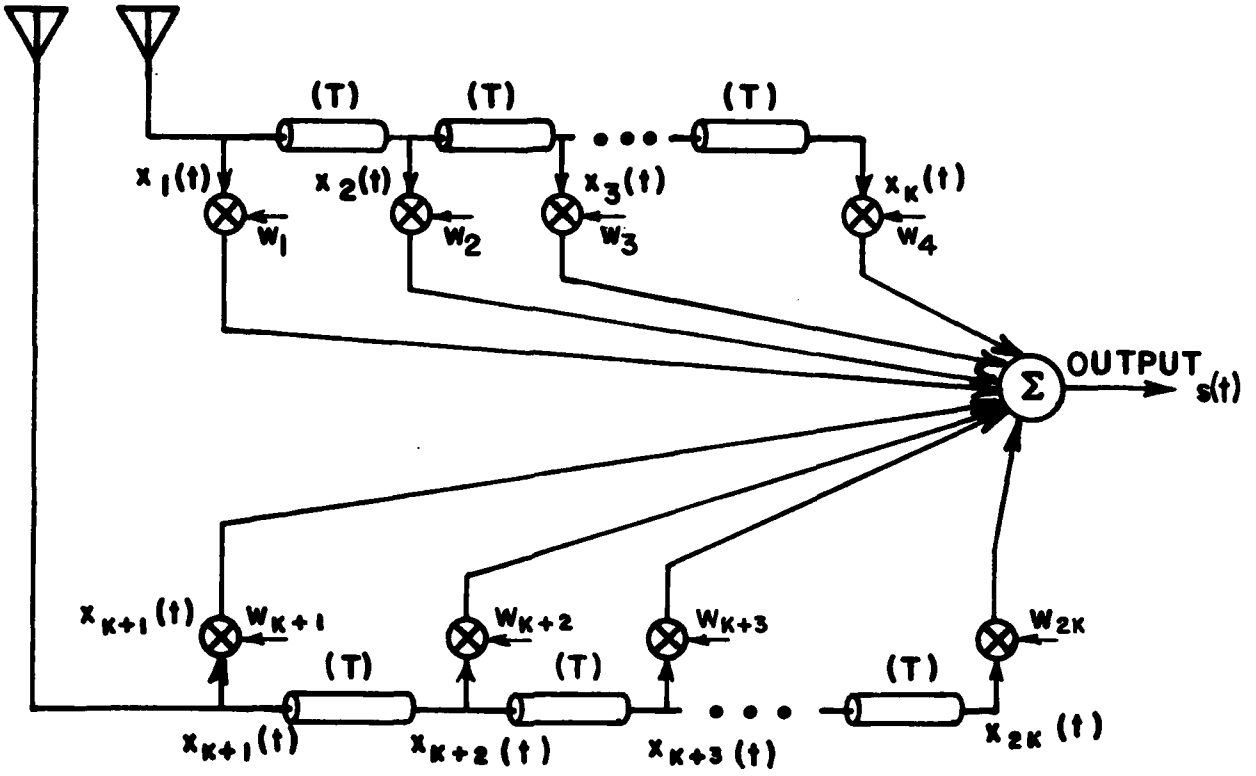


Fig. 3. Array with tapped delay-line processing.

of the LMS algorithm developed by Widrow, et al. [6], and discussed in many reports on adaptive arrays [7,9]. The general properties of constrained gradient techniques have been discussed by Eveleigh [10] and by Gill and Murray [11].

Let us assume that each of the signals $X_i(t)$ is sampled periodically in time, with an interval between samples equal to Δt. Let $X_i(j)$ denote the value of $X_i(t)$ at the j^{th} sampling instant and also let

$$(1) \qquad X(j) = \begin{bmatrix} X_1(j) \\ X_2(j) \\ \vdots \\ X_{NK}(j) \end{bmatrix}$$

be a column vector with components $X_i(j)$. Also, let us assume that at the j^{th} sampling instant, the weights in the array have the value $W_i(j)$, which may be represented by the column vector $W(j)$;

$$(2) \qquad W(j) = \begin{bmatrix} W_1(j) \\ W_2(j) \\ \vdots \\ W_{NK}(j) \end{bmatrix}$$

We wish to develop an iterative algorithm by which these weights can be adjusted at each sample to maximize the array gain on an incoming signal.

Intuitively speaking, maximum array gain on an incoming signal will result in the greatest output signal power from the array. Since the array output signal is

$$(3) \qquad S(t) = \sum_{i=1}^{NK} W_i X_i(t) \quad ,$$

the average output power is

$$(4) \qquad \overline{S^2(t)} = \sum_{i=1}^{NK} \sum_{1=1}^{NK} W_i W_j X_i(t) X_j(t)$$

where the overbar denotes the time average. We would like to choose the weights so $\overline{S^2(t)}$ is maximum. However, there is clearly no upper limit to $\overline{S^2(t)}$ if the W_i are allowed to take on arbitrary values, so it is necessary to maximize $\overline{S^2(t)}$ subject to some constraint on the values of the weights. Several types of constraints appear to be possible, but in this report we assume the weights must satisfy the constraint equation

$$(5) \qquad \sum_{i=1}^{NK} W_i^2 = 1 \quad .$$

I.e., the weights are constrained to lie on the surface of a hypersphere of unit radius.

To optimize the gain of the array, we adopt the following iterative algorithm

$$W(j+1) = W(j) + k\nabla(j) \tag{6}$$

where

$W(j)$ = weight vector at the j^{th} iteration

$W(j+1)$ = weight vector at the $(j+1)^{st}$ iteration

k = scalar constant controlling the rate of adaptation

and $\nabla(j)$ = a vector correction term chosen to move the weights toward the maximum gain setting.

The correction term $\nabla(j)$ is obtained by taking the gradient of $\overline{S^2(t)}$ and retaining only its component parallel to the hypersphere. Since the gradient of $\overline{S^2(t)}$ is given by

$$\nabla_W(S^2) = \begin{bmatrix} \frac{\partial \overline{S^2}}{\partial W_1} \\ \frac{\partial \overline{S^2}}{\partial W_2} \\ \vdots \\ \frac{\partial \overline{S^3}}{\partial W_{Nk}} \end{bmatrix} \tag{7}$$

we find from Eq. (4) that

$$\nabla_W(\overline{S^2}) = 2\begin{bmatrix} X_1(t) \\ X_2(t) \\ \vdots \\ X_{NK}(t) \end{bmatrix} \sum_{j=1}^{NK} W_j X_j(t) = 2\begin{bmatrix} X_1(t)S(t) \\ X_2(t)S(t) \\ \vdots \\ X_{NK}(t)S(t) \end{bmatrix} \tag{8}$$

This gradient vector can be written as the sum of two components -- one perpendicular to the weight hypersurface and one tangent to the hypersurface. That is,

$$\nabla_W(\overline{S^2}) = \nabla_{W_{\parallel}} (\overline{S^2}) + \nabla_{W_{\perp}} (\overline{S^2}) \tag{9}$$

where $\nabla_{W_{\parallel}} (\overline{S^2})$ and $\nabla_{W_{\perp}} (\overline{S^2})$ are tangent and perpendicular, respectively, to the surface of the hypersphere, as shown in Fig. 4. To obtain $\nabla_{W_{\perp}} (\overline{S^2})$, we let n_W denote a unit vector normal to the surface of the hypersphere. (n_W is also shown in Fig. 4.) $\hat{n}_W$ will be given by

$$\hat{n}_W = \begin{bmatrix} \frac{W_1}{R} \\ \frac{W_2}{R} \\ \vdots \\ \frac{W_{NK}}{R} \end{bmatrix} \tag{10}$$

where

$$R = \sqrt{W_1^2 + W_2^2 + \cdots W_{NK}^2} \tag{11}$$

$\nabla_W (\overline{S^2})$ is then given by (T denotes the transpose)

$$\begin{aligned} \nabla_{W_{\perp}} (\overline{S^2}) &= [\hat{n}_W^T \nabla_W(\overline{S^2})] \hat{n}_W \\ &= \left(\frac{W_1\overline{X_1S} + W_2\overline{X_2S} + \cdots + W_{NK}\overline{X_{NK}S}}{R} \right) \hat{n}_W \\ &= \left(\frac{\overline{S^2}}{R} \right) n_W \\ &= \frac{\overline{S^2}}{R^2} \begin{bmatrix} W_1 \\ W_2 \\ \vdots \\ W_{NK} \end{bmatrix} \end{aligned} \tag{12}$$

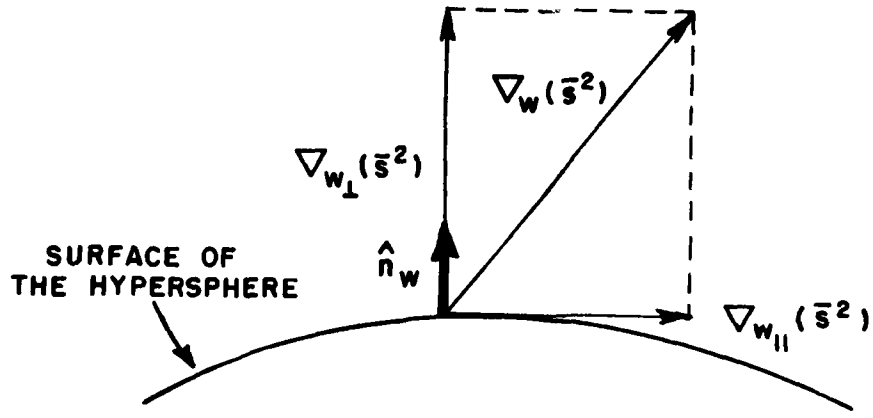


Fig. 4. Perpendicular and tangential components of $\nabla_w(\bar{S}^2)$.

By subtracting $\nabla_w \ (\overline{S^2})$ from $\nabla_w(\overline{S^2})$ (see Eq. (9)), we obtain

$$(13) \qquad \nabla_{w_{||}} \ (\overline{S^2}) = \begin{bmatrix} \overline{X_1 S} \\ \overline{X_2 S} \\ \vdots \\ \overline{X_{NK} S} \end{bmatrix} - \begin{bmatrix} \frac{\overline{S^2} W_1}{R^2} \\ \frac{\overline{S^2} W_2}{R^{22}} \\ \vdots \\ \frac{S^2 W_{NK}}{R^2} \end{bmatrix}$$

In view of the constraint equation,

$$(14) \qquad R^2 = \sum_{i=1}^{NK} W_i^{\,2} = 1 \quad , \qquad (14)$$

we may drop the R^2 term to obtain

$$(15) \qquad \nabla_{W_n} (\overline{S^2}) = \begin{bmatrix} \overline{SX_1} \\ \overline{SX_2} \\ \vdots \\ \overline{SX_{NK}} \end{bmatrix} - \begin{bmatrix} W_1\overline{S^2} \\ W_2S^2 \\ \vdots \\ W_{NK}\overline{S^2} \end{bmatrix} = \begin{bmatrix} \overline{S(X_1 - W_1S)} \\ \overline{S(X_2 - W_2S)} \\ \vdots \\ \overline{S(X_{NK} - W_{NK}S)} \end{bmatrix}$$

Since we would like to control the weights in real time, it is impractical to compute the time average of the quantities $S(X_i - W_iS)$ indicated by the overbar in Eq. (15). It would be possible to compute the average of this quantity over a finite time period and consider this as an estimate of the infinite time average. However, as is done in the LMS algorithm (6), we will adopt the simplest estimate of all, namely, we ignore the averaging completely and just use

$$(16) \qquad \nabla_{W_{ii}} \simeq \begin{bmatrix} S(X_1 - W_1S) \\ S(X_2 - W_2S) \\ \vdots \\ S(X_{NK} - W_{NK}S) \end{bmatrix}$$

The iterative algorithm given in Eq. (6) thus becomes

$$(17) \qquad W_i(j+1) = W_i(j) + kS(j) \quad [X_i(j) - W_i(j)S] \quad .$$

Let us first consider what happens with this algorithm if the weights are off the constraint surface for some reason. We would like the algorithm to be such that if the weights drift off the surface, they will automatically return to the surface. With this algorithm, it turns out they will do this. To see why, note that the second term in Eq. (13) represents the perpendicular component of the gradient. In simplifying Eq. (13), we obtained Eq. (15) by dropping the term

$$R^2 = \sum_{i=1}^{NK} W_i^2 \ ,$$

which is unity if the weight constraint is satisfied. Note, however, that if the weights are off the hypersphere for some reason, the equation

$$\sum W_1^2 = 1$$

will not hold. Hence, the second vector in Eq. (15) will be parallel to the perpendicular component of the gradient, but will have the wrong magnitude. If, for example,

$$\sum W_i^2 > 1,$$

the vector

$$\begin{matrix} W_1\overline{S^2} \\ W_2\overline{S^2} \\ \vdots \\ W_{NK}\overline{S^2} \end{matrix} \tag{18}$$

has a greater magnitude than $\nabla_{W_\perp}(\overline{S^2})$. Then in Eq. (15), in the second term, we subtract a vector greater than the perpendicular component of $\nabla_W(\overline{S^2})$. Note that since the perpendicular component of $\nabla_W(\overline{S^2})$ always points outward away from the origin, (we can always increase the output power $\overline{S^2}$ from the array simply by increasing all weight coefficients in proportion), the second term subtracted in Eq. (15) overcorrects for the perpendicular component and the resulting total vector in Eq. (15) points back toward the hypersphere; i.e., it has the correct parallel component but has a perpendicular component pointing inward. This inward perpendicular component will cause the weights to move back onto the constraint surface.

In a similar way, if

$$\sum W_i^2 < 1,$$

the second term in Eq. (15) will have a smaller magnitude than the perpendicular component of $\nabla_W(\overline{S^2})$. The net vector in Eq. (15) will not have its perpendicular component completely cancelled--there will be a residual radial component, which will move the weights back out toward the surface.

Thus, we see that dropping the term

$$\sum W_i^2 = 1$$

in going from Eq. (13) to Eq. (15), which was done to simplify the algorithm, is important because it makes the weights always tend toward the constraint surface.

Finally, we discuss the stability of the iterative algorithm in Eq. (17). Because the signals are sampled at time increments Δt and a correction is applied to the weights at each sample instant, the system controlling the weights may be viewed as a sampled data control system. Hence, we may expect that if the loop gain constant k is too large, the system will become unstable.

To obtain an idea of the suitable range of values for k, let us suppose that in the steady state, the i^{th} array weight will have the value W_{oi}. (Note that the final steady-state solution is not unique -- there are many sets of array weights that will maximize the array gain toward a given signal. W_{oi} represents one possible set of steady-state weights.) Suppose at the j^{th} iteration, $W_i(j)$ differs from W_{oi} by $\Delta W_i(j)$:

$$W_i(j) = W_{oi} + \Delta W_i(j) \tag{19}$$

Using this equation in Eq. (17) yields

$$W_{oi} + \Delta W_i(j+1) = W_{oi} + \Delta W_i(j) + kS(j)[X_i(j) - W_{oi}S(j) - \Delta W_i(j)S(j)] \; . \tag{20}$$

Since W_{oi} is a steady-state solution to the weight iteration equation, W_{oi} will have the property that the average value of the quantity

$$S(j)\,[X_i(j) - W_{oi}S(j)]$$

is zero; i.e., in the steady-state, the correction term in Eq. (17) will average to zero over many samples, so both $W_i(j)$ and $W_i(j+1)$ assume the same value, W_{oi}. Cancelling the W_{oi} term and dropping the term $S(j)\,[X_i(j) - W_{oi}S(j)]$ from Eq. (2) leaves

$$\begin{aligned} \Delta W_i(j+1) &= \Delta W_i(j) - kS(j)\Delta W_i(j)S(j) \\ &= \Delta W_i(j)\,[1 - kS^2(j)] \end{aligned} \tag{21}$$

In order for the algorithm to converge, the average value of the factor $1 - kS^2(j)$ must be such that

$$\left| \frac{\Delta W_i(j+1)}{\Delta W_i(j)} \right| = |1 - kS^2(j)| < 1 \tag{22}$$

so that at successive iterations the difference between $W_i(j)$ and W_{oi} becomes smaller. To satisfy Eq. (22), k must lie in the range

$$(23) \qquad 0 < K < \frac{1}{2\overline{S^2(j)}}$$

Furthermore, since

$$(24) \qquad S(j) = X^T W = W^T X \quad ,$$

we have

$$(25) \qquad \overline{S^2(j)} = W^T \overline{XX^T} W \quad .$$

Note that the constraint equation

$$\sum W_i^2 = 1$$

implies that the vector W is a unit vector. Hence, the maximum possible value of $\overline{S^2(t)}$ will equal the maximum eigenvalue of the matrix

$$(26) \qquad \Phi = \overline{XX^T}$$

(This case will occur if the unit vector W lies along the principal axis of Φ associated with this eigenvalue.) Thus, if λ_{max} denotes the maximum eigenvalue of Φ, the feedback algorithm will be stable as long as k lies in the range

$$(27) \qquad 0 < k < \frac{1}{2\lambda_{max}} \quad .$$

We can obtain a more readily usable upper bound for k by noting that

$$(28) \qquad \lambda_{max} \leq \text{Trace } \Phi = \sum_{i=1}^{NK} \overline{X_i^2(t)} \quad ,$$

so the algorithm will be stable if k is restricted to the narrower range

$$(29) \qquad 0 < k < \frac{1}{2 \sum_{i=1}^{NK} \overline{X_i^2(t)}} \quad .$$

Note that

$$\sum_{i=1}^{NK} X_i{}^2(t)$$

is proportional to the total power incident on the array.

Examples

Now we give two examples of the use of this algorithm. The first example is a two-element array with quadrature hybrid processing and a narrowband signal. The second example is a two-element array with tapped delay-line processing and a wide bandwidth signal.

EXAMPLE 1:

Consider a two-element array as shown in Fig. 5. A signal is assumed to propagate into the array from an angle relative to broadside. (We assume the antenna elements to be isotropic and non-interacting.) We will assume also that there is no noise. As a result, the received signals in the elements are given by

$$(30) \qquad Y_1(t) = a \cos [\omega_o t]$$

$$(31) \qquad Y_2(t) = a \cos [\omega_o t - \phi_o]$$

where a is the amplitude of the signal, ω_o is the carrier frequency, and ϕ_o is the interelement phase shift due to the propagation delay:

$$(32) \qquad \phi_o = \frac{2\pi L}{\lambda_o} \sin \theta$$

(L is the element spacing and λ_o is the free-space wavelength.) The signal $Y_1(t)$ is arbitrarily chosen to have zero electrical phase angle.

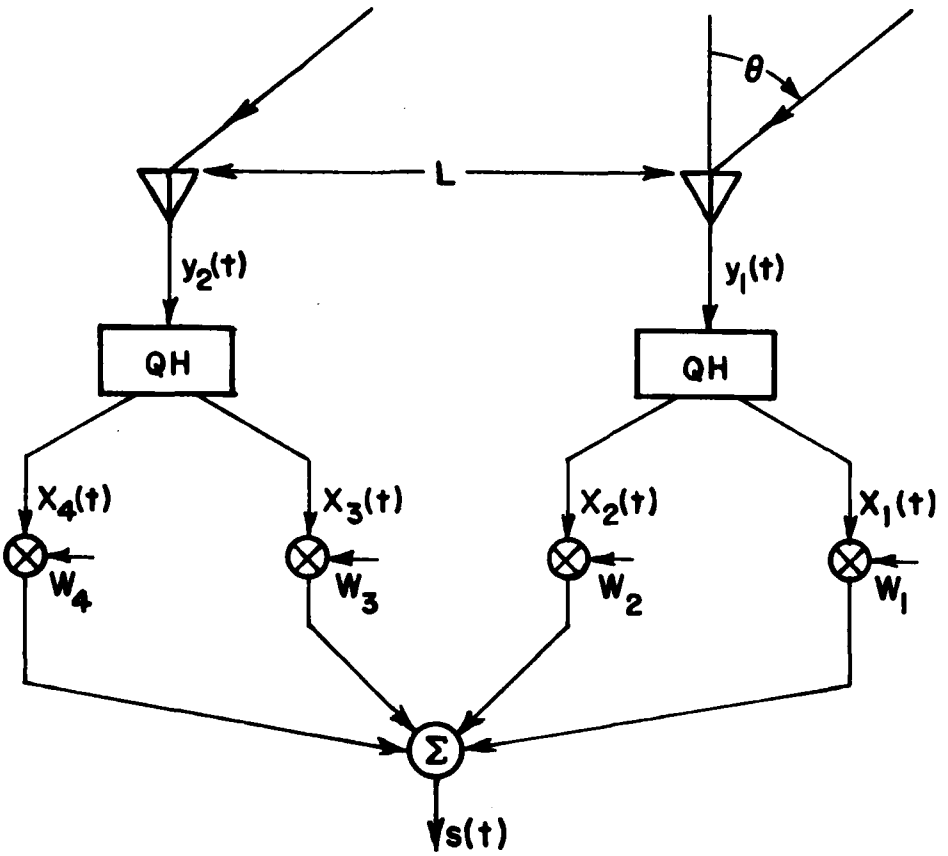


Fig. 5. A two-element array with quadrature hybrids.

The signal from each element is passed through a quadrature hybrid; the signals out of the hybrids are denoted by $X_i(t)$. $X_1(t)$ and $X_2(t)$ are the in-phase and quadrature components, respectively, of $Y_1(t)$, and $X_3(t)$ and $X_4(t)$ are the in-phase and quadrature components of $Y_2(t)$. Thus, we have

(33) $$X_1(t) = a \cos(\omega_0 t)$$

(34) $$X_2(t) = a \sin(\omega_0 t)$$

(35) $$X_3(t) = a \cos(\omega_0 t - \phi_0)$$

(36) $$X_4(t) = a \sin(\omega_0 t - \phi_0)$$

A Fortran computer program was written (see Appendix I) to simulate the behavior of the iterative algorithm in Eq. (17). In the examples to be shown below, we have arbitrarily chosen

(37) $a = 1$

(38) $L = \lambda_o/2$ (half-wavelength spacing),

and for the initial value of the weight vector,

(39) $$W = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} .$$

(Note that this satisfies the constraint equation.)

Some results of this simulation ar displayed in Figs. 6 to 13.

In Figs. 6 and 7, the desired signal arrives from broadside ($\theta = 0°$). Figure 6 shows the transients that result in the four weights, and Fig. 7 shows the final array pattern after the weight transients have ended. It may be seen that the beam maximum points in the proper direction, and it can be shown that the final array pattern has the maximum possible gain in the direction of the desired signal.* Figures 8 and 9 show similar results for the case when the desired signal arrives from $\theta = 30°$. Figures 10 and 11 show $\theta = 60°$, and Figs. 12 and 13 show $\theta = 90°$. In all cases, the final weights in the array yield a maximum possible gain in the signal direction, and it can be seen from the patterns how the beam is steered toward the signal.

*For a broadside signal, maximum gain occurs when $W_1 = W_3$ and $W_2 = W_4$.

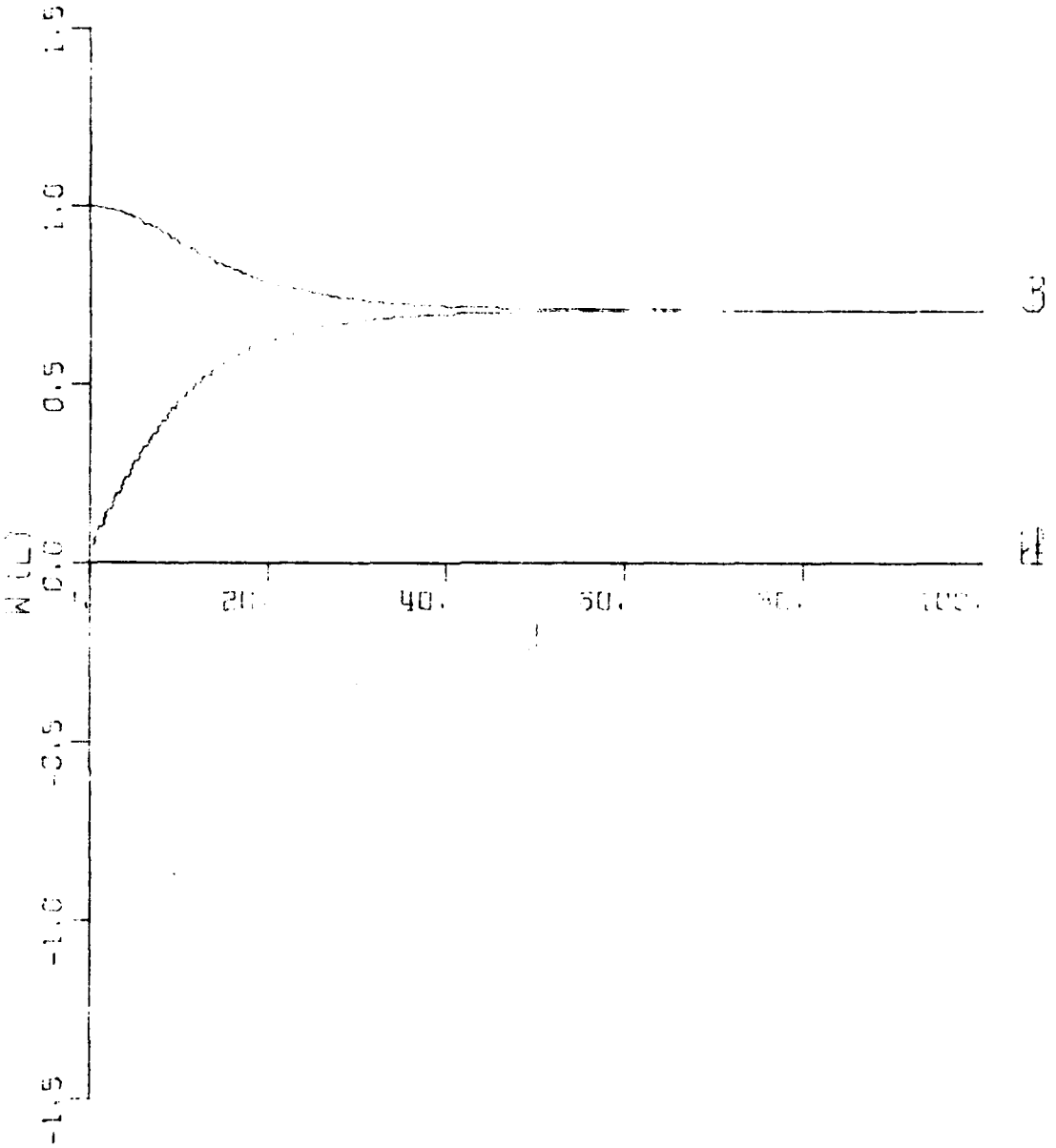


Fig. 6. Weight transients θ=0° (quadrature hybrids).

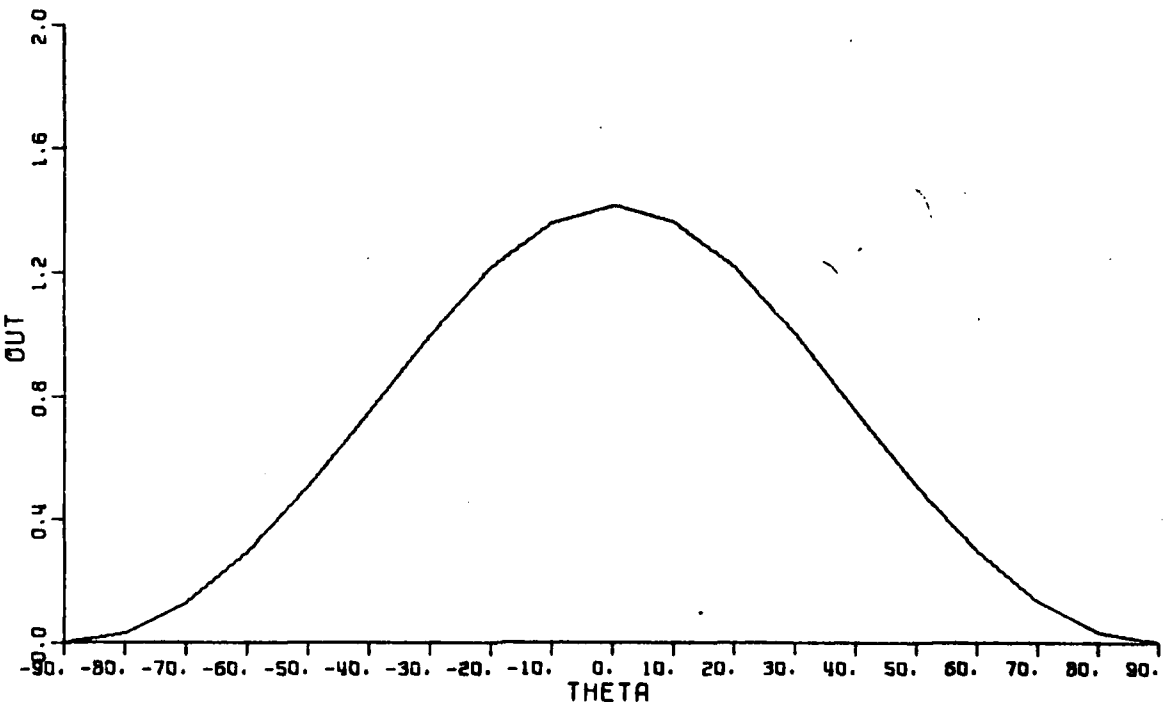


Fig. 7. Final pattern, θ=0° (quadrature hybrids).

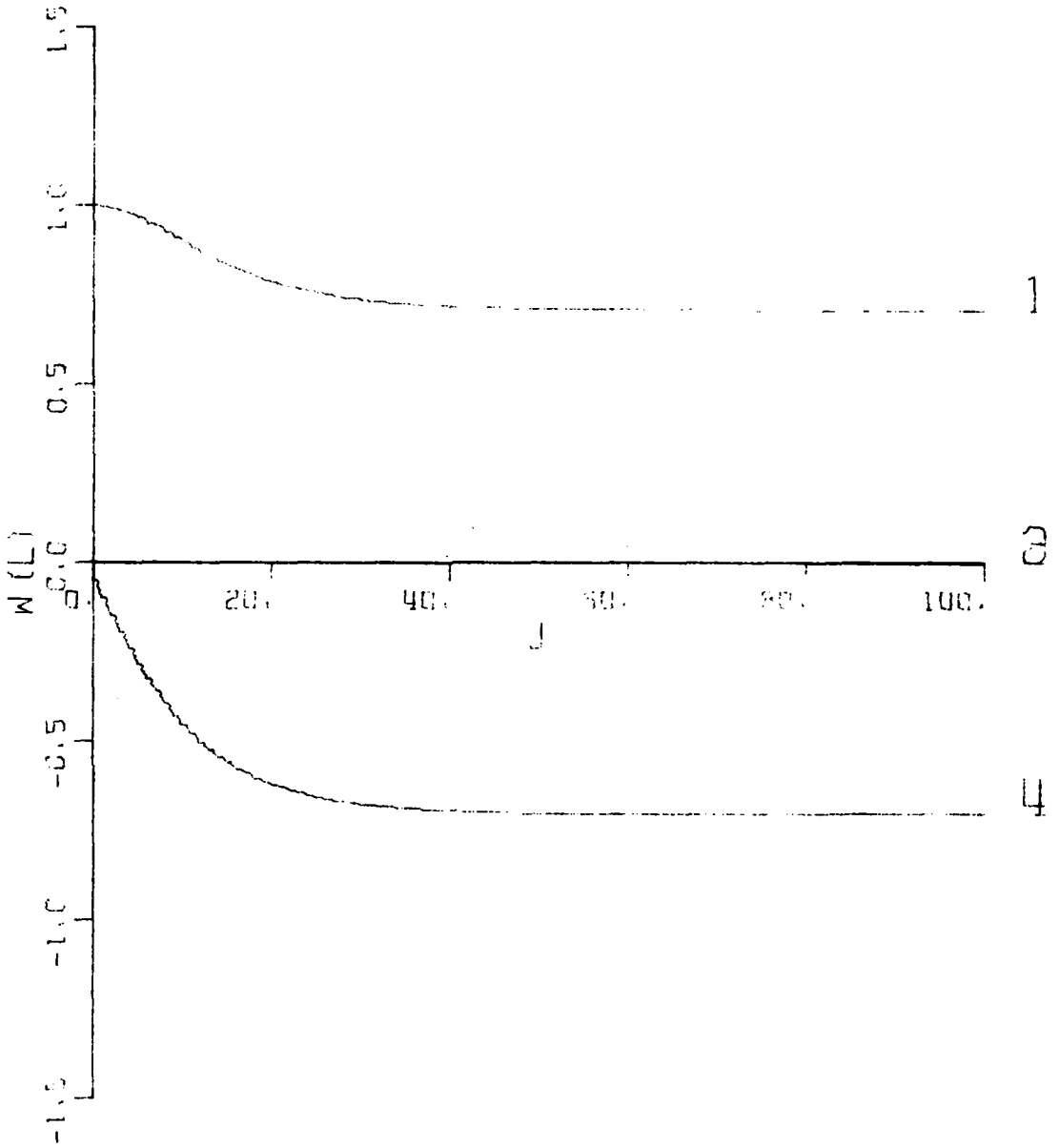


Fig. 8. Weight transients, θ=30° (quadrature hybrids).

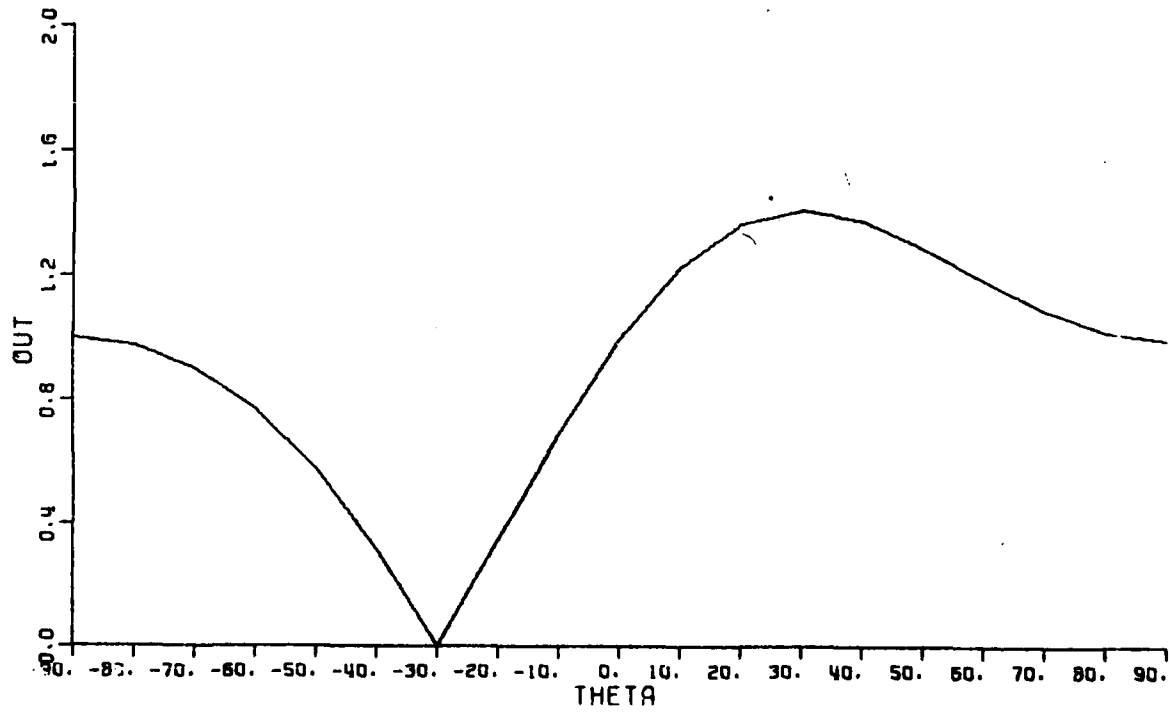


Fig. 9. Final pattern, θ=30° (quadrature hybrids).

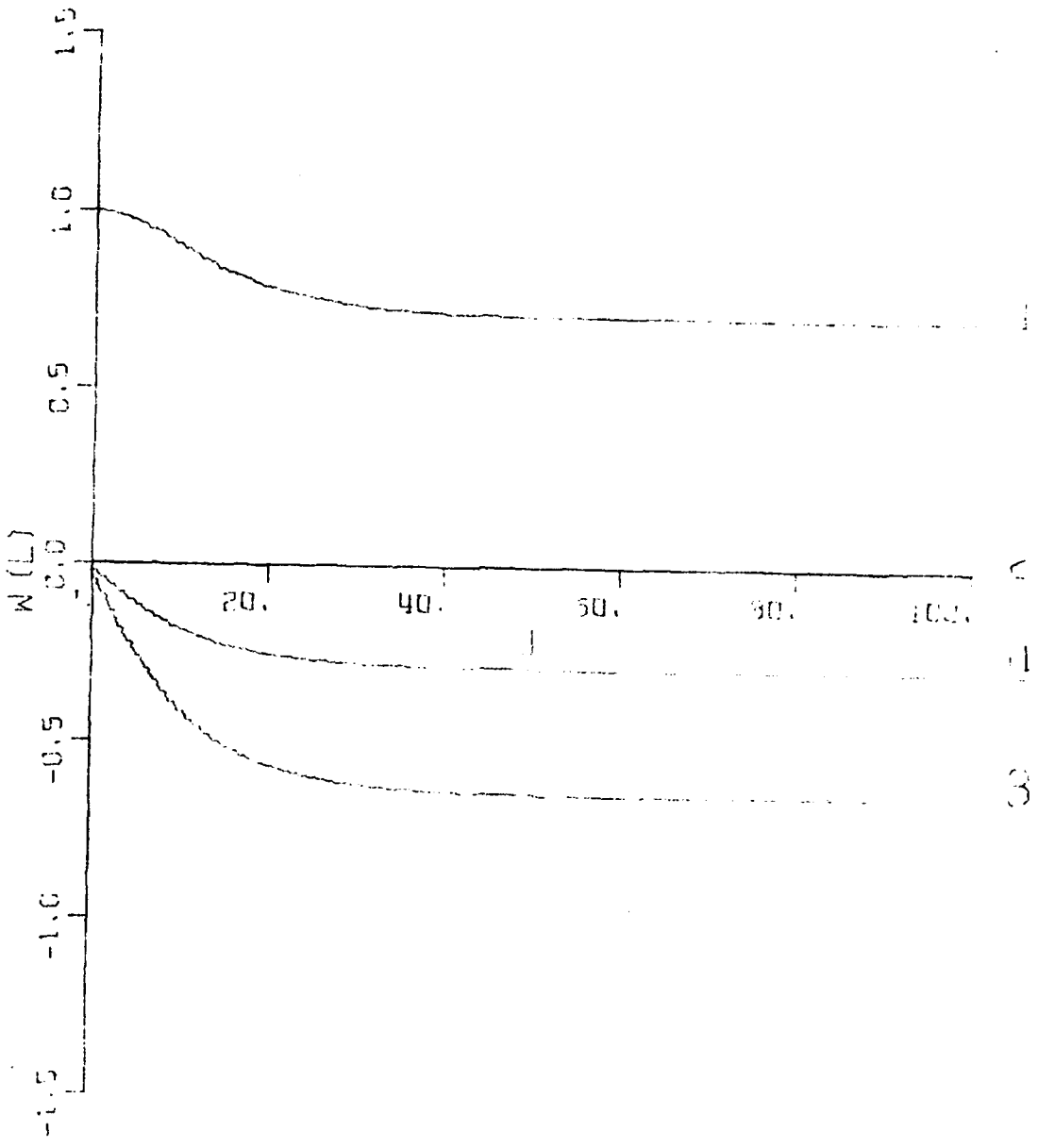


Fig. 10. Weight transients, θ=60° (quadrature hybrids).

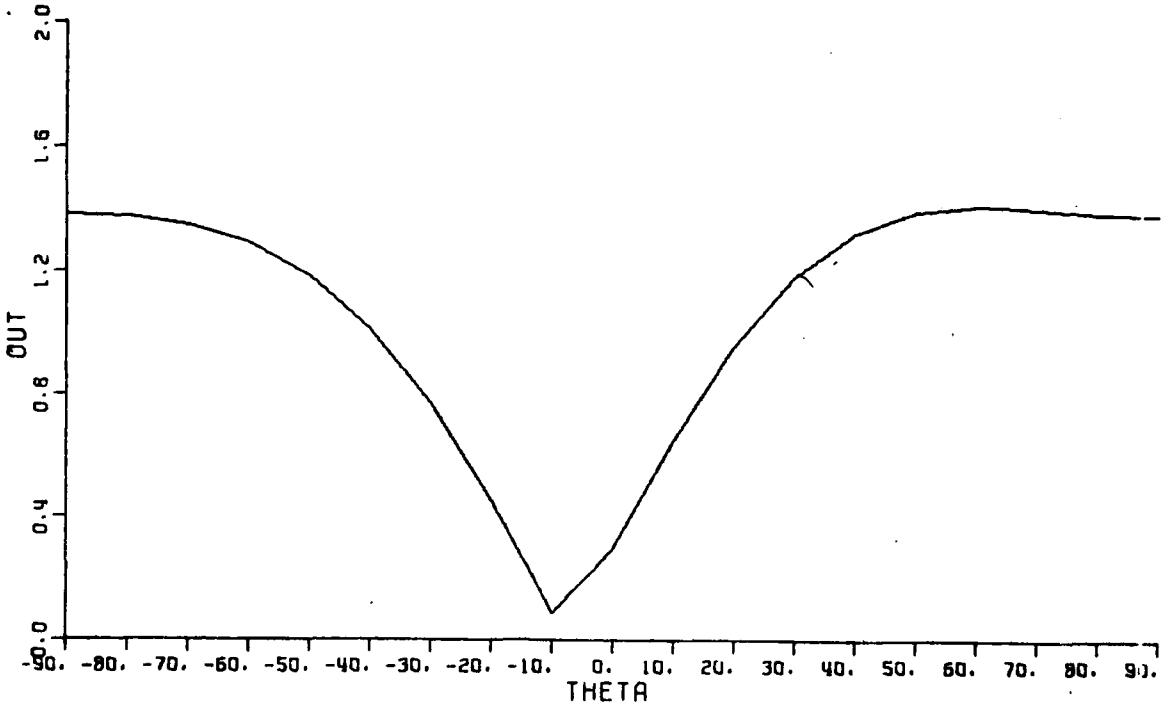


Fig. 11. Final pattern, θ=60° (quadrature hybrids).

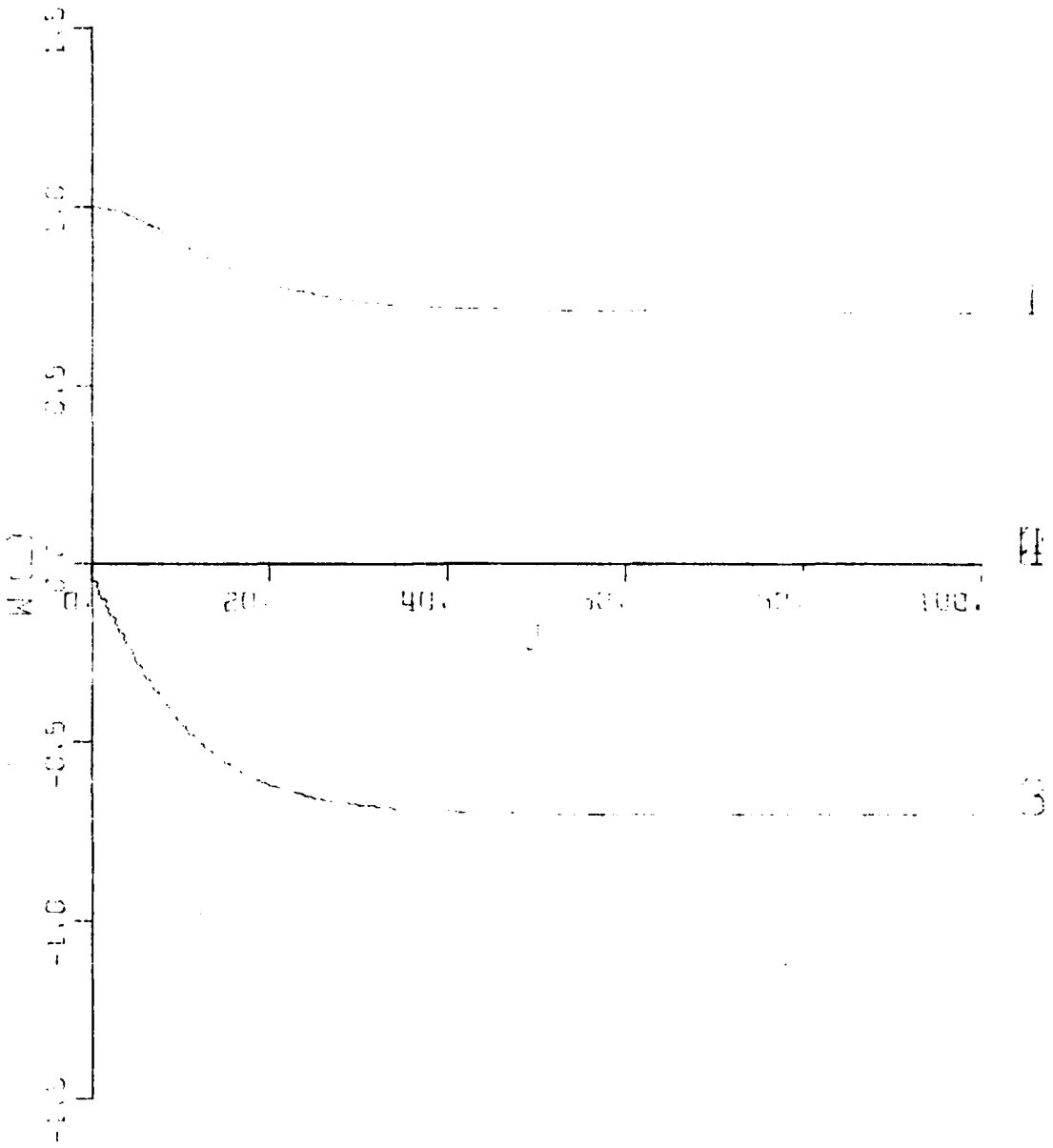


Fig. 12. Weight transients, θ=90° (quadrature hybrids).

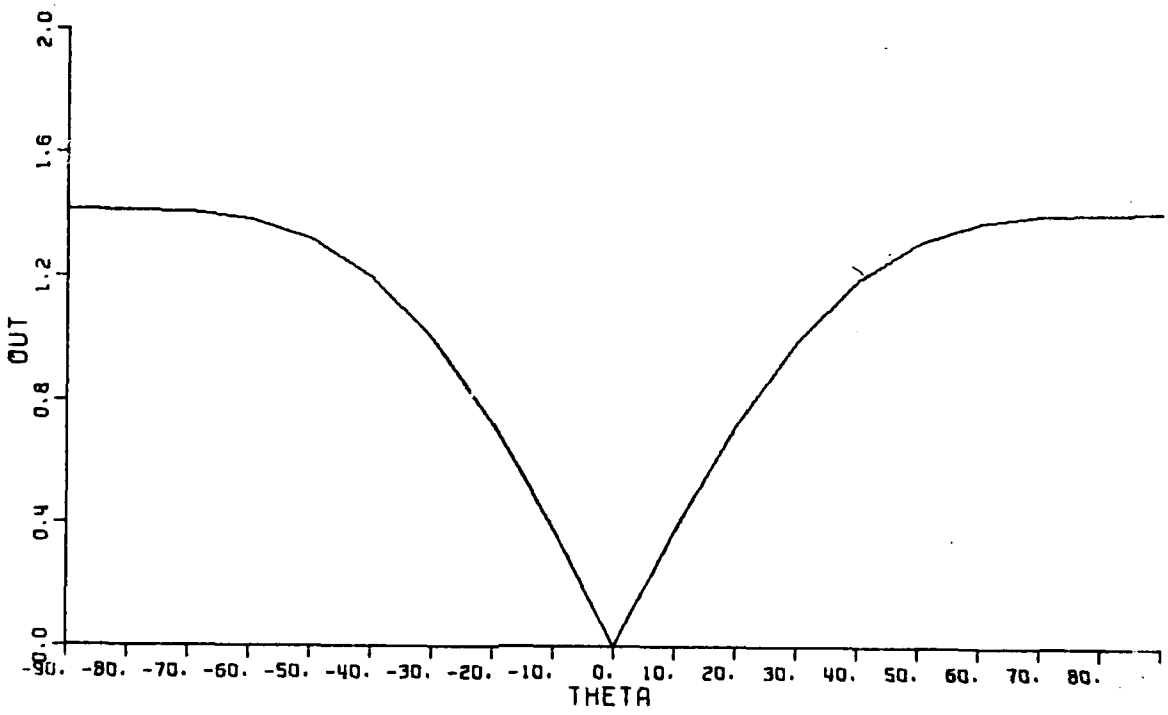


Fig. 13. Final pattern, θ=90° (quadrature hybrids).

EXAMPLE 2

We again consider a two-element array but with a two-section tapped delay line processor behind each element, as shown in Fig. 14. Each delay line section is a quarter wavelength long at the center frequency of the signal.

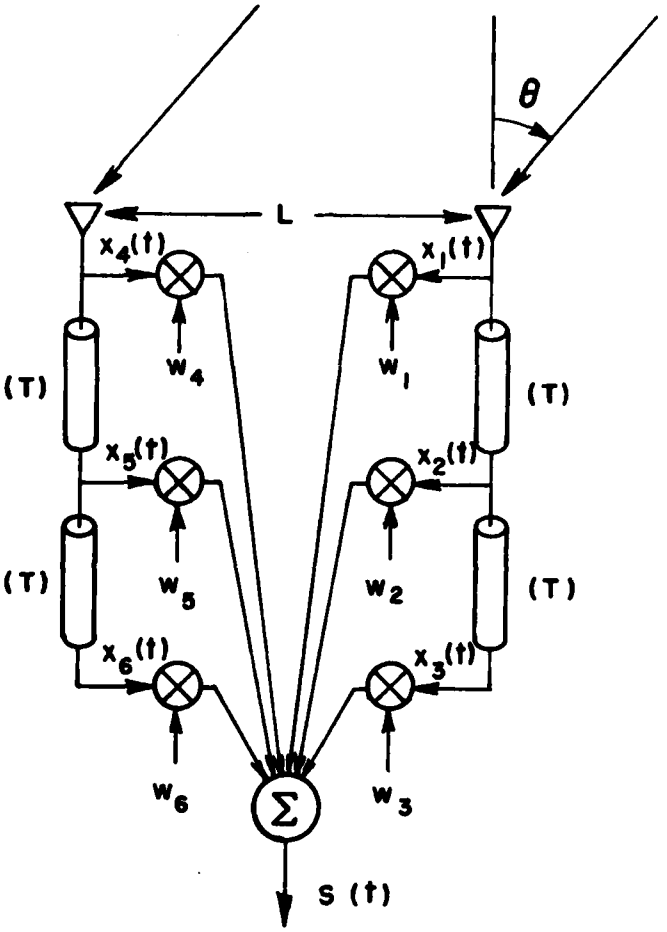


Fig. 14. A two-element array with tapped delay-lines.

An amplitude modulated signal is assumed to propagate into the array and to produce element signals

$$Y_1(t) = a[1 + \cos \omega_m t] \cos [\omega_o t] \tag{40}$$

(41) $$Y_2(t) = a[1 + \cos\ \omega_m(t-\tau)] \cos\ [\omega_o(t - \tau)]$$

where a is the amplitude, ω_m is the modulation frequency, ω_o is the carrier frequency, and τ is the propagation time delay between elements, given by

(42) $$\tau = \frac{L}{c} \sin \theta \quad .$$

Each delay line section (between taps) is a quarter wavelength long at frequency ω_o and hence causes a time delay T equal to

(43) $$T = \frac{\tau}{2\omega_o}$$

Thus

(44) $$X_1(t) = Y_1(t) \quad ,$$

(45) $$X_2(t) = Y_1(t - T) \quad ,$$

(46) $$X_3(t) = Y_1(t - 2T) \quad ,$$

(47) $$X_4(t) = Y_2(t) \quad ,$$

(48) $$X_5(t) = Y_2(t - T)$$

and

(49) $$X_6(t) = Y_2(t - 2T) \quad .$$

The feedback algorithm in Eq. (17) has been simulated in this problem (see Appendix II) with the following parameter values

(50) $$a = 1$$

(51) $$\omega_m = \frac{\omega_o}{8}$$

(52) $$L = \frac{\lambda_o}{2}$$

and the arrival angle $\theta = 0°$, 30°, 60°, and 90°. The initial weight vector was chosen to be

$$W = \begin{matrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{matrix} .$$

The weight transients and the resulting array patterns (computed at frequency ω_o) are shown in Figs. 15 through 22.

Figures 15 and 16 show the weight transients and the final array pattern (at frequency ω_o) when $\theta = 0°$. Figures 17 and 18 show similar results for $\theta = 30°$, Figs. 19 and 20 show $\theta = 60°$, and Figs. 21 and 22 show $\theta = 90°$. In all cases, the resulting weight settings maximize the pattern response toward the signal.

CONCLUSIONS

This report has discussed an iterative algorithm for adjusting the weights in an adaptive array to maximize the array gain on an incoming signal. The algorithm is based on a steepest-ascent maximization of the array output power subject to the constraint that the sum of the squares of the array weights is constant. It was shown that the algorithm prevents the weights from drifting away from the constraint surface and also that the algorithm is stable for a suitable range of the feedback gain constant.

Two examples showing that the algorithm does optimize gain were presented, one with a CW signal and guadrature hybrid processing behind the elements, and the other with a modulated signal and tapped delay-line processing behind the elements.

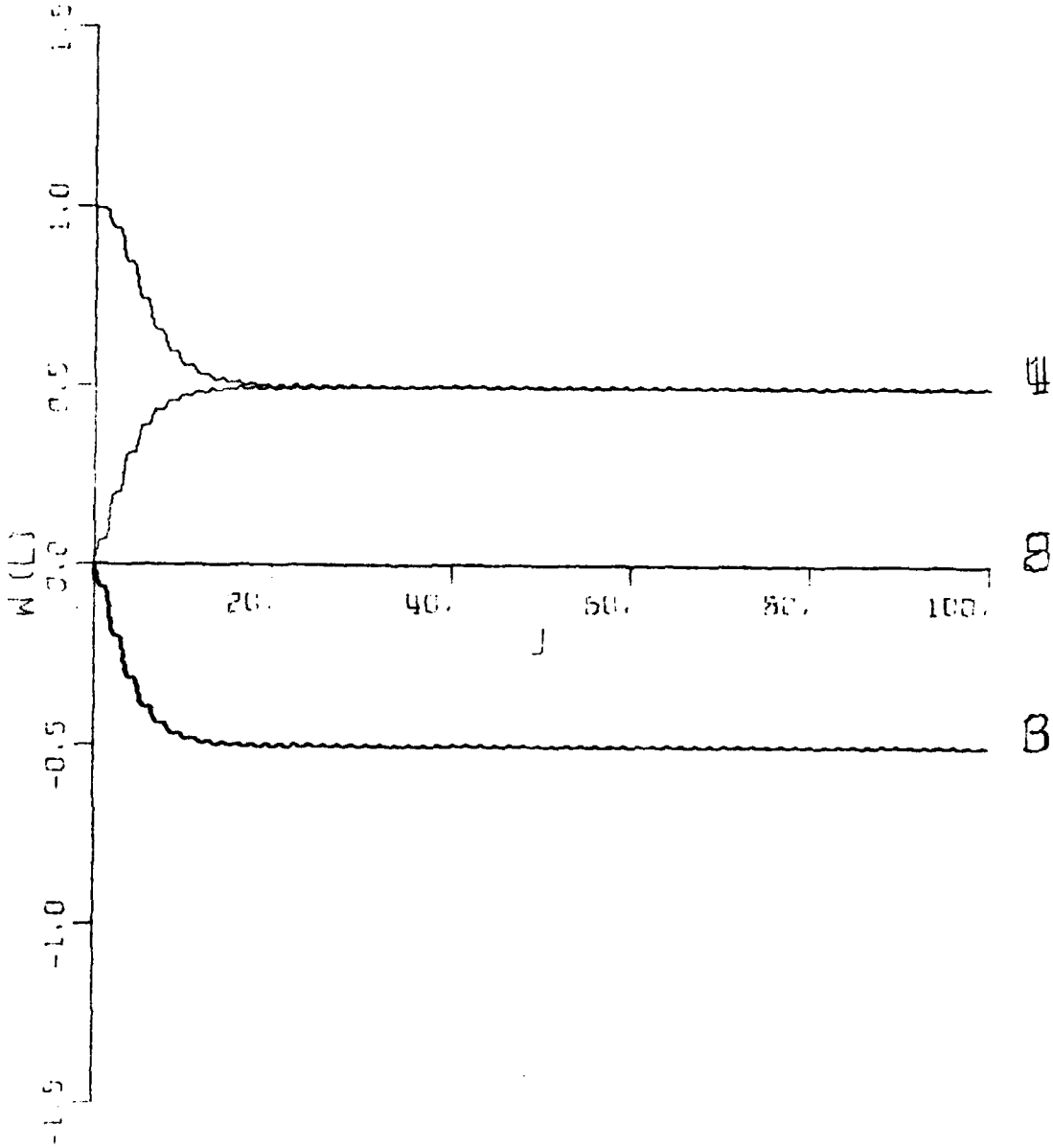


Fig. 15. Weight transients, θ=0° (tapped delay-lines).

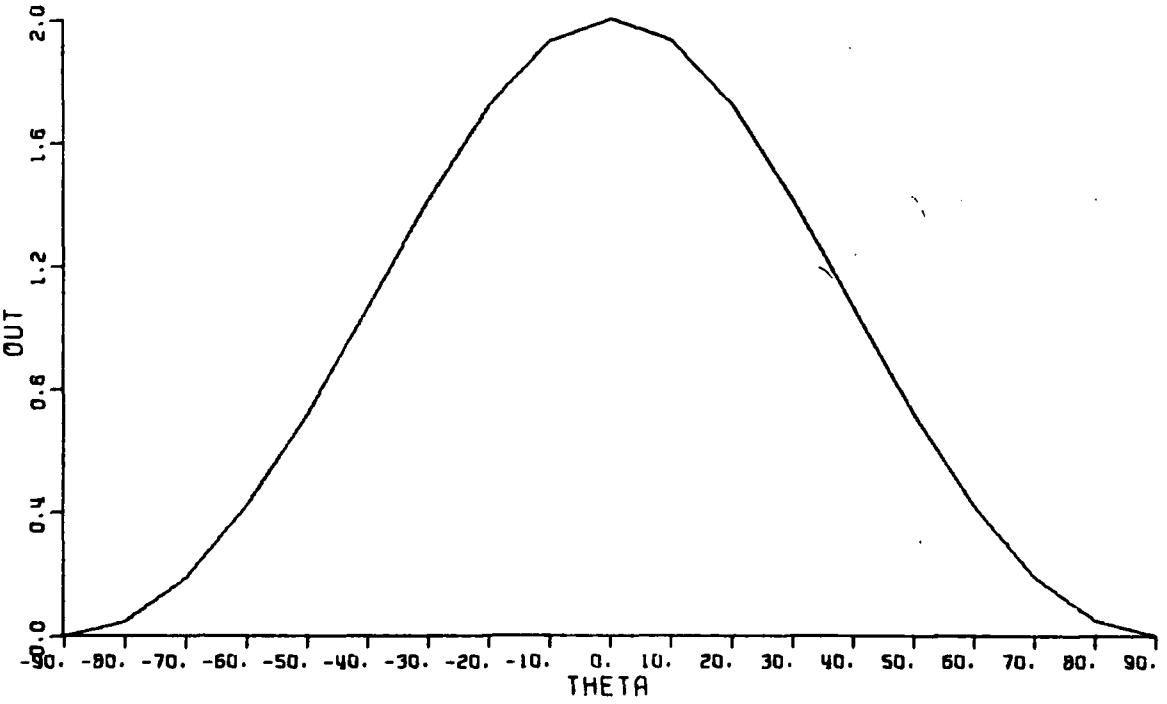


Fig. 16. Final pattern, θ=0° (tapped delay-lines).

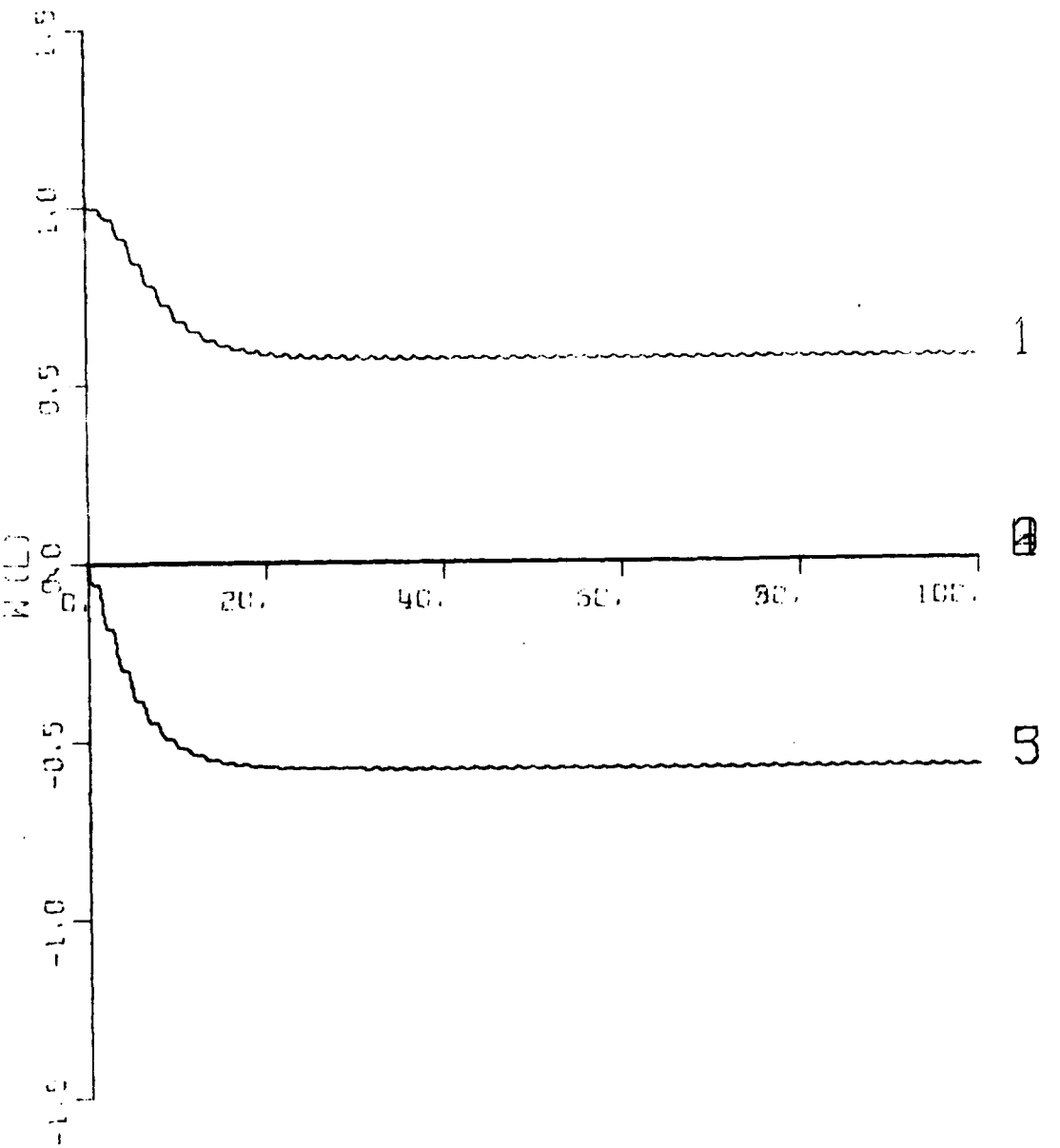


Fig. 17. Weight transients, θ=30° (tapped delay-lines).

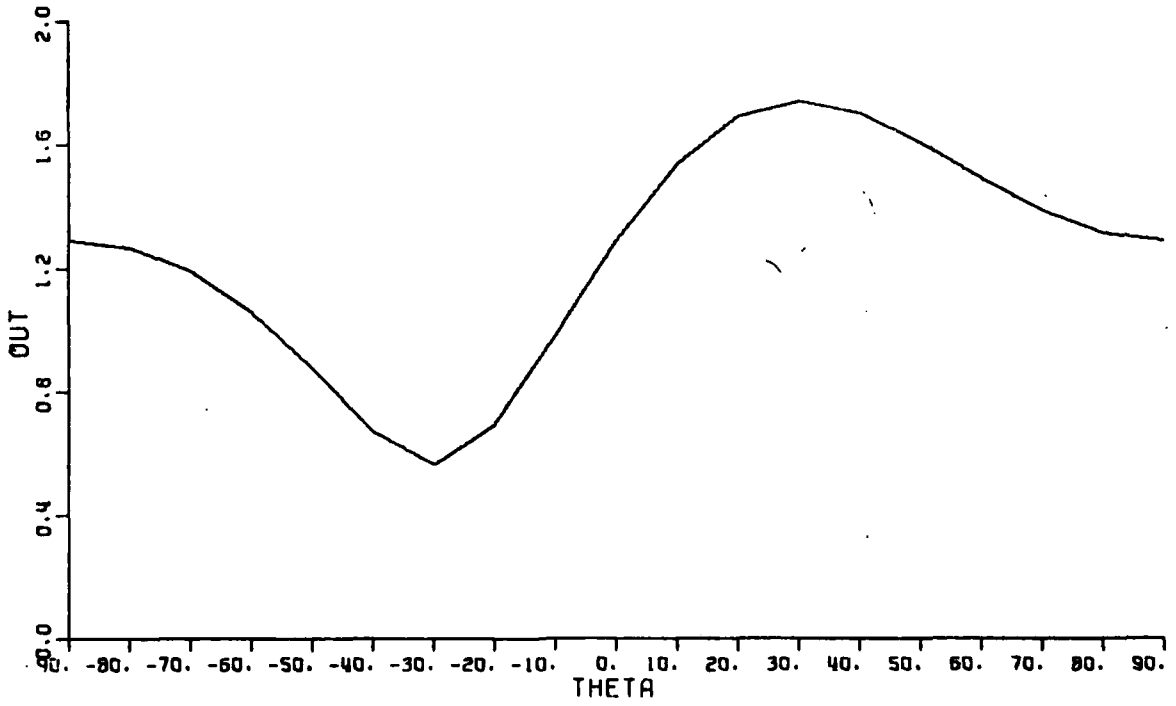


Fig. 18. Final pattern, θ=30° (tapped delay-lines).

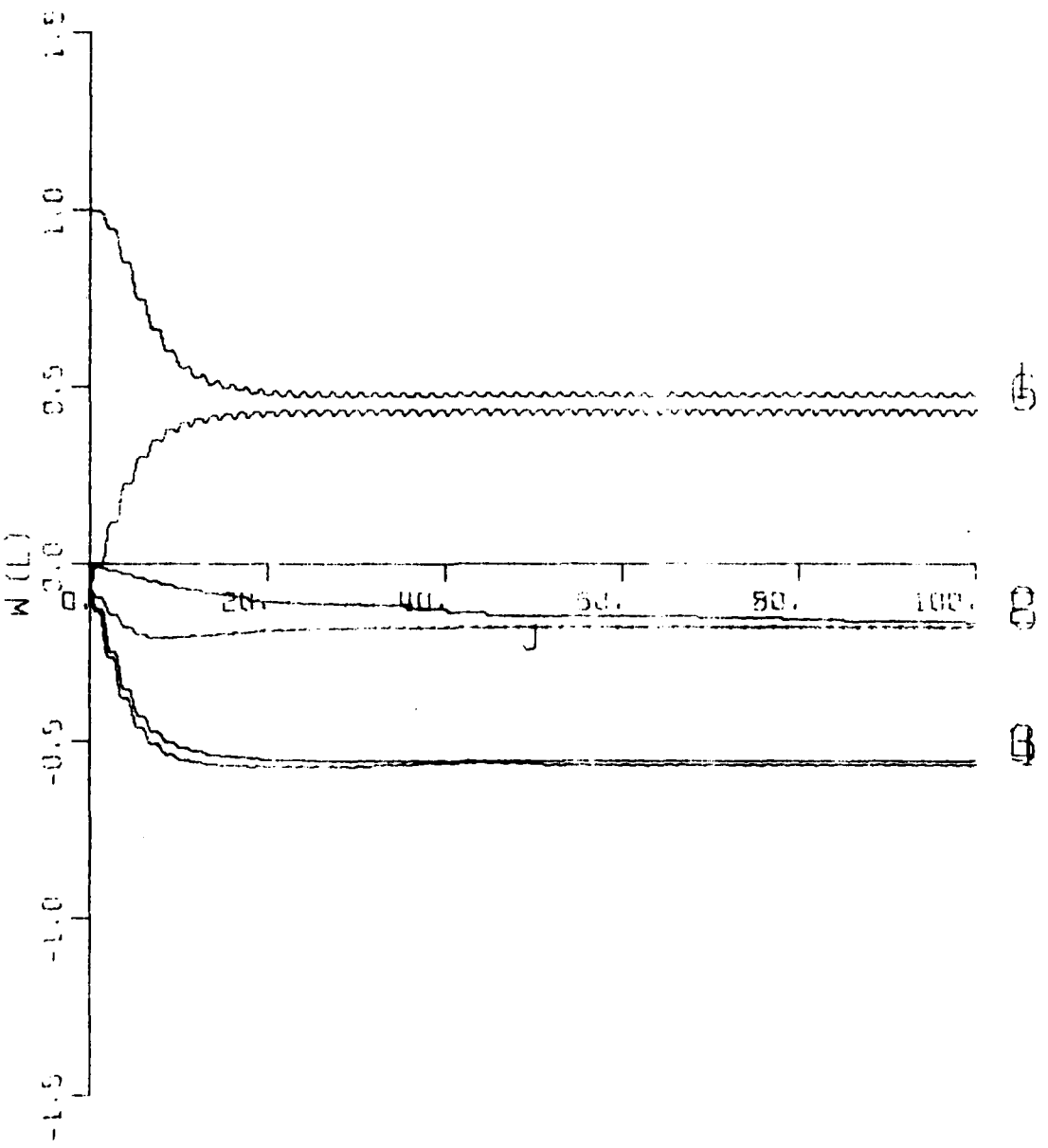


Fig. 19. Weight transients, θ=60° (tapped delay-lines).

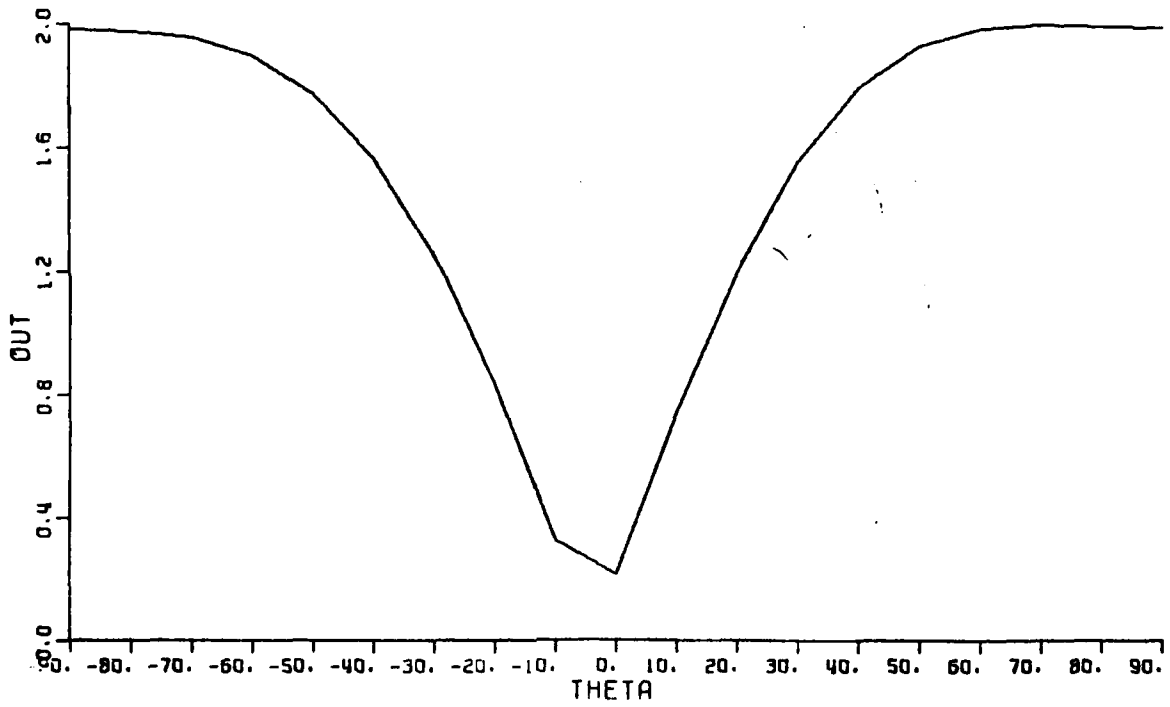


Fig. 20. Final pattern, $\theta = 60°$ (tapped delay-line).

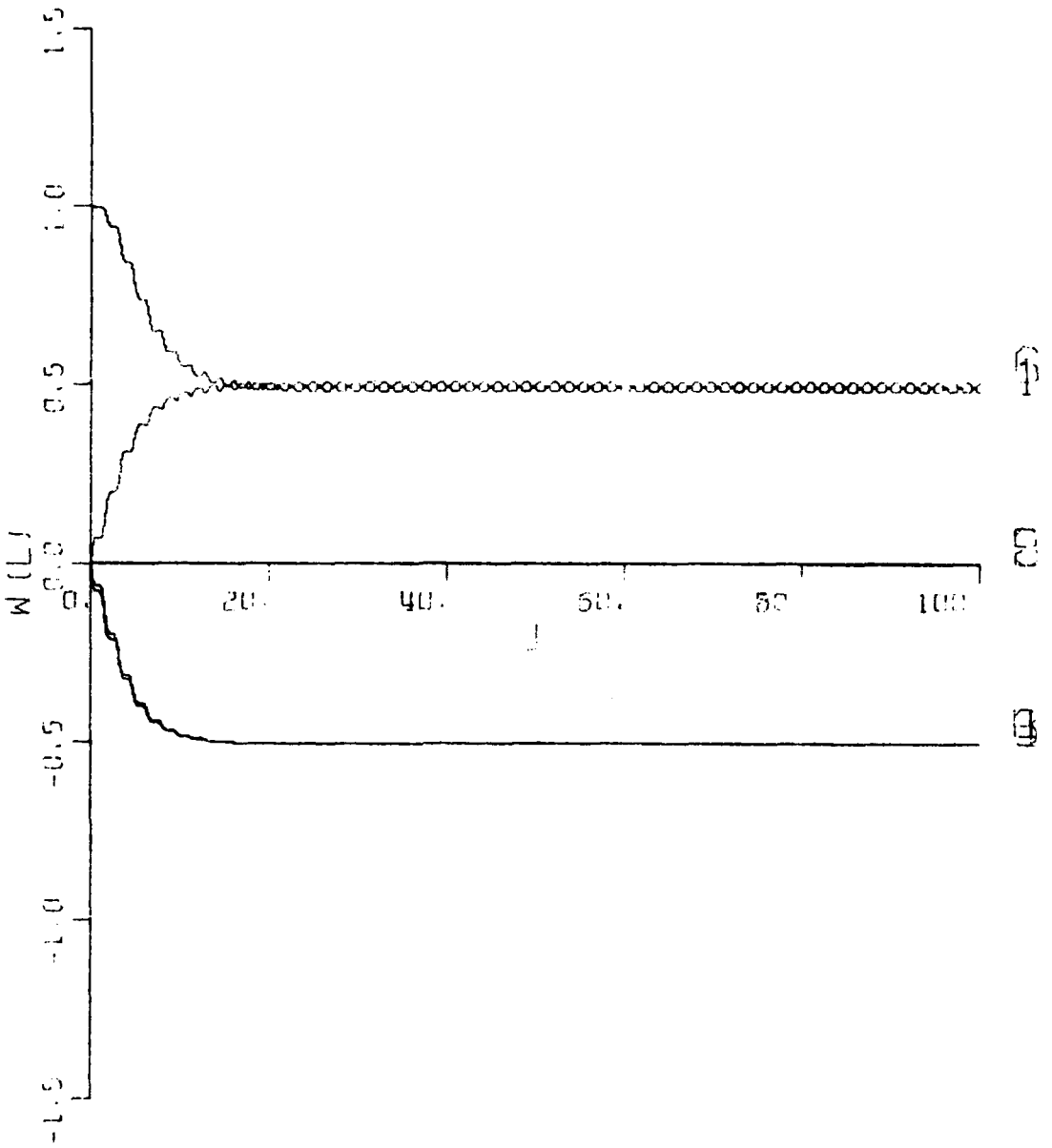


Fig. 21. Weight transients, $\theta = 90°$ (tapped delay-line).

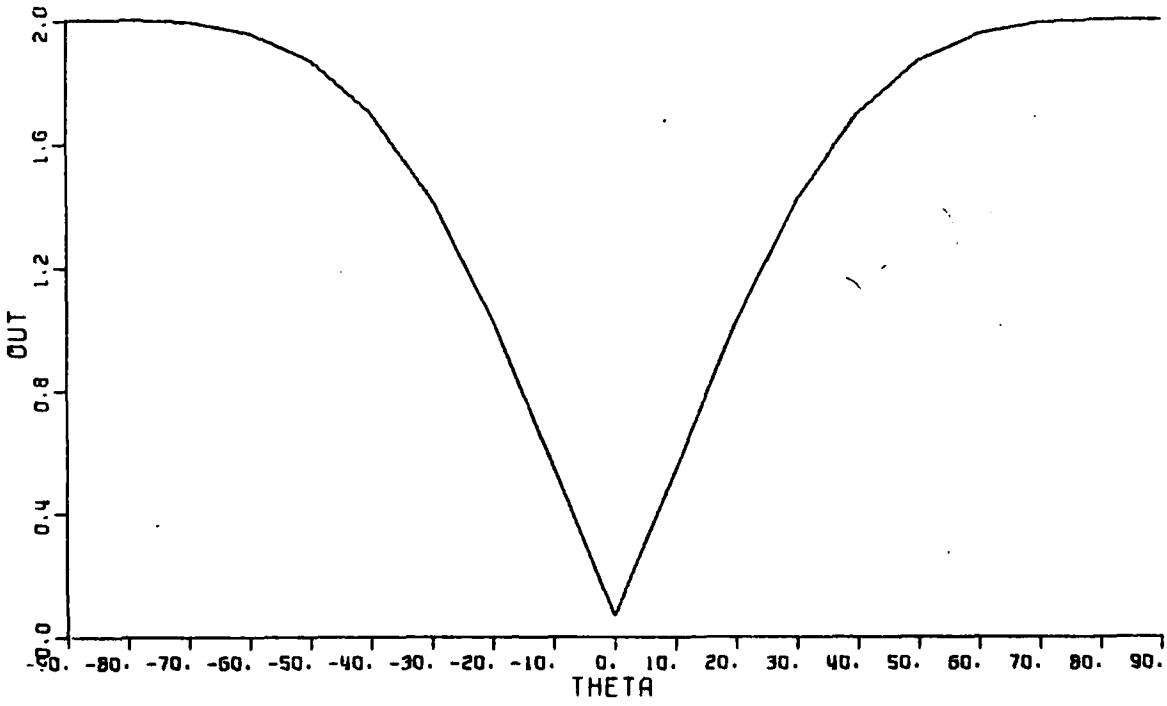


Fig. 22. Final pattern, $\theta = 90°$ (tapped delay-line).

REFERENCES

1. Compton, R. T., Jr., "Adaptive Arrays - On Power Equalization with Proportional Control, Report 3234-1, December 1971, The Ohio State University ElectroScience Laboratory, Department of Electrical Engineering; prepared under Contract N00019-71-C-0219 for Naval Air Systems Command.

2. Lee, D. W. and Compton, R. T., Jr., "The Transient Response of a Power Equalization Array with Coherent CW Signals," Report 3234-2, March 1972, The Ohio State University ElectroScience Laboratory, Department of Electrical Engineering; prepared under Contract N00019-71-C-0219 for Naval Air Systems Command.

3. Schwegman, C. W. and Compton, R. T., Jr., "Power Inversion in a Two-Element Adaptive Array," Final Technical Report 3433-3, December 1972, The Ohio State University ElectroScience Laboratory, Department of Electrical Engineering; prepared under Contract N00019-72-C-0184 for Naval Air Systems Command.

4. Schwegman, C. W., "Adaptive Arrays - Interference Suppression of Multiple Interfering Signals," Report 3576-1, December 1973, The Ohio State University ElectroScience Laboratory, Department of Electrical Engineering; prepared under Contract N00019-73-C-0195 for Naval Air Systems Command.

5. Lao, I. K. and Compton, R. T., Jr., "Power Inversion in a Tapped Delay-Line Array," Technical Report 3832-2, March 1975, The Ohio State University ElectroScience Laboratory, Department of Electrical Engineering; prepared under Contract N00019-74-C-0141 for Naval Air Systems Command.

6. Widrow, B., Mantey, P. E., Griffiths, L. J. and Goode, B. B., "Adaptive Antenna Systems," Proc. IEEE, 55, 12 (December 1967).

7. Riegler, R. L. and Compton, R. T., Jr., "An Adaptive Array for Interference Rejection," Proc. IEEE, 61, 6 (June 1973), 748.

8. Rodgers, W. E. and Compton, R. T., Jr., "Adaptive Array Bandwidth with Tapped Delay-Line Processing," Technical Report 3832-3, May 1975, The Ohio State University ElectroScience Laboratory, Department of Electrical Engineering; prepared under Contract N00019-74-C-0141 for Naval Air Systems Command.

9. Schwegman, C. W. and Compton, R. T., Jr., "An Experimental Spread Spectrum Adaptive Array," Report 3098-4, January 1974, The Ohio State University ElectroScience Laboratory, Department of Electrical Engineering; prepared under Contract N00014-67-A-0009 for Office of Naval Research.

10. Eveleigh, V. W., "Adaptive Control and Optimization Techniques," McGraw-Hill Book Co., New York, 1967.

11. Gill, P. E. and Murray, W., "Numerical Methods for Constrained Optimation," Academic Press, New York, 1974.

APPENDIX I

```
      1         DIMENSION IOUT(512),X(4),Y(4)
      2   2     FORMAT(4F10.5)
      3 C       LOOP GOTO CONSTANT
      4         PHI=0.0
      5         DO 5 KL=1,4
      6         Y=(KL-1)*20
      7         PI=3.1415926535
      8         TWOPI=6.2831853071
      9         Z=PI*SIN(Y*PI/180)
     10         WRITE(6,1)
     11   1P    FORMAT(1X,"READY TO PLOT?")
     12         READ(5,*)KG
     13         IF(KG.EQ.111)GO TO 20
     14         CALL FACTOR(.90)
     15         CALL PLOTS(IOUT,512,3)
     16         CALL PLOT(0.0,5.0,-3)
     17         CALL AXIS(0.0,0.0,1HJ,-1.5,0.0,0.0,20.0,1.0)
     18         CALL AXIS(0.0,-3.0,4HW(L),+4,6.0,90.0,-1.5,0.0,1.0,1)
     19         DO 5 L=1,4
     20         PHI=-PI/2.
     21         I(1)=1
     22         DO 6 I=2,4
     23         I(I)=0
     24   6     CONTINUE
     25         IPEN=3
     26         DO 8 J=1,21
     27         PHI=PHI+PI/2.
     28         IF(PHI.LT.1.0PI)GO TO 16
     29         PHI=PHI-TWOPI
     30   16    X(1)=COS(PHI)
     31         X(2)=SIN(PHI)
     32         X(3)=COS(PHI-Z)
     33         X(4)=SIN(PHI-Z)
     34         S=W(1)*X(1)+W(2)*X(2)+W(3)*X(3)+W(4)*X(4)
     35         DO 7 K=1,4
     36   7     A(K)=A(K)+FACL*S*(X(K)-A(K)*S)
     37         WRITE(6,2)(A(K),K=1,4)
     38         CALL PLOT((J-1)*0.025,A(L)*2.0,IPEN)
     39         IPEN=2
     40   8     CONTINUE
     41         FL=FLOAT(L)
     42         CALL NUMBER((J-1)*0.025+.2,W(L)*2.0,.2,FL,1.0,0.0,-1)
     43   5     CONTINUE
     44         CALL PLOT(12.0,-5.5,999)
     45   5     CONTINUE
     46   20    CALL EXIT
     47         END
```

APPENDIX II

```
      DIMENSION IBUF(512),X(6),W(6)
    1 FORMAT(7F10.5)
C     LOOP GAIN CONSTANT
      EMU=1.005
      A=1
      DO 5 KL=1,4
      Y=(KL-1)*90
      PI=3.1415926535
      TWOPI=6.2831853071
      Z=PI*SIN(Y*PI/180)
      WRITE(5,1)Z
   18 FORMAT(1X,"READY TO PLOT?")
      READ(5,*)NGO
      IF(NGO.EQ.111)GO TO 20
      CALL FACTI(100.)
      CALL PLOTS(IBUF,512,3)
      CALL PLOT(6.5,5.5,-3)
      CALL AXIS(0.,0.,1HJ,-1.5,.0,.0,.20,.1,.0)
      CALL AXIS(0.,-3.4HW(L),+4.F,.90,-1.5,0.5,1.,1)
      DO 3 L=1,6
      PHI=-PI/6.
      PHIM=-PI/6.
C     INITIAL SETTINGS FOR WEIGHTS
      W(1)=1
      DO 2 I=2,6
      W(I)=0
    2 CONTINUE
C     INITIAL OUTPUTS FOR ELEMENTS 1,2,4 AND 5
      X(1)=0
      X(2)=0
      X(4)=0
      X(5)=0
      IPLN=5
      DO 8 J=1,2001
      PHI=PHI+PI/2.
      IF(PHI.LT.TWOPI)GO TO 17
      PHI=PHI-TWOPI
   17 PHIM=PHIM+PI/6.
      IF(PHIM.LT.TWOPI)GO TO 16
      PHIM=PHIM-TWOPI
   16 X(3)=X(2)
      X(2)=X(1)
      X(1)=A*(1+COS(PHIM))*COS(PHI)
      X(6)=X(5)
      X(5)=X(4)
      X(4)=A*(1+COS(PHIM-Z/8.))*COS(PHI-Z)
```

```
      S=W(1)*X(1)+W(2)*X(2)+W(3)*X(3)+
     &W(4)*X(4)+W(5)*X(5)+W(6)*X(6)
      DO 7 K=1,6
7     W(K)=W(K)+EPSI*S*(X(K)-W(K)*S)
      CALL PLOT((J-1)*0.0025,W(L)*2.,IPEN)
      IPEN=2
8     CONTINUE
      FL=FLOAT(L)
      CALL NUMBER((J-1)*0.0025+.2,W(L)*2.,.2,FL,1.,0.,-1)
3     CONTINUE
      WRITE(6,1)(W(L),L=1,6),Y
      CALL PLOT(12.,-5.5,999)
5     CONTINUE
20    CALL EXIT
      END
```