STANFORD UNIVERSITY

CENTER FOR SYSTEMS RESEARCH

Research on Adaptive Antenna Techniques

FINAL REPORT

by

B. Widrow
Richard Chestek
J. R. Treichler

Information Systems Laboratory

7

Principal Investigator: Bernard Widrow (415)497-4949

Name of Contractor: Stanford University

11

Effective Date of Contract: 1 September 1975 to 28 February 1977

Short Title of Work: 6 Research on Adaptive Antenna Techniques

Amount of Contract: $51,741

12 123p.

9 FINAL REPORT

Part A:

ADAPTIVE SEPARATION OF SIGNALS IN NOISE IN TERMS OF THEIR RELATIVE POWER LEVELS

Part B:

ADAPTIVE BEAMFORMING WITH INJECTED NOISE

Part C:

A COMPARISON OF ADAPTIVE ALGORITHMS BASED ON

THE METHODS OF STEEPEST DESCENT AND RANDOM SEARCH

(Proc. IEEE paper by B. Widrow and J. M. McCool)

by

10 Bernard ~~B.~~ Widrow,
Richard Chestek,
J. R. Treichler
John M. McCool

This research was supported by the Naval Air Systems ~~Command of the~~ Department of Defense under Contract 15 N00019-76-C-0250.

406 720

INTRODUCTION

A primary goal of this research is to develop adaptive signal processing algorithms that will be useful in providing antijam A/J protection for aircraft receiving systems. Because of the motion of the aircraft and the uncertain position of the signal source, signal reception may be possible from almost any direction of incidence. The uncertainty and time variable nature of jammer positions requires rapid adaptive capability for elimination of one or more simultaneously operating jammers. Furthermore, aircraft receiving arrays generally have only a few elements, each having highly irregular and sometimes unpredictable radiation patterns. The problem is not simple.

This report is divided into three parts, each one representing a major effort contributing to adaptive A/J technology.

Part A describes single channel algorithms for separating signals based on their power levels. Jamming signals, when of concern, are generally large in amplitude. By siphoning off the strongest input components, the desired signal can be decoded from the remainder. Adaptive signal processors are proposed, analyzed, and computer simulated that have the capability of separating signals by power with controllable SNR slicing thresholds. The "ABWIN" algorithm, the first conceived, requires injection of synthetic noise of controllable amplitude (to control slicing threshold). An improved "ABWAIN" is also described. This algorithm is quieter and simpler to implement. The effects of the synthetic noise are obtained algorithmically. The analysis proves stability conditions, determines rate of convergence, and determines noise in the adaptive filter weight vector and its effects on system performance. This approach is usable for signal separation when the signals are narrowband with non-overlapping passbands. When these passbands overlap, separation would only be possible with a multichannel system connected to an array of antenna elements rather than to a single element. Development of multichannel adaptive power separators has been proposed for future work.

Part B of this report describes, simulates, and analyzes an adaptive antenna scheme that sustains (via a "soft constraint") an approximately uniform sensitivity in all directions except those corresponding to arrival directions of strong signals (which presumably are jammers). The threshold level dividing strong and weak signals is controllable. This scheme has never been tried before and appears to be quite workable and simple. Computer simulations show that several strong jammers (which may be either narrowband or broadband) can be eliminated simultaneously when the antenna array contains only a few elements. Irregularities in the individual element patterns cause nonuniformity in the overall system receiving pattern, but do not significantly reduce the system's ability to notch out strong jammers. Many analytical problems remain to be solved, such as how many jammers can be eliminated simultaneously, how deep will the nulls be vs. SNR, bandwidth, direction of arrival, what determines rate of convergence, etc. The present algorithm requires the injection of synthetic noise. A new scheme without injected noise is under development.

Part C is a reprint of a paper published in the September 1976 issue of IEEE Transactions on Antennas and Propagation. The paper describes, among other things, work on the "linear random search" algorithm. This adaptive algorithm is by no means as efficient as the LMS algorithm (in terms of noise in the solution weight vector vs. the speed of convergence), but is generally simpler to implement and can be applied to systems whose patterns are adjusted by phase shift control rather than by variable attenuators. LMS can only be used in the latter systems, not in the former. The linear random search algorithm is shown to have operational properties similar to those of a steepest descent adaptive algorithm which estimates gradient components one at a time. The random search algorithm is expected to have wide applicability and to be implementable at RF and IF frequencies. It could be applied to almost any form of adjustable system parameter, from microwave cavity paddles, to adjustable tuning stubs, to phase shifters, to attenuators, etc. Many theoretical problems remain to be solved, such as behavior in systems with multimodal performance surfaces, and derivation of relationships between system performances versus speed of convergence. This is a new algorithm. It appears to be analyzable in many circumstances. Because of its linear nature and relative simplicity compared to other random search algorithms, it may become very widely used.

Part A

ADAPTIVE SEPARATION OF SIGNALS IN NOISE IN TERMS OF THEIR RELATIVE POWER LEVELS

I. Introduction

In this section we describe a study of an adaptive device which can strip off the coherent signal component with the highest power. It has two outputs, one a filtered version of the selected component, and the other contains the total input signal with the selected component cancelled out. This device can be used alone to provide one degree of A/J protection (we assume the jammers to be powerful), or, with several in tandem, can be used to strip off and rank the various coherent signal components by power. It may also be generalized to an array configuration. This section describes the device, explores its theoretical behavior briefly, and presents some results of computer simulations of the device's performance.

II. Background

The output of a receiving antenna array can often by modeled as wideband noise plus several narrowband signal components of differing frequencies and power levels. The problem addressed in this section is that of automatically ranking the coherent components in order of their respective powers while disregarding wideband components such as noise. Such a scheme has many uses. Usually only one or a few of the signal components are useful. The others are not useful and under some circumstances may hinder detection and estimation of the desired component. A signal ranking scheme would allow the processor to deal only with the desired component(s). In other applications the ranking might provide information in itself. Generalized to an array configuration it might be used to separate signal components and identify the azimuth of each source. The basic approach uses the "Adaptive Line Enhancer [1] (ALE)" in a structure that allows it to strip off the most powerful coherent component of the input signal and pass all the rest. Similar additional stages could strip off the successively less powerful components [2]. This concept is diagrammed in Figure A-1. Modifications to the ALE's adaptive algorithm will be shown to improve the separation properties.

An inherent advantage of the ALE configuration is that it could provide two outputs. One is the input signal with the most powerful component subtracted out, thus providing the input for the next stage. The other output is a filtered version of the stripped component, allowing that component to be processed independently to find its parameters (e.g., frequency, azimuth).

The Adaptive Line Enhancer was introduced and described in reference [1]. Reference [2] describes its behavior with inputs consisting of white noise and a sinusoid. A diagram of the ALE is shown in Figure A-2.

The error signal $\varepsilon(k)$ is the difference between the input $x(k)$ and that signal delayed by Δ time units and filtered by an adaptive transversal filter. The error signal is used by the Widrow-Hoff Least Mean Square (LMS) algoirthm to adjust the weights of the adaptive fitler to minimize the error power [1]. This system's behavior is best exemplified with an input of a sinusoid plus white noise. Since the sinusoid is coherent in time it is completely predictable and a filter can be found (via the adaptive algorithm) which filters the delayed signal to provide an output $y(k)$ of the same phase. Thus a sinusoid may be successfully subtracted from the input signal and the error power minimized thereby. However, since the noise is incoherent in time there is no way that a filtered version of the delayed noise can cancel any of the input noise. Thus to minimize mean

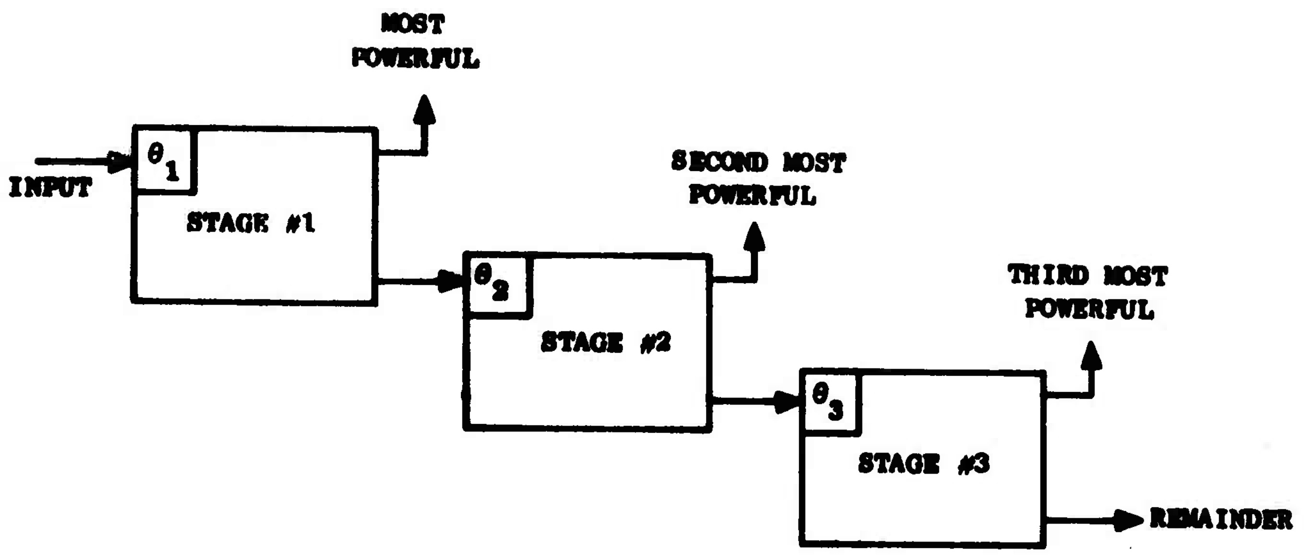


Figure 1. A Scheme for Ranking and Sorting Signals by Their Relative Powers

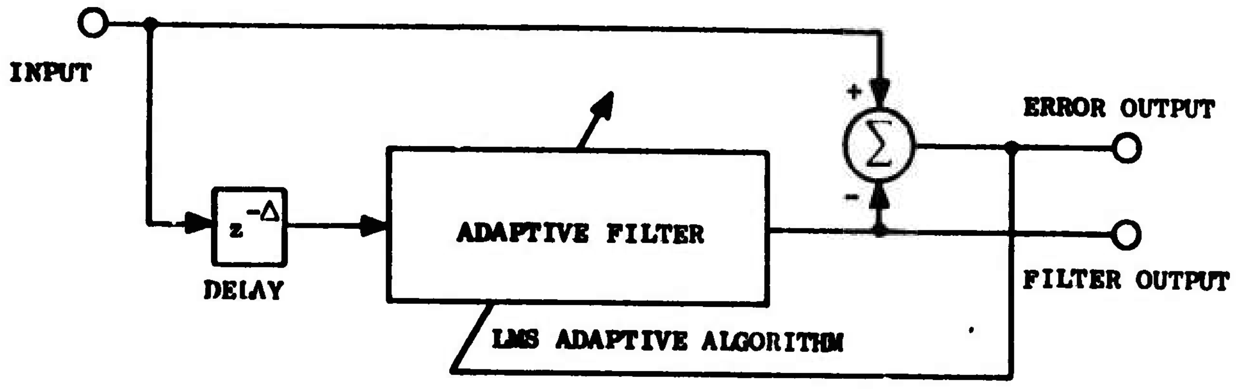


Figure 2. The Adaptive Line Enhancer (ALE)

square error the adaptive algorithm must find a filter impulse response which allows the sinusoid through but inhibits the noise as much as possible. In fact the adaptive filter found in this case is a matched filter with sinusoidal impulse response, which passes the sinusoidal component but has the smallest possible bandwidth to minimize the noise power in the filter output. The sinusoid and the noise may be viewed as adversaries to the adaptive porcess. Were the input just the sinusoid, the filter would converge to a form which had a gain of 1 at the sinusoid's frequency thereby cancelling the sinusoid altogether in $\varepsilon(k)$. If the input were white noise only, the filtered signal would actually increase the error power so the LMS algorithm turns off the filter by adjusting all the weights to zero. If the input contains both signal and noise then the adaptive algorithm must make a tradeoff to minimize the total error power.

Quarterly reports 1 and 2 [2] discuss the behavior of the ALE at some length. Two significant points were made.

1) For an input consisting of a single sinusoid of frequency f_0 and white noise, the convergent filter gain at frequency f_0 is given by:

$$a^* = \frac{\frac{n}{2} \cdot \mathrm{SNR}}{1 + \frac{n}{2} \cdot \mathrm{SNR}} ,$$

where SNR is defined as the ratio of the input sinusoid's power to that of the input white noise, and n is the number of weights in the transversal filter. A graph depicting a* as a function of SNR is shown in Figure A-3. It may be seen that $a^* \to 0$ as $\mathrm{SNR} \to 0$. Clearly the behavior of a* is a nonlinear function of SNR.

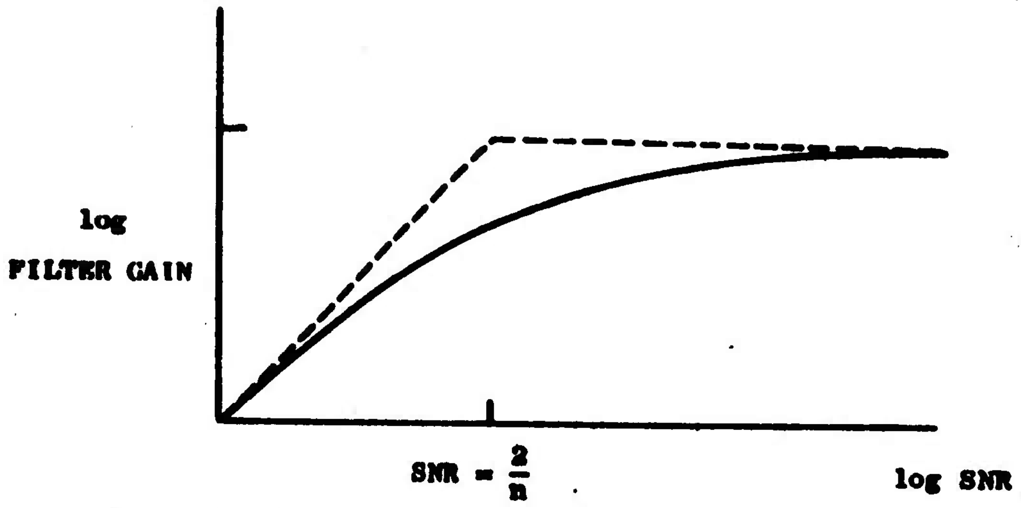


Figure 3. Filter Gain of the ALE at Convergence versus Input Power

2) If the input signal consists of coherent components which are sufficiently separated in frequency and if the number of weights n is large enough, then the convergent filter's impulse response will be the superposition of contributions from each of the coherent inputs. Each of these contributions is the same as if that coherent component were the only input. This property is called "pseudolinearity". If the input is assumed to be composed of several sinusoids plus white noise it can be shown that a_i^*, the optimal filter gain at the frequency of the ith sinusoid, is given by:

$$a_i^* = \frac{\frac{n}{2} \cdot \text{SNR}_i}{1 + \frac{n}{2} \cdot \text{SNR}_i} ,$$

where SNR_i is the ratio of the power of the ith input sinusoid to that of the total input noise. The fact that each optimal gain a^*_i is not dependent on the power of any coherent input other than the ith is a result of the ALE's pseudolinearity.

With these two principles it was shown in reports 1 and 2 [2] that the ALE can be used to strip the most powerful coherent component out of an input signal. This may be seen in the particular case where the input consists of several sinusoids plus white noise. For those sinusoidal components for which $\frac{n}{2} \cdot \text{SNR}_i >> 1$ the filter gain is approximately one and they are almost completely cancelled out of the error signal $\varepsilon(k)$. They are of course fully represented in the filter output $y(k)$. However the components for which $\frac{n}{2} \cdot \text{SNR}_i << 1$ have associated filter gains tending near zero. As a result they appear in the error signal and not in the filter output. The same is true of the broadband noise components. Thus the ALE can perform a separation of the input components on the basis of their input powers and bandwidths. The threshold of separation for sinusoids or narrowband signals is given by $\frac{n}{2} \cdot \mathrm{SNR}_i = 1$ or $P_1 = \frac{2\sigma^2}{n}$, where σ^2 is the power of the white input noise and P_i is the power of the ith sinusoid $\mathrm{SNR}_i = P_i/\sigma^2$. The threshold, denoted Θ, is a function both of the input noise power σ^2 and the number of filter weights n. Figure A-4 shows an ALE adjusted to slice off the most powerful sinusoidal component.

Almost any practical application of such level separator would require that the separation threshold Θ be adjustable over a wide range. However, as reference [2] shows, there are compelling reasons for changing neither the input noise power or the tapped delay line length n. For obvious reasons the input noise power cannot be decreased. It can be increased artificially however by adding extra white noise to the ALE input, but this has the disadvantage that the extra noise propagates on through the error signal into successive stages. Changing the filter length n changes both the filter dynamics and the number of components which can be handled simultaneously and independently. Usually the user would desire to have these parameters remain constant. To allow alteration of the threshold level without changing either σ^2 or n, an alternate separator was suggested [2]. This processor, the ALE with injected noise (ALEWIN), diagrammed in Figure A-5, provides for the variable threshold of $\Theta = \frac{2(\sigma^2+\sigma_A^2)}{n}$, leaving σ^2 and n fixed. This is accomplished by adding extra white noise of power σ_A^2 into the adaptive filter input. The added noise decreases the apparent SNRs of the coherent input components and therefore modifies the power slicing level. Figure A-6 shows the optimal filter gain as a function of SNR_i for the ALEWIN. The term SNR_i' is defined by $P_i/(\sigma^2 + \sigma_A^2)$.

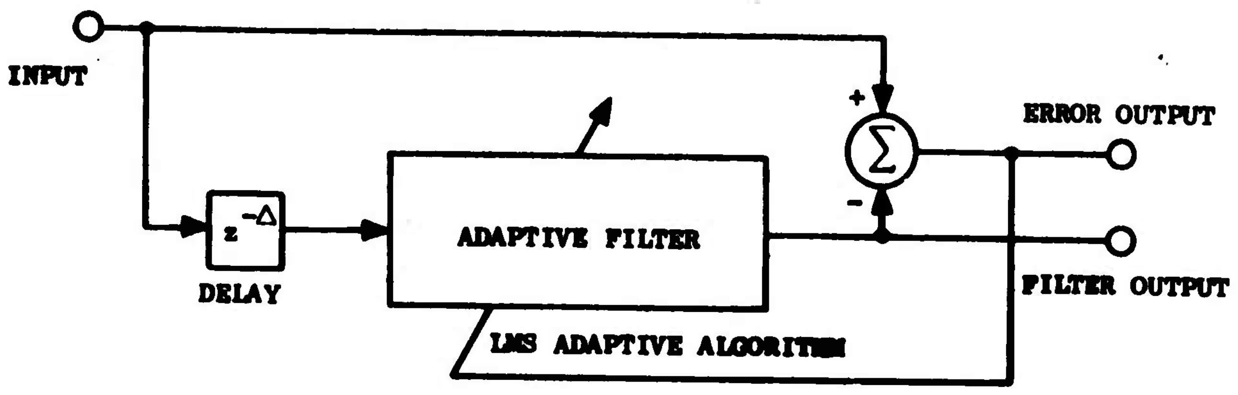


Figure 4. The ALE Configured to Strip Off the Most Powerful Component

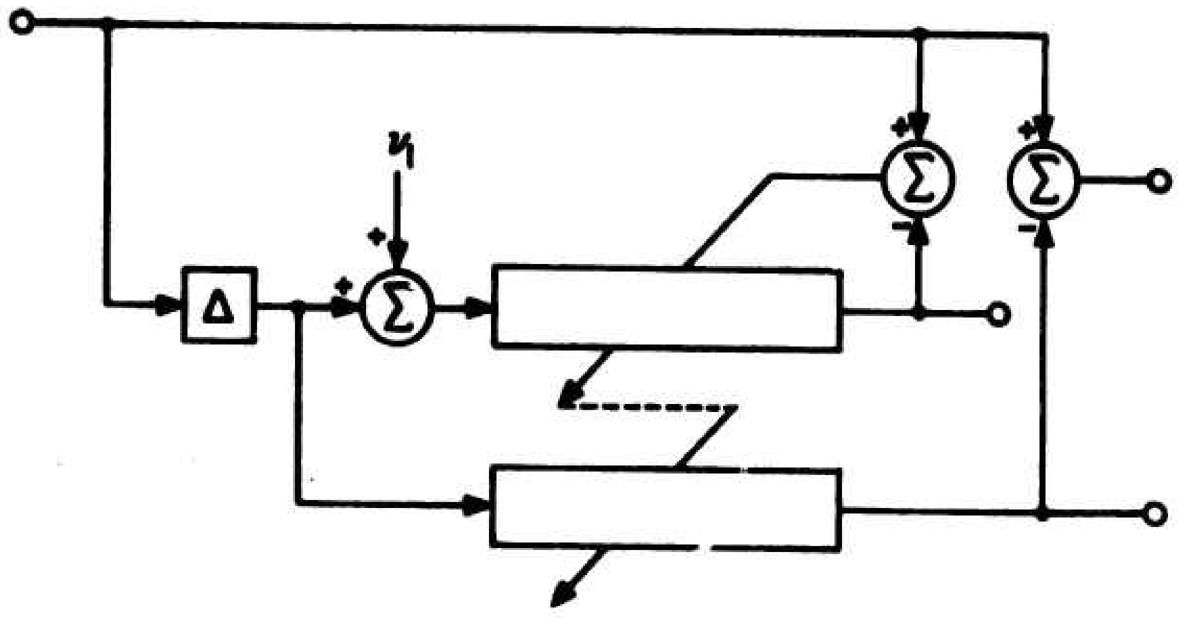


Figure 5. The Adaptive Line Enhancer With Injected Noise (ALEWIN)

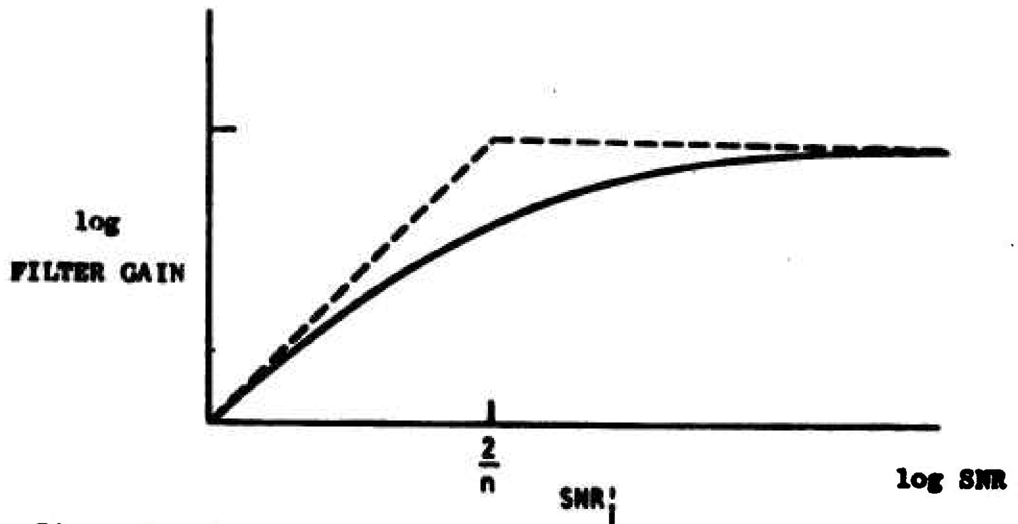


Figure 6. Convergent Filter Gain of the ALEWIN for the ith Signal Component versus its SNR.

This new scheme has the disadvantage however that the added noise propagates through the filter, into the error signal, and into succeeding stages. It is not as bad as would be the case if the noise were injected into the ALE input, but still the effect is undesirable. This problem can be modified to some extent by adding a "slave" filter which filters a delayed version of the input but without its added noise. This slave filter uses weights copied from the adaptive filter. Since no noise is actually added into its input, the slave filter output and the error signal formed with it are devoid of the direct effects of the injected noise. This method is practical and may have application in some situations. Zahm [6], for example, has suggested its use for the suppression of strong jamming in an adaptive beamformer without obliteration of desired weak input signals. However this method still has a major flaw in that the injected noise increases the adaptive filter's "misadjustment." The adaptive algorithm which determines the weights of the adaptive filter produces errors or noise in its estimates of the optimal weights. Weight noises are a function, among other things, of the input noise. In a normal well-designed adaptive processor, weight noises are tolerably small and weight errors are not a problem. However in this case the large amount of injected noise can cause large and bothersome noises in the weights, and cause significant amounts of random modulation of the filter output.

Further research has shown that this problem can also be solved. By appropriately modifying the adaptive algorithm used to adjust the filter weights, behavior similar to that caused by the injected white noise can be attained. In addition the modified algorithm does not cause the increased misadjustment that the injected noise does, nor does it require the slave filter. It has been dubbed the "ALE with algorithmically injected noise" (ALEWAIN), and its operation is clearly superior to that of the ALEWIN [2]. The next section will describe the mathematical formulation of the ALEWAIN and show its relationship to the ALE and ALEWIN.

III. Theoretical Motivation for the ALEWAIN

This section will provide motivation for the ALEWAIN configuration by first analyzing the mechanism of filter optimization for the ALEWIN and then showing that in expectation the same effect can be achieved by modifying the algorithm. To proceed, some definitions and background are required.

Figure A-7 is a block diagram of the ALEWIN. The input signal, $x(k)$, passes into two paths, one directly to a differencing circuit, and the other through a decorrelating delay of Δ time units, through a tapped delay line filter, and into the other input of the differencing circuit. This difference $\varepsilon(k)$, termed the error signal, is used by the adaptive algorithm to adjust the filter in such a way as to minimize the expected power of $\varepsilon(k)$. The following definitions allow this to be put into mathematical form:

$x(k)$ = the input signal

$n(k)$ = the noise injected into the filter input

$f(k)$ = the actual input into the filter $[=x(k-\Delta)+n(k)]$

$F(k) = [f(k)\ f(k-1)\ \ldots\ f(k-n+1)]^T$ = samples of filter input in the tapped delay line

$W(k) = [w_0(k)\ \ldots\ w_{n-1}(k)]^T$ = the impulse response of the tapped delay line filter (also called the "weight vector")

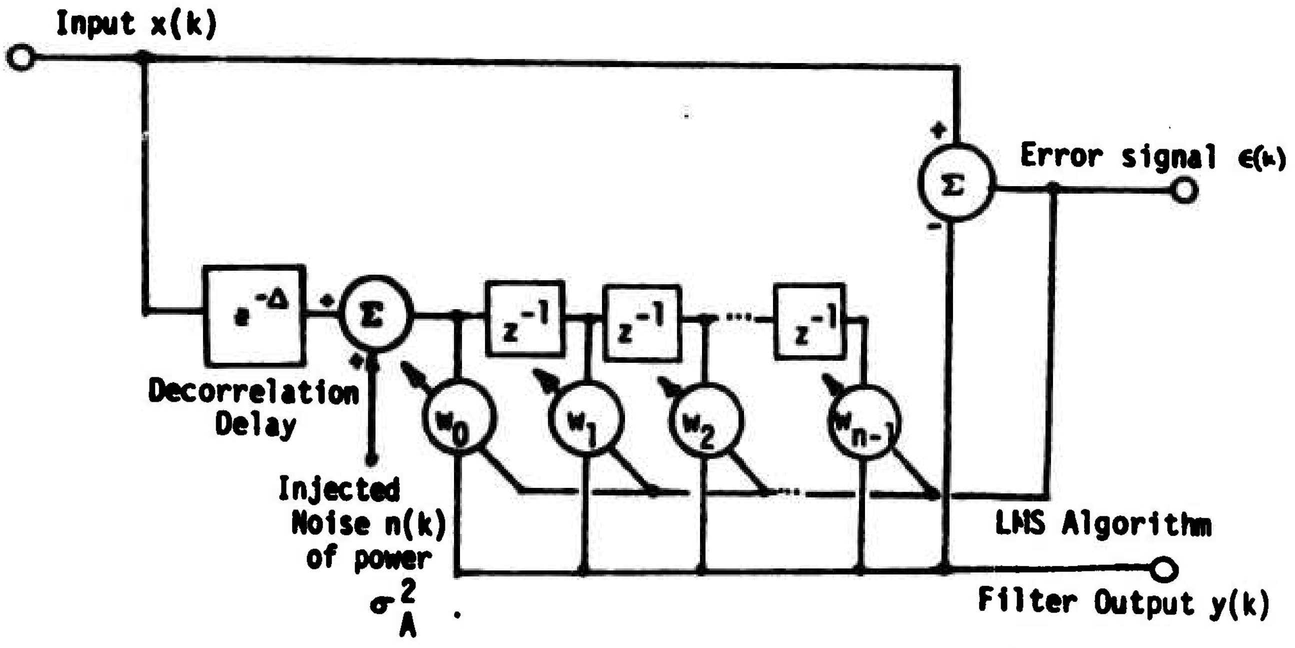


Figure 7. The Structure of the Adaptive Line Enhancer with Injected Noise (ALEWIN)

$y(k)$ = filter output = $W^T(k)F(k)$

$\varepsilon(k)$ = the error signal = $x(k) - y(k) = x(k) - W^T(k) \cdot F(k)$

It will be assumed that $x(k)$ and $f(k)$ are random variables which are statistically independent, stationary and zero-mean. The autocorrelation function of $x(k)$ is $r_x(\tau)$ and that of $n(k)$ is $r_n(\tau)$. The injected noise, $n(k)$, is assumed white. Therefore $r_n(\tau) = \sigma_A^2\delta(\tau)$, where σ_A^2 is the power of the injected noise process.

The filter impulse response (the weight vector $W(k)$) is iteratively adjusted toward its optimum value by the Widrow-Hoff LMS algorithm, which, in expectation, will reduce $\varepsilon^2(k)$ to its minimum value. The LMS algorithm updates its estimate of the optimal weight vector at each sample interval by making an instantaneous estimate of the gradient of the error surface and then moving toward the minimum. Mathematically this may be written:

$$W(k+1) = W(k) + \mu \cdot \varepsilon(k) \cdot F(k), \quad W(0) = W_0 , \qquad \text{III-1}$$

where $\varepsilon(k) \cdot F(k)$ is the magnitude of the instantaneous gradient estimate and μ, the adaptation constant, determines how much the weight vector will be changed in response to that estimate.

By making some substitutions this recursion equation can be written in another useful form. Note that by definition the error is given by $\varepsilon(k) = x(k) - F^T(k) \cdot W(k)$. Substituting this into equation III-1 and collecting terms in $W(k)$:

$$W(k+1) = [I - \mu F(k) \cdot F^T(k)] \cdot W(k) + \mu \cdot x(k) \cdot F(k), \; W(0) = W_0 \qquad \text{III-2}$$

From this equation $W(k)$ can be computed iteratively given only the input signal and injected noise.

Suppose the expected value of the weight vector were examined. This would represent the average behavior of the ALEWIN in the statistical sense. If the expected values of both sides of Eqn. III-2 are taken and if it is assumed that $F(k)\cdot F^T(k)$ and $W(k)$ are uncorrelated,* this can be done.

$$E[W(k+1)] = [I-\mu R_F]\cdot E[W(k)] + E[x(k)\cdot F(k)],\ E[W(0)] = W(0) = W_0 \qquad \text{III-3}$$

where $R_F = E[F(k)\cdot F^T(k)]$, the autocorrelation matrix of the process $F(k)$. The tapped delay line data vector, $F(k)$ can be written as the sum of $X(k)$, the vector representing the component due to the input signal, and $N(k)$, the component due to the injected noise. Further $x(k)$ and $n(k)$ have been assumed independent. Therefore $E[F(k)\cdot F^T(k)]$ becomes $E[X(k)\cdot X^T(k)]+E[N(k)\cdot N^T(k)] = R_x + R_n$, the sum of the two autocorrelation matrices for the two separate processes. By the definition of the autocorrelation matrix the ij<u>th</u> element $[R]_{ij} = r(i-j)$. Since $r_n(\tau) = \sigma_A^2\delta(\tau)$ this implies that $R_n = \sigma_A^2\ I$, where I is the identity matrix. The input signal matrix R_x is not in general diagonal.

The same facts as above may be applied to the evaluation of the driving term $E[x(k)\cdot F(k)]$. Since $F(k) = X(k) + N(k)$ and since $x(k)$ and $n(k)$ are independent then the term becomes $E[x(k)\cdot X(k)] = P_x$, the autocorrelation vector of the process $x(k)$. The i<u>th</u> element of P_x is given by $[P_x]_i = r_x(\Delta+i+1)$.

*Assuming that the transition term $F(k)F^T(k)$ and the weight vector $W(k)$ are uncorrelated is a common simplifying assumption in work on adaptive signal processors [1,5,7]. It is an excellent assumption when the constant μ is small enough that adaptation is slow.

With these observations equation III-3 may be simplified to the following:

$$E[W(k+1)]=[(I-\mu(R_x+\sigma_A^2\cdot I))]\cdot E[W(k)] + \mu P_x,\ E[W(0)] = W_0 \qquad \text{III-4}$$

This equation then describes the expected behavior of the weight vector of the ALEWIN. In fact if this equation is solved to find the convergent behavior of the algorithm and if assumptions of the sort used in a previous report [2] are applied, this equation yields the formulas obtained in this report by the Parseval's Theorem approach. Furthermore if x(k) is assumed to contain only white noise and sinusoidal components and if σ_A^2 is set to zero, then Eqn. III-4 converges to the functional forms described in reference 3. Thus equation III-4 describes the operation and behavior of both the ALE and the ALEWIN.

With this background, the modified algorithm may be introduced. Suppose that the system used is exactly as in Fig. 1 except that there is no injected noise. If so then F(k) = X(k) and the recursion expression for the expected weight vector would be:

$$E[W(k+1)] = [I-\mu R_X]\ E[W(k)] + \mu P_x,\ E[W(0)] = W_0 \qquad \text{III-5}$$

As pointed out in the previous paragraph this is simply the weight vector recursion for the ALE, the limiting case of the ALEWIN as σ_A^2 approaches zero. However suppose that instead of the standard LMS algorithm, another adaptive algorithm (the "leaky" LMS algorithm) is used. Suppose the weight vector update equation is given by:

$$W(k+1) = \gamma\cdot W(k) + \mu\cdot\varepsilon(k)\cdot X(k),\ W(0) = W_0 \qquad \text{III-6}$$

where $\gamma > 0$. For this work γ will also be assumed to be less than or equal to 1. The action of this algorithm at each sample instant to add in the new instantaneous estimate of the error surface gradient but also to diminish the weight vector by a small factor (causing it to "leak"). The rationale for this will be discussed later.

Suppose that the weight vector equation of Eqn. III-6 is applied to the ALE configuration. Making the appropriate changes to Eqn. III-5 then becomes:

$$E[W(k+1)] = \gamma[I-\mu R_x]\, E[W(k)] + \mu P_x,\ E[W(0)] = W_0, \qquad \text{III-7a}$$

or, $$E[W(k+1)] = [\gamma I-\gamma\mu R_x]\, E[W(k)] + \mu P_x,\ E[W(0)] = W_0, \qquad \text{III-7b}$$

or, $$E[W(k+1)] = [I-\mu[\frac{(1-\gamma)}{\mu} I+R_x]]\, E[W(k)] + \mu P_x,\ E[W(0)] = W_0. \qquad \text{III-7c}$$

But Eqn. III-7c is exactly the same form as that of the normal LMS implementation of the ALE (compare with Eqn. III-5) if $[\frac{(1-\gamma)}{\mu} I + R_x]$ were interpreted as the autocorrelation matrix of the tapped delay line data vector. However, by comparison with Eqn. III-4, it may be seen that this is exactly the recursion expression for the expected value of the ALEWIN weight vector if $\frac{(1-\gamma)}{\mu} I = R_n$. But $R_n = \sigma_A^2 I$; therefore, if γ is chosen so that $\gamma = 1 - \mu\sigma_A^2$ then the ALE configuration with the modified (leaky) adaptive algorithm and no injected noise would have the same expected weight vector as the ALE with injected noise of power σ_A^2 using the normal LMS adaptive algorithm. Thus modifying the adaptive algorithm has the same effect as injecting noise into the filter input. The line enhancer driven by the leaky LMS algorithm is called ALEWAIN.

IV. Discussion

The previous section shows that the ALEWAIN gives the same mean weight vector as that of the ALEWIN (but not the same variance in the weight vector). Whatever value of σ_A^2 that would have been chosen to separate the most powerful coherent component with the ALEWIN can be related to the proper value of $\gamma(=1-\mu\sigma_A^2)$ which will allow the ALEWAIN to achieve the same effect in the mean. However, since no noise is actually injected in the ALEWAIN, it significantly outperforms the ALEWIN and is cheaper to implement. The ALEWIN requires the actual injection of noise into the filter input. Therefore the filter output $y(k)$ and the difference (error signal) $\varepsilon(k)$ are noiser than they would be if only the input signal were driving the adaptive algorithm. Since both the filter output and the difference signal are desired outputs of the power separator, this extra noise is deleterious. This problem was partially eliminated in previous work [1] by using a duplicate filter, the so-called "clean" filter, whose impulse response is copied from the main filter. Since no noise is injected into its input, the output and ensuing difference signal are not corrupted by the injected noise. But its weights are noisier than need be because the main filter weights are made more noisy by the injected noise. In addition, the use of the clean filter increases by 50% the number of multiplications required for each iteration of the filter. With the ALEWAIN the duplicate filter is not needed since no noise is actually injected. Furthermore, the weights of the ALEWAIN are less noisy for the same speed of convergence. Look again at the adaptive update scheme for the ALEWIN. The estimate of the gradient of the error surface is $-\varepsilon(k)\cdot F(k)$. Suppose the ALEWIN has converged. If so then the expected value of the gradient is zero. If the weight vector were driven by the true gradient, then the weight vector would be unchanging at convergence. However it is actually driven by an instantaneous estimate of the gradient. The more noise that is injected then the more $-\varepsilon(k)\cdot F(k)$ will differ instantaneously from its expected value of zero at convergence. Thus the weight vector will not stay at its optimal value but will be perturbed away. This is the manifestation of weight noise mentioned earlier and in the case of the ALEWIN, it increases with the value of σ_A^2. Since the ALEWAIN configuration injects no noise, its weight noise is a function only of the input signal as well as γ and the adaptation constant μ.

Some insight into the equivalence of the ALEWIN and ALEWAIN can be gained by discussing the effect of the injected noise on the algorithm. It was shown [1] that the effect of adding noise to the filter input was to decrease the effective SNR of each input component. Since the optimal gain $a^* = \frac{SNR_i'}{1+SNR_i'\cdot n/2}$ is monotonically decreasing with decreasing SNR_i', then increasing the injected noise has the effect in expectation of reducing the contribution in the weight vector from the less powerful component. In terms of the weight vector adjustment algorithms, Eqn. III-4, the expected weight vector for the ALEWIN may be compared with Eqn. III-5, the expected weight vector for the ALE. It may be seen that in expectation they both have the same driving term μP_x. Thus the injected noise contributes nothing to the driving term. The only place the injected noise appears is in the transition term. It serves only to decrease the magnitude of the weight vector at each iteration. The higher the injected noise, the greater is the decrease in the weight vector. Since $n(k)$ is white, then in expectation all weights are decreased equally. Thus P_x tends to increase the magnitude of the weight vector and R_n tends to decrease it. The terms vie, on the basis of power, to determine the convergent weight vector magnitude. The vector P_x must be large to compensate for the decrease in the weight vector caused by R_n.

The ALEWAIN performs exactly the same function by altering the adaptive algorithm so that it deterministically decreases the weight vector at each iteration rather than relying on the statistical effects of the white injected noise. To remain fully represented in the weight vector, each input component must be strong enough to counteract the effects caused by γ.

This modified algorithm has been termed leaky LMS (LLMS) since if $\varepsilon(k) \cdot X(k) = 0$ then the weight vector tends to leak away to zero as k tends to infinity. It is also a good model of analog implementations of the LMS algorithm where imperfect (leaky) integrators are used. A distinction should be drawn here. In the case of analog integrators, the "leakiness" is an undesirable feature and much design effort goes into trying to minimize it. However, the work in this report shows that a controlled amount of leakiness can have a desirable effect in the application of signal separation by power level. Another important observation is that leaky LMS does not minimize the mean square error. It does find a Wiener solution but for a performance function corresponding to an input which contains an artificial additive noise term. Weak input signals excluded from the filter output y(k) because of their lesser powers.

IV. Amplitude Correction Via a One-Weight Noise Canceller

Reference 2 shows the results of computer simulations of the ALEWAIN. These simulations demonstrate that the ALEWAIN can in fact slice off the most powerful sinusoid of an input signal and that its performance is clearly superior to that of the ALEWIN. However, for the ALEWAIN to function well in the power separation scheme shown in Figure A-1, not only must the most powerful component be isolated in the filter output, but it must also be completely removed from the error signal. If this signal is not completely extracted then some following stage might attempt to isolate the residual rather than the next lower-powered signal. Unfortunately it can be shown that the power ratio of the most and second most powerful signals must be infinite to allow complete separation in one ALEWAIN stage. To achieve complete separation with a finite ratio, an additional modification can be made. This modification uses a single-weight Adaptive Noise Canceller (ANC) [3].

A block diagram of the adaptive noise canceller is shown in Figure A-8. The ANC has two inputs, the primary which contains the desired signal plus some corruptive influence, and the reference input which contains noise which is correlated with the corruptive influence in the primary signal. By adaptive filtering the reference signal, noise may then be subtracted from the primary signal to reduce the effect of the corruptive influence. The use of an adaptive filter makes this noise subtraction possible and practical by changing the adaptive filter weights to best adjust the spectral content and phase of the reference signal, even if the character of the signal and the undesired corruption change with time.

Using this concept, the problem of complete extraction of the most powerful component can be dealt with. Refer to Fig. A-9. The input signal

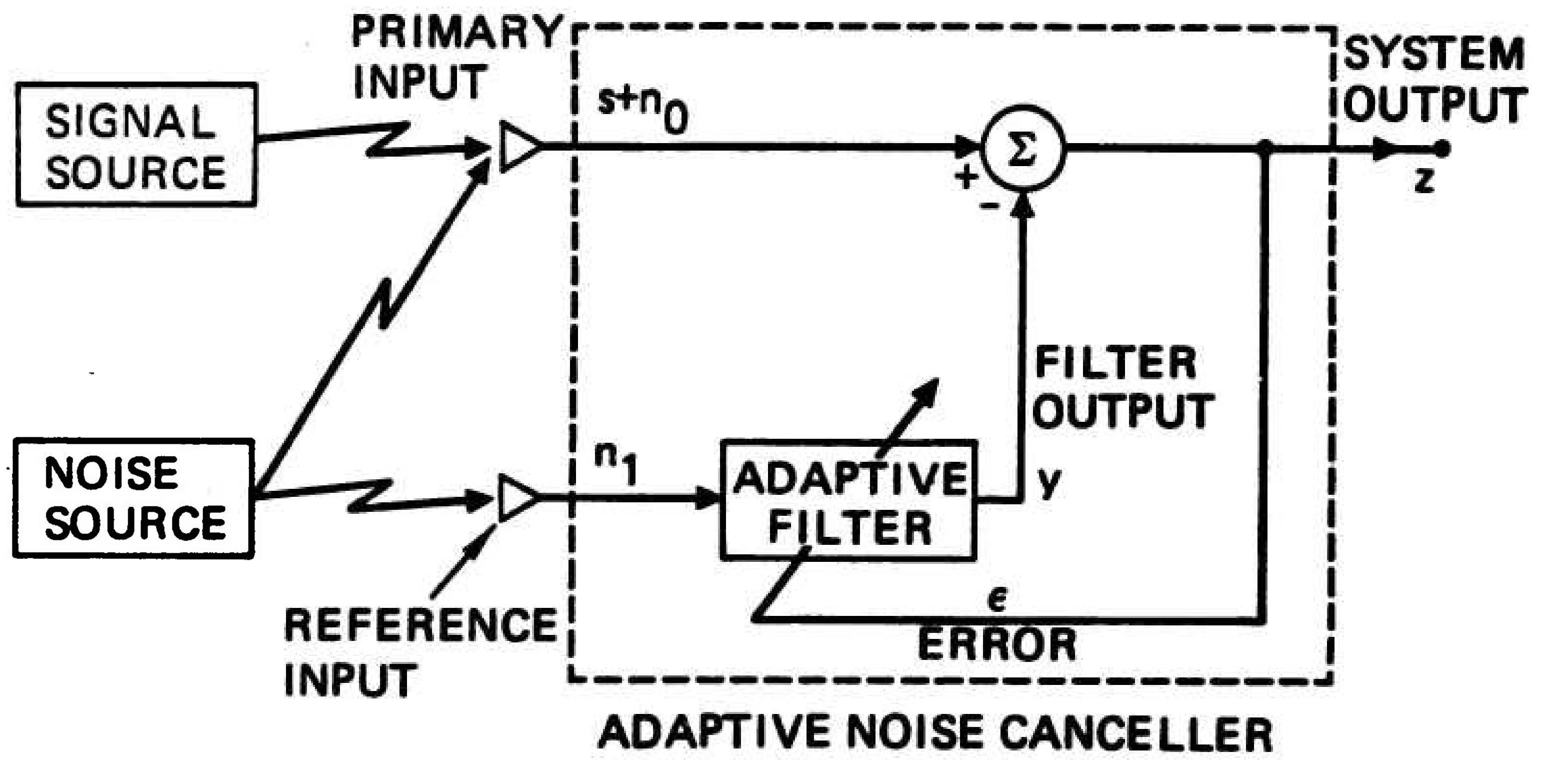


Fig. 8 The Adaptive Noise Canceller

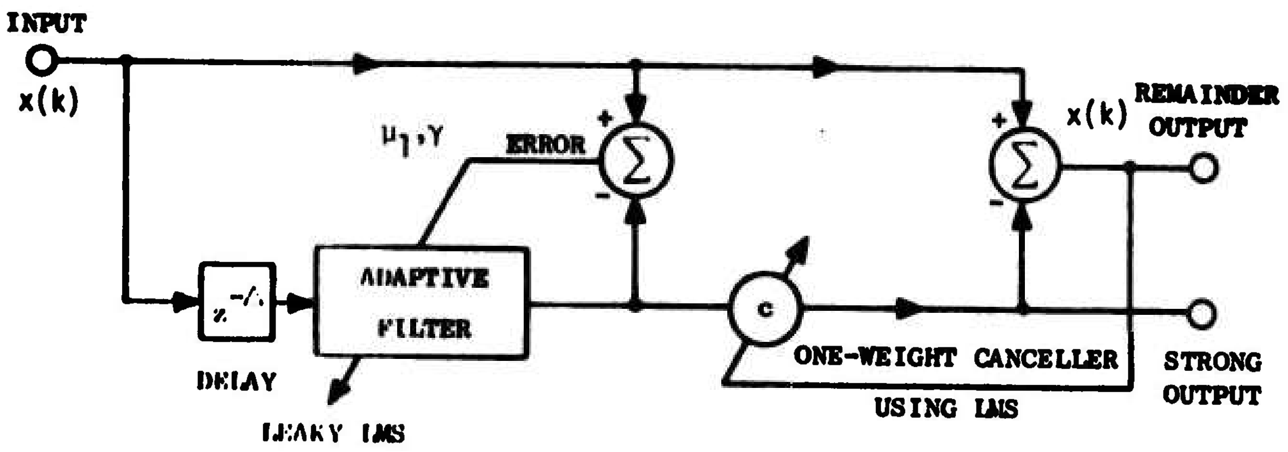


Figure 9. ALEWAIN Modified with One-Weight "Noise Canceller" to Effect Complete Cancellation of Most Powerful Component

x(k) forms the primary input. It may now be viewed as the sum of many desirable components plus an undesired corruptive signal, the most powerful coherent component. The reference signal, a correlated version of this component, can be supplied by the ALEWAIN filter output y(k). This concept is illustrated in Figure A-9. In the particularly useful case in which the most powerful component is sinusoidal, only a single adaptive weight is required in the noise canceller stage. This is a result of the fact that the sinusoid in the ALEWAIN output is in phase synchronism with the most powerful sinusoid in the input signal. Because only the gain (and not the phase) must be adjusted, only one adaptive weight is required.

This concept may be viewed in another way. To perform signal separation with the ALEWAIN it must do several things:

1) Identify the most powerful component
2) Isolate the most powerful component (i.e. form a filter to pass it)
3) Adjust the gain of the most powerful signal to provide complete cancellation,
4) And, minimize the gain of the filter for less powerful components.

Unfortunately points 3) and 4) are contradictory. Attempting to make the gain of the adaptive filter equal to one at the frequency of the most powerful component has the simultaneous but undesirable effect of increasing the gain for the less powerful components. The ALEWAIN with the NC modification allows the ALEWAIN to perform functions 1, 2, and 4, while the noise canceller stage adjusts the isolated component to have unity gain for ideal cancellation from the input signal. As will be shown in Section V the ALEWAIN with tandem one-weight adaptive noise canceller (denoted AKEWAIN + NC) performs very well with narrowband inputs.

V. Experiments Results

To demonstrate the performance of the ALEWAIN + NC configuration, it was simulated on an HP 2116B minicomputer. An experiment was designed to test the ability of the ALEWAIN + NC to successfully strip off the coherent components of an input in order of their power. The input signal was composed of three sinusoids of different frequencies and powers plus white noise of unit variance. The powers and frequencies of these sinusoids are as follows:

	Frequency (Hz)	Power
Sinusoid #1	179.	78.625
Sinusoid #2	312.5	3.125
Sinusoid #3	607.0	0.125

Forty-eight hundred samples of the input signal were stored in a disk file, serving as input data for the program which simulates the ALEWAIN + NC. The program operates by taking input samples from a designated file and storing the resulting noise canceller filter and difference outputs in additional disk files. In this manner the same program can be used to provide several stages of separation by simply specifying the input file of the current run to be the difference signal from the previous run. In such a fashion the input signal was subjected to three levels of slicing. At each level the length of the ALEWAIN filter n and the decorrelation delay Δ were held constant (64 and 1, respectively). To achieve separation within the desired number of iterations, however, μ_1, μ_2, and γ were varied on the three runs. The values actually used are as follows:

	μ_1	ECT_A	μ_2	ECT_{NC}	γ	Equv. σ_A^2
Run #1	1.6×10^{-6}	500	2×10^{-4}	500	.999875	78.125
Run #2	2×10^{-5}	1000	2.5×10^{-3}	1000	.9999375	3.125
Run #3	10^{-3}	1000	0	-	1	0

The abbreviation ECT stands for "estimated convergence time," expressed in number of iterations. The column to the right of that for γ shows the value of σ_A^2 which would be required by the equivalent ALEWIN. The strategy used for the choice of these operating parameters is discussed in Section V.I.

In actual practice, all the cascaded adaptive processors might be allowed to begin adaptation simultaneously. For this simulation however each cascaded processor was not allowed to adapt until the preceding processor had reached convergence. The estimated convergence times were used to determine the startup times. As noted before, this is not mandatory in practice. It is done here to uncouple the transient nature of each stage from all the others in order to study these adaptive transients.

Figure A-10 is a plot of 512 points of the input signal and the associated power spectrum. The format of this figure will be used several times so it will be explained in detail here. Part (a) of the figure presents a 512-point record of the signal of choice (in this case, the input to the first separator stage). The middle curve is the magnitude squared of the DFT of the data record. This plot is scaled by its maximum value. The bottom curve is the logarithm of the middle plot. While taking the logarithm presents a less spectacular picture than the linear plot it makes the second order effects such as input noise and the effects of filter weight noise more visible. The arrows below the frequency axes indicates the frequencies of the input sinusoids. They will be shown on all such spectrum plots. Note thet the power relationship between all the coherent

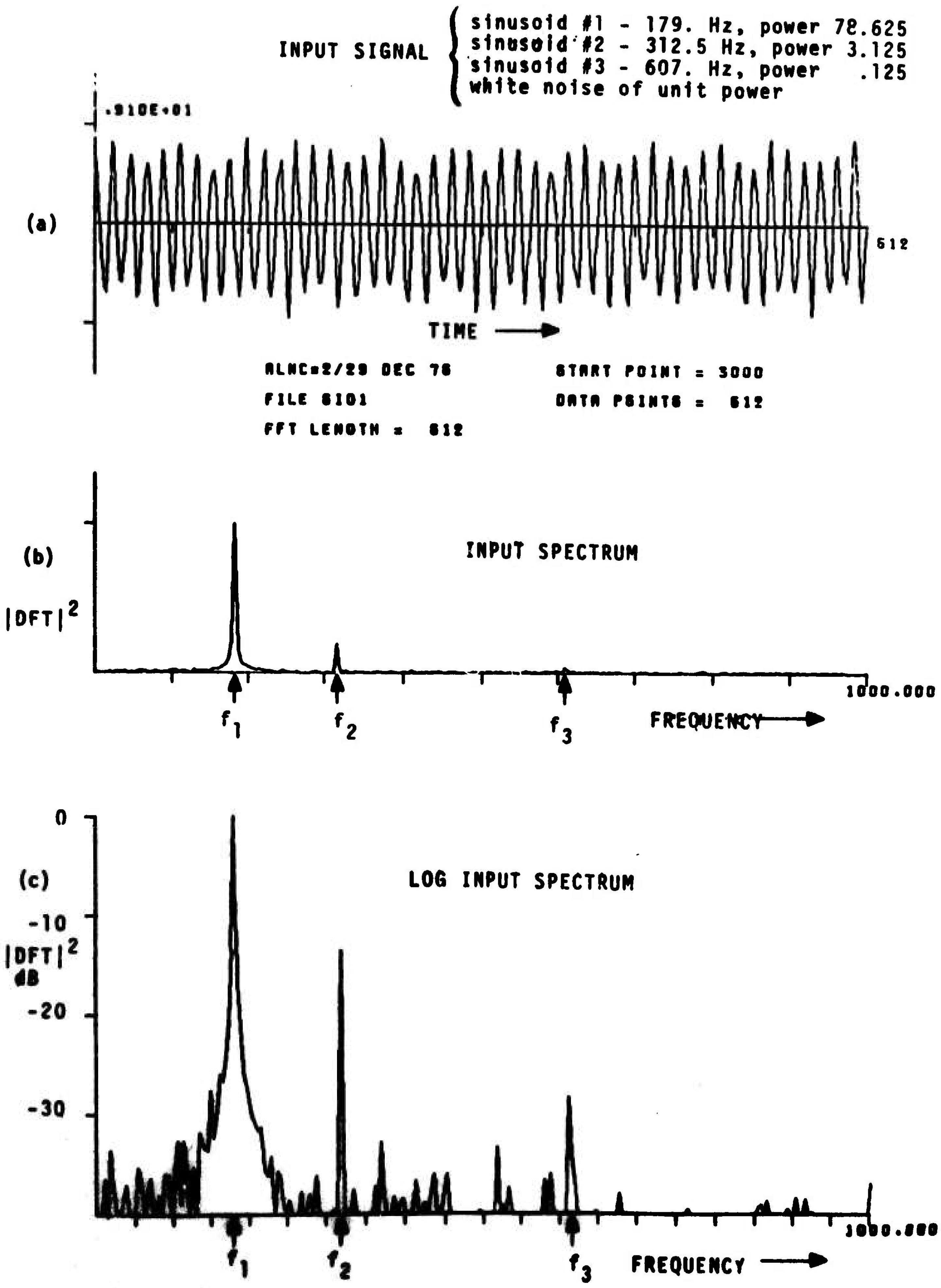


FIGURE A-10. INPUT SIGNAL AND SPECTRUM FOR ALEWAIN+NC SEPARATION EXPERIMENT input components is visible in Figure A-10c.

Figure A-11 shows the filter output of ALEWAIN #1 at the beginning of adaptation. The exponential growth of its envelope is just as predicted by other work on the ALE structure [1]. Figure A-12 shows the filter output signal and its spectrum well after convergence (iteration 3000). From its spectrum it is clear that the most powerful sinusoid has been essentially isolated.

Figure A-13 shows the strong output of stage #1. Adaptation begins at iteration 500 (after ALEWAIN #1 converges) and converges in roughly 300 iterations to provide the proper scaling for sinusoid #1. That this scaling has been properly found is illustrated in Figure A-14. This is the remainder output of stage #1. At iteration 500 (when its adaptation begins) the contribution from sinusoid #1 begins to decrease dramatically and by iteration 800 it is virtually gone. Figure A-15 further demonstrates this point by showing the remainder signal #1, and its spectrum beginning at iteration 3000. From the log spectrum plot it is clear that the most powerful sinusoid #1 has been highly attenuated. In fact it has been decreased by approximately 60 dB. (Additional experiments have shown this to be a typical value).

Figures A-16 through -18 show the behavior of the second separation stage. Its input is simply remainder output #1 (Fig. A-15). Adaptation was begun at iteration 1000 (after the convergence of NC #1). Figure A-16 shows the output of ALEWAIN #2 and its spectrum after convergence (iteration 3000). Sinusoid #2 appears almost exclusively at the filter output. Figure A-18 demonstrates remainder signal #2 after convergence. Sinusoid #2 (and #1, as well) has been almost completely extracted. The spectral plot shows that only sinusoid #3 and the input noise remain in the remainder output #2.

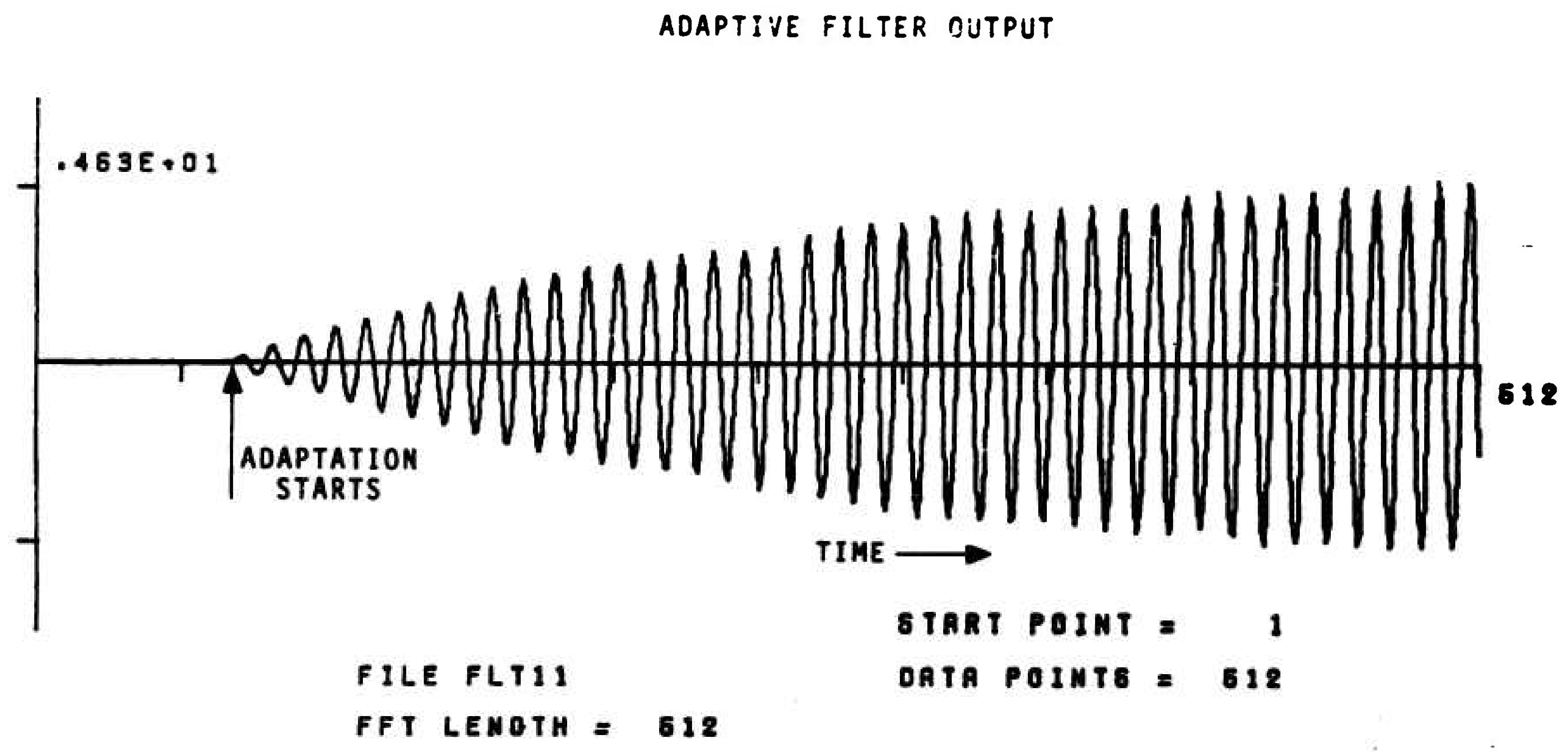


FIGURE A-11. FILTER OUTPUT SIGNAL FROM ALEWAIN #1 AT START OF ADAPTATION

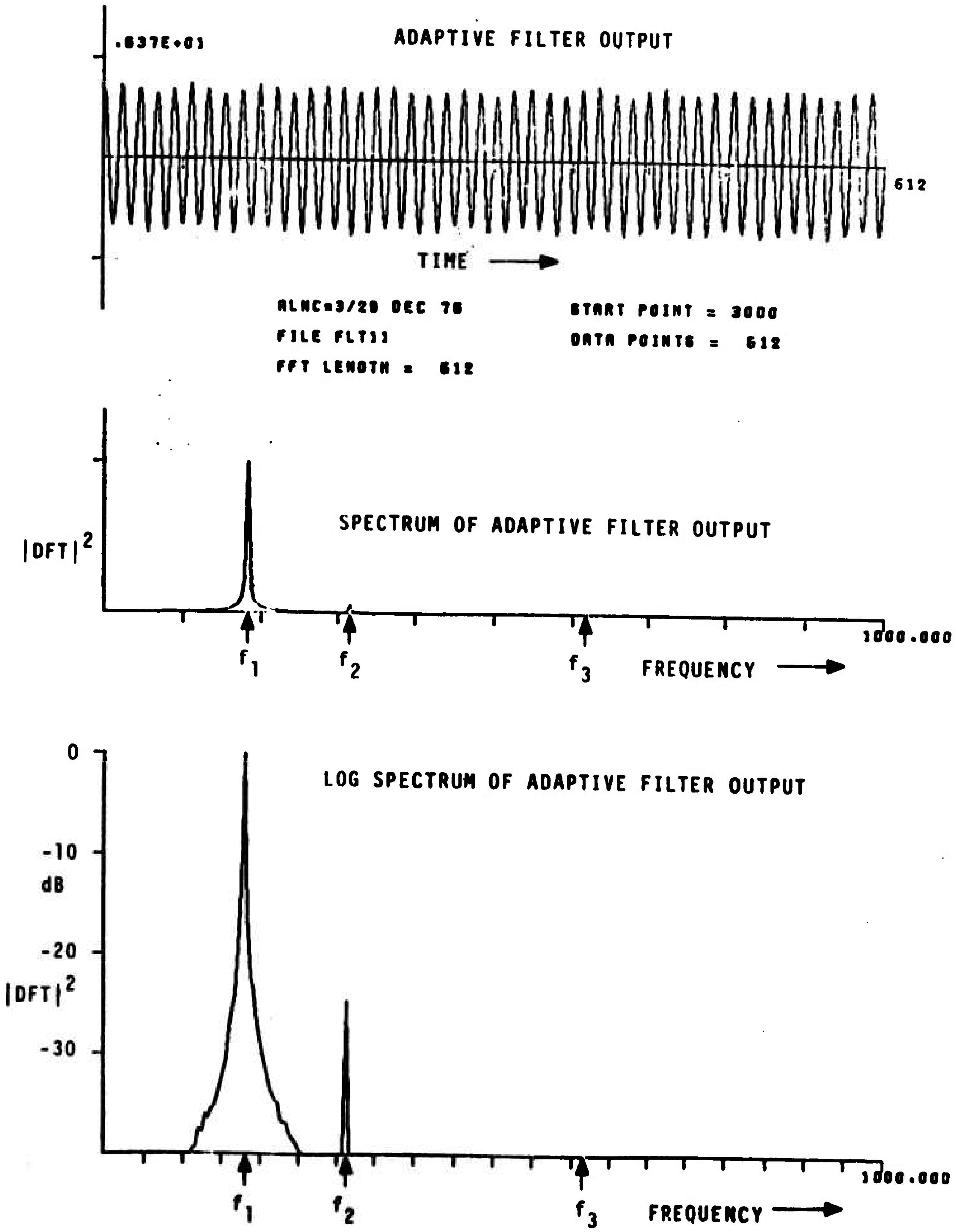


Figure A-12. FILTER OUTPUT SIGNAL FROM ALEHAIN #1 AT CONVERGENCE

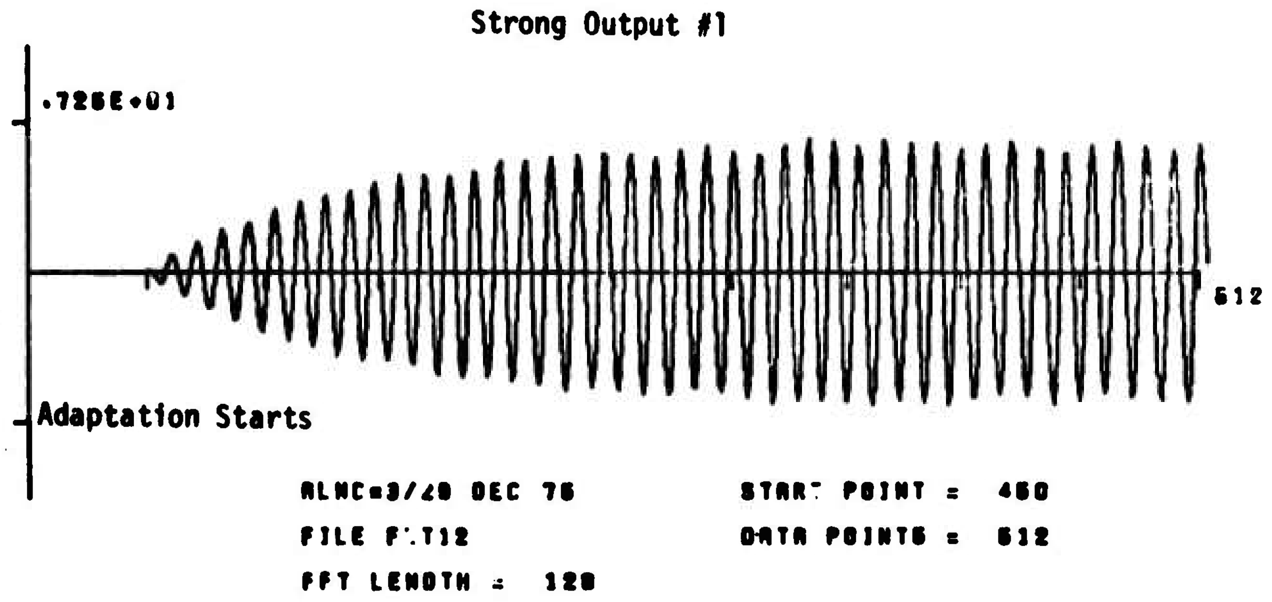


Fig. 13 Strong Output of NC #1 at Start of Adaptation

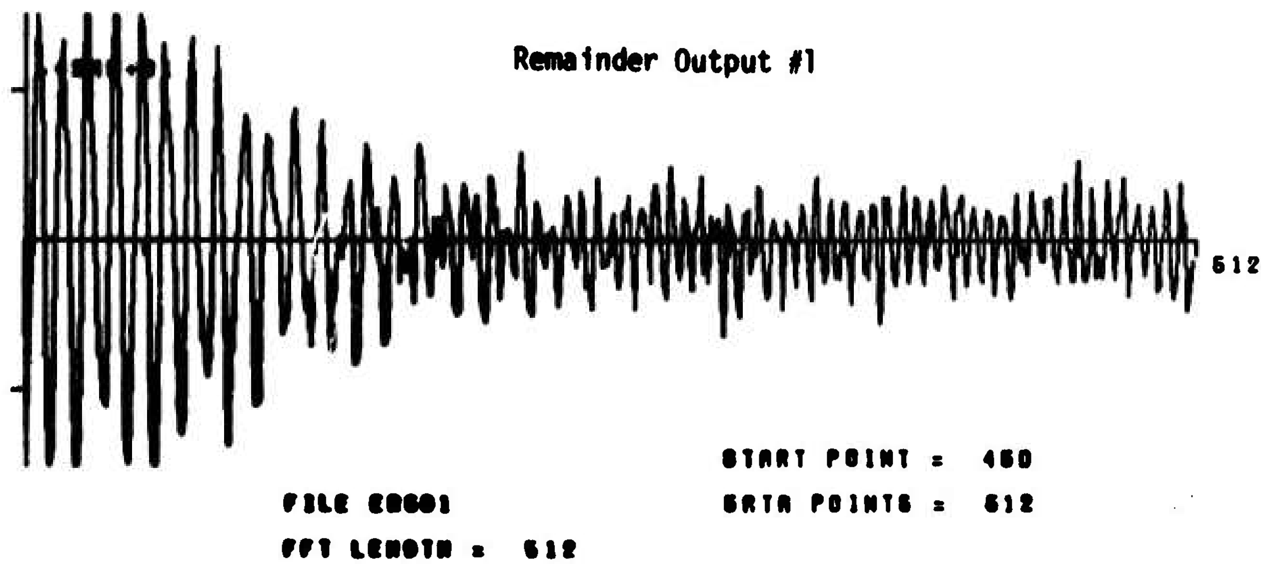


Fig. 14 Remainder Output of NC #1 at Start of Adaptation

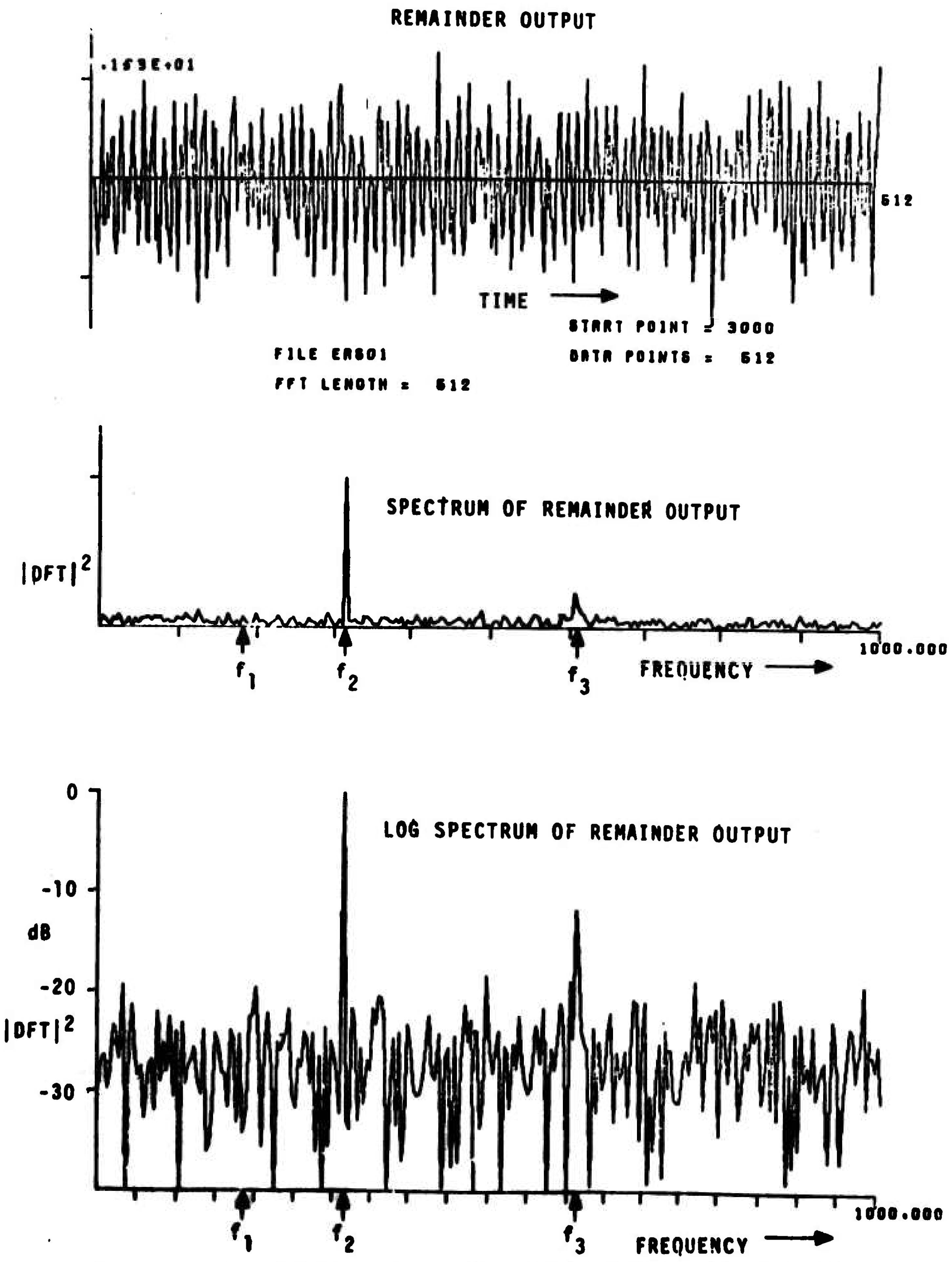


Figure A-15. REMAINDER OUTPUT AND SPECTRUM AT CONVERGENCE

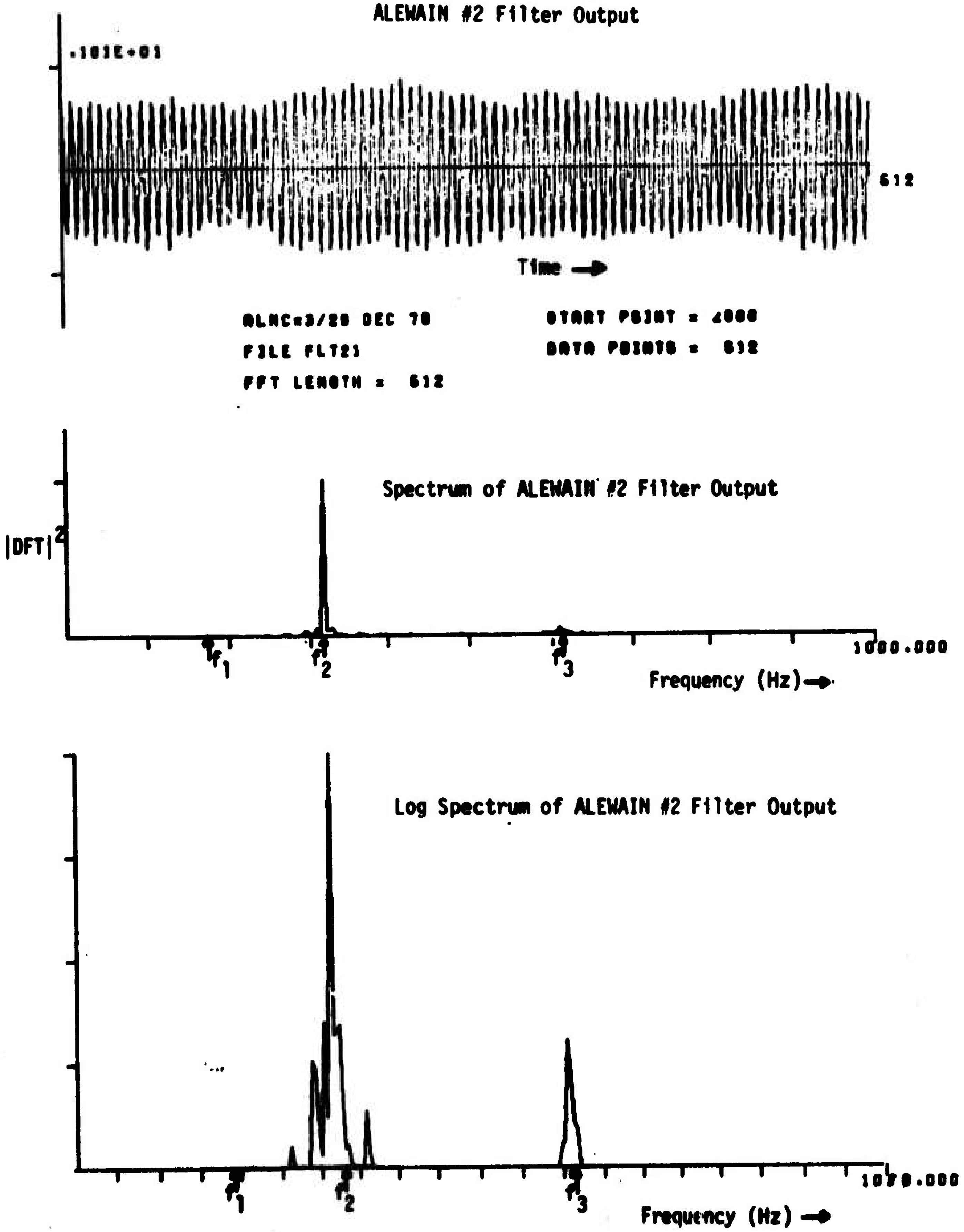
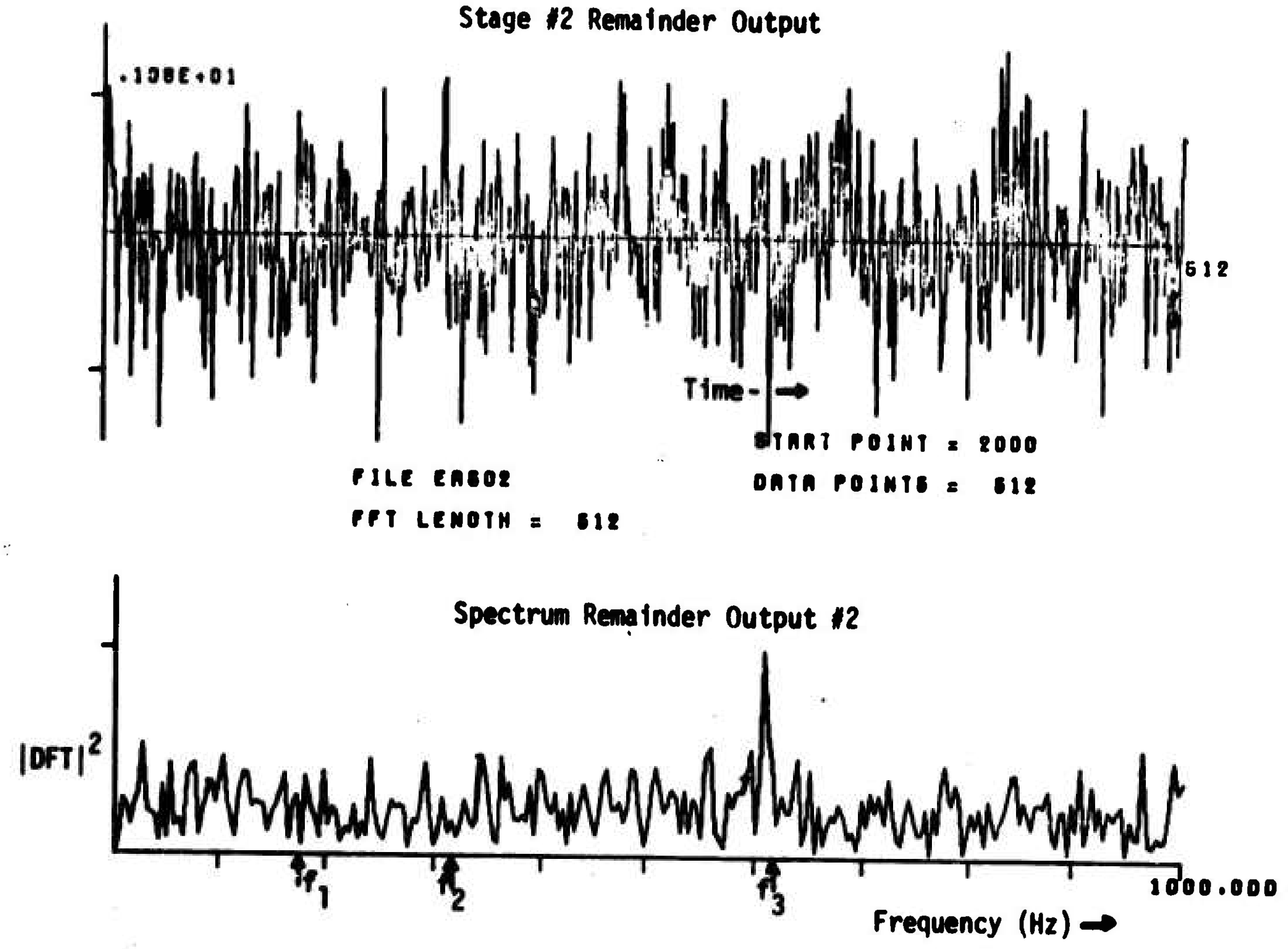


Fig. 16. ALEWAIN #2 Filter Output and Spectrum at Convergence



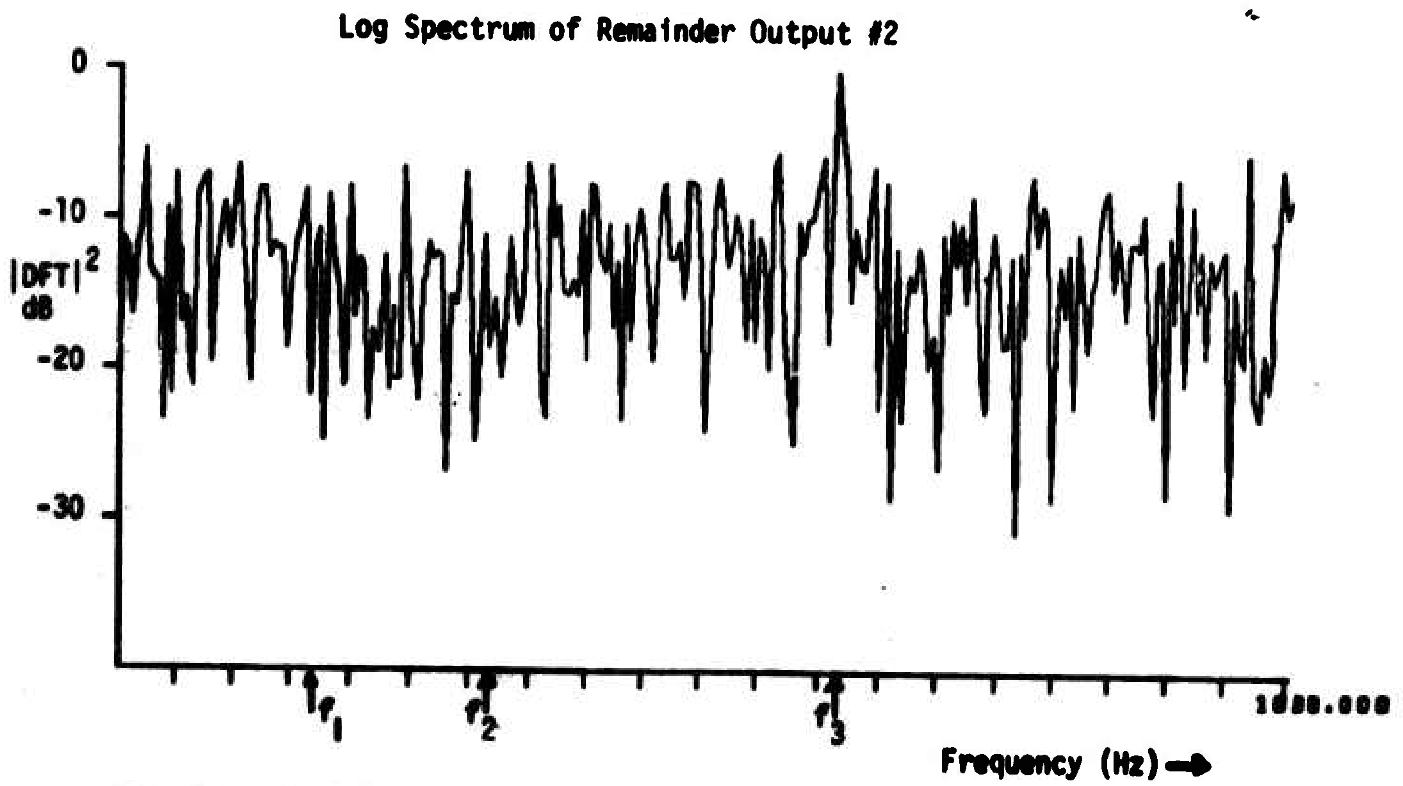


Fig. 17. MC #2 Remainder Output and Spectrum at Convergence

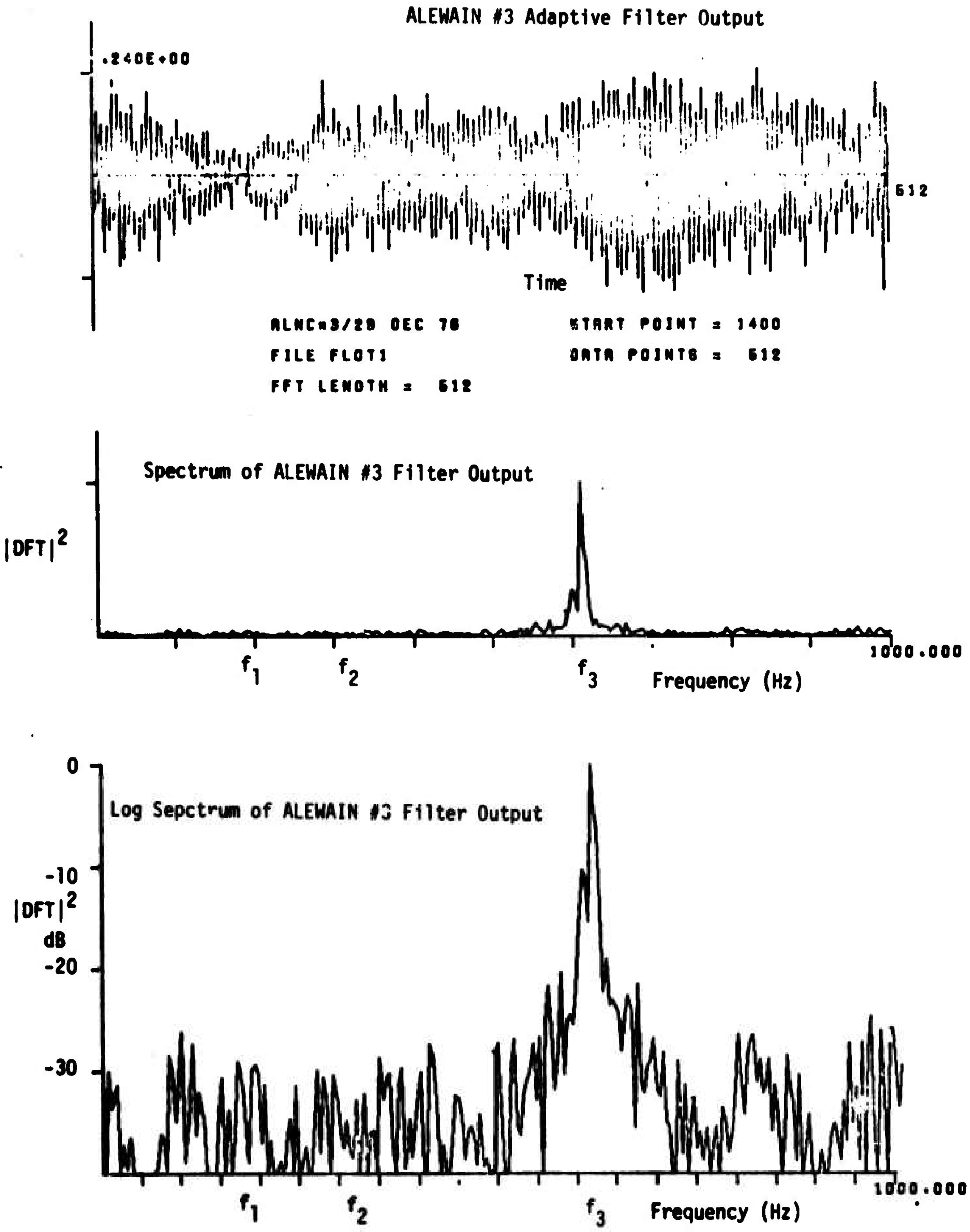


Fig. 18. ALEWAIN #3 Filter Output and Spectrum near Convergence

If this difference output is applied to the third separation stage the remaining sinusoid (#3) can be separated from the input noise. This signal and its spectrum are shown in Figure A-19.

This experiment demonstrates that the ALEWAIN + NC configuration can successfully be used to effect separation of narrowband signals on the basis of power when they do not overlap in frequency. The theory developed so far (and the experiment presented) are based on the use of sinusoids as the coherent inputs. With finite bandwidth narrowband signals, so long as the bandwidth of the relatively coherent components is considerably less than $1/\Delta$ (Δ=the decorrelation delay), each signal separation stage will slice off a separate signal. However it may be necessary to use more than one weight in the NC stage to effect complete separation. This will be discussed further in the next section.

VI. Choice of Operating Parameters

The previous theory and simulations have assumed that sufficient a priori knowledge about the input signal is available to the user so that the values of the adaptation and leakage constants can be appropriately adjusted. This section will explore some considerations in their choice.

A. Choice of the Leakage Coefficient γ

The slicing level of each separation stage is controlled by the leakage coefficient γ used for the associated ALEWAIN. This choice is determined by the fact that the most powerful component must be passed while less powerful components must be suppressed as much as possible. The nature of this problem can be understood by reexamining Figure A-6. This is the operating curve of the ALEWIN/ALEWAIN and shows the convergent gain of the <u>ith</u> input component as a function of the SNR_i' (the input SNR as modified by the algorithmically injected noise). Suppose that the input contains two coherent components whose power ratio is $P_1/P_2 = P > 1$. The goal of making a_2^* as small as possible while holding a_1^* fixed is clearly achieved by operating on the left side of the operating curve (where the curve is increasing approximately linearly). If the coherent components are assumed to be sinusoids, the maximum possible ratio of gains can be found analytically.

$$\frac{a_1^*}{a_2^*} = \frac{\dfrac{\frac{n}{2} \cdot \mathrm{SNR}_1'}{1 + \frac{n}{2} \cdot \mathrm{SNR}_1'}}{\dfrac{\frac{n}{2} \cdot \mathrm{SNR}_2'}{1 + \frac{n}{2} \cdot \mathrm{SNR}_2}} = \frac{\dfrac{2\sigma'^2}{nP_2} + 1}{\dfrac{2\sigma'^2}{nP_1} + 1} ,$$

where σ'^2 is the total equivalent noise power (i.e. $\sigma^2 + \sigma_A^2$, for the ALEWIN, or $\sigma^2 + \frac{1-\gamma}{\mu}$, for the ALEWAIN). If SNR_1' and SNR_2' are large compared to $\frac{2}{n}$ then $a_1^*/a_2^* \approx 1$ and there is no gain difference. This corresponds to locating both components on the right hand side of Figure A-6. Since there is no gain difference, separation cannot be attained. If, however, SNR_1' and SNR_2' are both considerably less than $\frac{2}{n}$, then:

$$\lim_{\sigma'^2 \to \infty} \frac{a_1^*}{a_2^*} = \frac{P_1}{P_2} = P .$$

This indicates that the best separation is attained by operating as far left as possible in Figure A-6. Unfortunately this desirable behavior is offset by the fact that operation in this region (where the injected noise completely controls the dynamics of adaptation) tends to disturb the gain relationships between the various parts of a non-sinusoidal input component. As a result a single weight noise canceller will not suffice to completely cancel this component in the NC remainder output. This problem can be circumvented to some degree by adding more weights to the NC stage. Experiments have shown however that generally good performance can be attained for both sinusoidal and narrowband inputs by setting γ (or σ_A^2) so that a_1^*, the gain for the most powerful component equals .5. This implies that $SNR_1' \cdot \frac{n}{2} = 1$ and corresponds to the "knee" or breakpoint in the a* vs. SNR' curve. The values of γ used in the various separation stages of the experiment shown in section V were chosen in this manner.

In actual practice the power of the most powerful component is usually unknown and this complicates the choice of γ. The value of γ could be swept to search for a solution. By simply examining the input waveform and attributing its maximum excursion to the most powerful sinusoid, the power of this component may be roughly estimated by the RMS power of the input itself. In the case where the coherent inputs have widely disparate powers and all have SNRs greater than one, this is a fairly accurate estimate and the value of γ found from this estimate will work well. If the conditions are not satisfied, then the performance will be poorer. However this adaptive structure is quite tolerant of small parameter mischoices and will usually provide very good performance even if the power is mis-estimated. If the SNR condition is not met, then the background noise will determine the RMS input power and hence the choice of γ.

B. Time Constant Determination

The convergence times of the ALEWAIN and the NC are determined by μ_1 and μ_2, respectively. The convergence behavior of these adaptive processes an be quantified by finding the time constants associated with the uncoupled modes in each of the processes' weight vectors. To determine this behavior for the ALEWAIN, consider again Equation III-7c,

$$E[W(k+1)] = \left\{ I-\mu_1 [\frac{1-\gamma}{\mu_1} I + R_x] \right\} E[W(k)] + \mu_1 P_x .$$

Since R_x is real and symmetric it is possible to find a coordinate transformation which uncouples the modes of the adaptive process [8]. If this is done the expected value of a typical uncoupled weight, $w_i(k)$ say, can be written [8] as:

$$E[w_i'(k+1)] = [\gamma+\mu_1\lambda_i] \; E[w_i'(k)] + \mu d_i,$$

where σ^2 is the input noise power and λ_i is the eignevlaue of R_x associated with the ith uncoupled input mode. The growth time constant of such a recursion expression can be shown to be:

$$\tau_i = \frac{1}{1-\gamma+\mu_1\lambda_i} \quad , \; 1 \leq i \leq n$$

Notice that if $\gamma = 1$ then the time constant degenerates to that for the ALE. If $\mu_1\lambda_i \gg 1-\gamma$ then the adaptive time constant for this mode is determined by μ_1 and the powers of the input noise and the uncoupled coherent component. If, however, μ_1 or the input powers are so small that $1-\gamma>\mu_1\lambda_i$ then the adaptive dvnamics are determined only by γ. If $1-\gamma>\mu_1\lambda_i$, for all i, then the ALEWAIN becomes a recursive correlator [9] with a time constant of $\frac{1}{1-\gamma}$ for all modes. In the case of sinusoidal inputs these solutions can be put in terms of the power of the most powerful sinusoidal input. It can be shown [8] that a sinusoidal input induces two eigenvalues of R_s which are approximately equal to $\frac{nP}{2}$, where P is the power of the sinusoid. In this case the adaptive time constant associated with the two modes of interest is given by:

$$\tau_s = \frac{1}{1-\gamma+\mu_1(\sigma^2+\frac{nP}{2})}$$

Convergence of the adaptive algorithm is a matter of definition but a practical value is twice the longest growth time constant of interest. Therefore, for a sinusoidal input, the estimated ALEWAIN convergence time is given by:

$$ETC_A = \frac{2}{1-\gamma+\mu_1(\sigma^2+\frac{nP}{2}}$$

If the sinusoid is very strong then ETC_A tends to $4/\mu_1 nP$. However, if μ_1 is very small then ETC_A tends to $2/(1-\gamma)$(i.e. the recursive correlator case).

Determination of the time constant for the one-weight noise canceller is quite simple. The expected behavior of the weight is described by:

$$E[w(k+1)] = (1-\mu_2 P_y)\; E[w(k)] + \mu_2 C,$$

where P_y is the expected power in $y(k)$, the ALEWAIN filter output, and C is the crosscorrelation of the primary input with the ALEWAIN filter output. In a fashion similar to that above the estimated convergence time can be shown to be:

$$ETC_{NC} = \frac{2}{\mu_2 P_y}$$

The power in $y(k)$ is determined by the value of a* chosen for the ALEWAIN via γ. If the recommended choice of a* = .5 is made and if the component of interest is sinusoidal with input power P then ETC_{NC} reduces to:

$$ETC_{NC} = \frac{16}{\mu_2 P}$$

As the assumption regarding infinite coherence (sinusoidal inputs) is violated the time constant estimates given here become poorer. However these estimates work well in practice and should give good results. Further insight into the convergence times of adaptive processor such as the ALE and ALEWAIN can be found in reference 8.

VII. Conclusions

This work has demonstrated in a preliminary way the practicalitv of the ALEWAIN + NC adaptive processor as a signal sorter or separator, particularly in the case where the desired singals are narrowband and their signal-to-noise ratios are high (>1). Jamming signals would have such high SNR's. Results presented here have demonstrated how to design a signal separator, how to choose (either manually or automatically) the operating parameters, and how to estimate the convergence time of such a processor. These results should be very useful for a variety of communications and signal processing applications (i.e., anti-jamming). However an equally important part of this research was the discovery and preliminary examination of the leaky LMS (LLMS) adaptive algorithm as applied to adaptive line enhancing. Even though much remains to be explored about the behavior of this algorithm it seems clear that it will have wide applicability to sonar and radar signal processing. In spite of the fact that it was conceived for the ALEWIN configuration through evolutionary development, it can be generalized to more complex temporal and spatial filtering applications.

BIBLIOGRAPHY

1. B. Widrow, J. R. Glover, Jr, J. M. McCool, J. Kaunitz, C. S. Williams R. H. Hearn, J. R. Zeidler, E. Dong, Jr. and R. C. Goodlin, "Adaptive Noise Cancelling: Principles and Applications," Proc. IEEE, Vol. 63, No. 12, pp. 1692-1716, Dec. 1975.

2. J. R. Treichler, B. Widrow, NAVAIR Quarterly report 1 and 2, "Adaptive Separation of Signals in Noise in Terms of Their Relative Powers," for interval 1 September 1975 to 29 February 1976.

3. J. R. Treichler, B. Widrow, NAVAIR Quarterly report 3 and 4, "Improved Methods of Adaptive Separation of Signals in Noise in Terms of Their Relative Powers," for interval 1 March 1976 to 31 August 1976.

4. J. R. Zeidler and D. M. Chabries, "An Analysis of the LMS Adaptive Filter used as a Spectral Line Enhancer," Naval Undersea Center Report, TN 1476, February, 1975.

5. B. Widrow, P. Mantey, L. Griffiths, and B. Goode, "Adaptive Antenna Systems," Proc. IEEE, Vol. 55, pp. 2143-2159, Dec. 1967.

6. G. Zahm, "Application of Adaptive Arrays to Suppress Strong Jammers in the Presence of Weak Signals," Trans of Aerospace and Electronic Systems, Vol. AES-9, No. 2, March, 1973, pp. 260-271.

7. T. Daniell, "Adaptive Estimation with Mutually Correlated Training Samples," Stanford Electronics Lab., Stanford Univ,. Rep. SEL-68-083, Aug. 1968, (Ph. D. dissertation).

8. J. Treichler, Ph.D. dissertation, "The Spectral Line Enhancer, " (in preparation).

9. J.R. Glover, letter (private communication).

Part B

ADAPTIVE BEAMFORMING WITH INJECTED NOISE

I. Introduction

In this section an adaptive antenna array system with a special pilot signal is proposed and studied. The goal is to produce an antenna system which responds to signals as a function of their power levels — the stronger the signal, the more attenuation desired. A system of this type would allow reception of weak signals in an environment containing stronger, undesired signals (i.e., jammers).

An adaptive system is desired to cope with the non-stationary character of most environments. The non-stationarity arises from signals turning on and off, signals fading, signal sources moving in space, and possibly the receiving array changing in physical orientation (due either to being mounted on a moving vehicle, or on a base subject to stretching and malformation).

The use of an antenna array as opposed to a single antenna is desired, since this allows spatial filtering in addition to frequency filtering. Therefore, the system can form nulls in its reception pattern in the direction of undesired signals.

A final objective of the system is to maintain reception sensitivity in directions where no signals are currently to be found. This allows for immediate acquisition of desired signals when they start up and for essentially unattenuated reception of low power signals arriving at unknown directions of incidence.

In conclusion, we wish the adaptive antenna system to have the following properties:

1) attenuation based on signal power strength
2) fast response to changes in the signals

3) response to changes in the array itself

4) receptivity in directions where no signals currently exist.

2. Presentation of the Algorithm

In this section we propose an adaptive algorithm for use in an array antenna system, designed to fulfill the functions outlined in the previous section. Due to the characteristics of the algorithm and its relationship to conventional adaptive beamforming antenna systems, this adaptive array antenna system will be called the Adaptive Beamformer With Injected Noise (ABWIN).

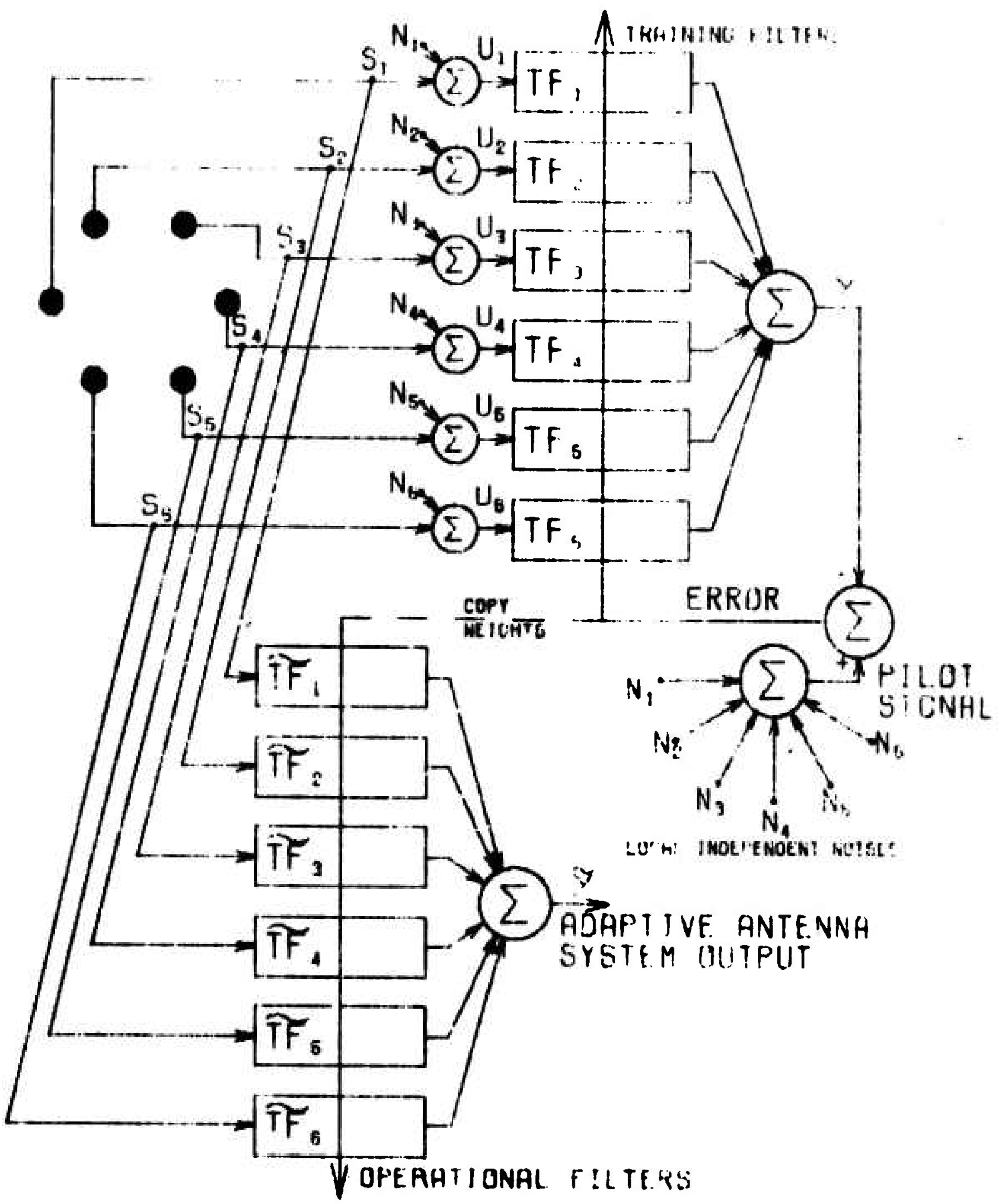
2.1 Introduction to the Algorithm

The basic idea behind the algorithm is to feed into an adaptive antenna array processor the signals received from the environment, augmented by a specially chosen pilot signal. The pilot signal of the ABWIN is designed to place "soft constraints" on the array's response to signals; the intent is to have an omnidirectional reception capability in the absence of strong (possibly jamming) signals, but attenuating strong signals when they do occur, the degree of attenuation being a function of the signal power and the pilot signal power.

2.2 The ABWIN

The structure of the adaptive array system will now be described; a discussion on the pilot signal will then be presented.

Figure B.1 illustrates the structure of the adaptive array system. Signals are received from the environment by an array of antenna elements (the array geometry is shown in the figure for illustrative purposes as six elements in a circular pattern — the geometry of an actual antenna array may be any configuration). Added to the outputs of the antenna elements are the individual components of the pilot signal (labelled n_1



TRAINING FILTERS
S_1 S_2 S_3 S_4 S_5 S_6
N_1 N_2 N_3 N_4 N_5 N_6
U_1 U_2 U_3 U_4 U_5 U_6
TF_1 TF_2 TF_3 TF_4 TF_5 TF_6
ERROR
COPY WEIGHTS
PILOT SIGNAL
LOCAL INDEPENDENT NOISES
$\hat{TF}_1$ $\hat{TF}_2$ $\hat{TF}_3$ $\hat{TF}_4$ $\hat{TF}_5$ $\hat{TF}_6$
ADAPTIVE ANTENNA SYSTEM OUTPUT
OPERATIONAL FILTERS

Figure B.1 Structure of the ABWIN for a six element antenna array.

through n_6 in the figure). The resulting signals are inputs to a set of transversal filters, whose outputs are summed to produce the array's output. This output is subtracted from the pilot signal producing an error signal which is used by the ABWIN for updating the impulse response of the transversal filters.

We see that the array's output contains the pilot as well as the received signals, which is clearly undesirable. To overcome this problem, a second set of transversal filters is established so that the received signal can be passed through this set without the addition of the pilot signal. Therefore, one set is used for the adaptation or training of the system; (the training filters) the second may be regarded as an operational set of filters whose output is the useful system output. We will assume that the reference signal is formed as illustrated in the figure: simply add the pilot signal components (n_1 through n_6), and use the result as the reference signal in the adaptation algorithm. Other methods of forming the pilot signal are possible and will be discussed later. The adaptation algorithm used for adjustment of the transversal filter weights is Widrow-Hoff Least Mean Square (LMS) algorithm [ ], which will be discussed in more detail in the next section.

The pilot signal of the ABWIN is constructed in a special manner. Each of the pilot signal components (n_1 through n_6 on the figure) is a noise signal, generated independently of the other pilot signal components, and the external signals. The pilot signal is then the sum of the pilot signal components. This method of construction gives the pilot signal the property that it does not appear to be arriving from any specific direction, unlike the pilot signal of conventional adaptive beamformers [ ]. The effect of a pilot signal constructed in this manner is described in a later section.

2.3 Mathematical definition of the ABWIN

We will now describe the ABWIN mathematically. Let there be M antenna elements. Denote the output of sensor i at time k by $s_i(k)$ $(i=1, \ldots, M)$. Denote the component of the pilot signal added to the ith sensor signal by $n_i(k)$. Denote their sum by $u_i(k)$.

$$u_i(k) = s_i(k) + n_i(k)$$

Associated with each element is a transversal filter (TF) where TF_i is associated with ith element. Each TF can be described by two M dimensional vectors:

a) the contents of the tapped delay line. For the training filters we will denote the contents of the delay elements at time k of TF_i by

$$U_i(k) = [u_i(k)\ u_i(k-1) \ldots u_i(k-n+1)]^T .$$

where n is the number of elements in the tapped delay line. For the operational filters we will denote the contents of the delay line of TF_i at time k by

$$S_i(k) = [s_i(k)\ s_i(k-1) \ldots s_i(k-n+1)]^T .$$

b) the weights of the transversal filter, which are the same for the training filters TF_i and the operational filters TF_i. We will denote the weight vector at time k of filters TF_i and TF_i by:

$$W_i(k) = [w_{i1}(k)\ w_{i2}(k) \ldots w_{i\,n}(k)]^T .$$

Using this notation, the output of TF_i at time k is: $Y_i(k) = W_i^T(k) U_i(k)$; and the output of TF_i at time k is: $Y_i(k) = W_i^T(k)\ S_i(k)$.

For the purpose of writing the adaptation algorithm in vector notation, we need the following vectors:

$U(k)$ = the augmented tapped delay line contents vector of the training set of filters

$$U(k) = \begin{bmatrix} U_1(k) \\ \hdashline U_2(k) \\ \hdashline \vdots \\ \hdashline U_M(k) \end{bmatrix}$$

$S(k)$ = the augmented tapped delay line contents vector of the operational set of filters (dimension of $M_n x l$)

$$S(k) = \begin{bmatrix} S_1(k) \\ \hdashline S_2(k) \\ \hdashline \vdots \\ \hdashline S_M(k) \end{bmatrix}$$

$N(k)$ = the augmented pilot signal vector (dimensioned $M_n x l$)

$= X(k) - S(k)$

$W(k)$ = the augmented TDL weight vector (dimensioned $M_n x l$)

$$W(k) = \begin{bmatrix} W_1(k) \\ \hdashline W_2(k) \\ \hdashline \vdots \\ \hdashline W_M(k) \end{bmatrix}$$

Therefore, the output of the training filters (which have the pilot signal) is $y(k) = W^T(k)U(k)$. The output of the operational filters (which contain only the sensor signals) is: $\tilde{y}(k) = W^T(k)S(k)$.

The operation of the ABWIN can now be described mathematically. At time k

1) Input the new data, shifting the old data down the tapped delay lines:

$$U(k) = \mathbb{A}U(k-1) + \mathbb{B}u(k)$$

$$S(k) = \mathbb{A}S(k-1) + \mathbb{B}s(k)$$

where $u(k) = [u_1(k)\dots u_M(k)]^T = s(k)+n(k)$

$s(k) = [s_1(k)\dots s_M(k)]^T$

A = the tapped delay line shift matrix

B = the tapped delay line input matrix

These equations can be written in a component form, as illustrated below for the U(k) vector:

$$U(k) = \begin{bmatrix} u_1(k) \\ \hline u_2(k) \\ \hline \vdots \\ \hline u_M(k) \end{bmatrix} = \left[\begin{array}{c|c|c|c} A & 0 & \dots & 0 \\ \hline 0 & A & \dots & 0 \\ \hline \vdots & \vdots & \ddots & \vdots \\ \hline 0 & 0 & \dots & A \end{array}\right] \begin{bmatrix} U_1(k) \\ \hline U_2(k)-1 \\ \hline \vdots \\ \hline U_M(k) \end{bmatrix} + \left[\begin{array}{c|c|c|c} B & 0 & \dots & 0 \\ \hline 0 & B & \dots & 0 \\ \hline \vdots & \vdots & \ddots & \vdots \\ \hline 0 & 0 & \dots & B \end{array}\right] \begin{bmatrix} u_1(k) \\ \hline u_2(k) \\ \hline \vdots \\ \hline u_M(k) \end{bmatrix}$$

where

$$A = \begin{bmatrix} 0 & 0 & 0 & \dots & 0 & 0 \\ 1 & 0 & 0 & \dots & 0 & 0 \\ 0 & 1 & 0 & \dots & 0 & 0 \\ 0 & 0 & 1 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & 1 & 0 \end{bmatrix} \qquad B = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \text{ A is nxn, B is nx1}$$

Thus

$$\mathbf{A} = \begin{bmatrix} A|0|\ldots|0 \\ 0|A|\ldots|0 \\ \\ 0|0|\ldots|A \end{bmatrix}$$

$$\mathbf{B} = \begin{bmatrix} B|0|\ldots|0 \\ 0|B|\ldots|0 \\ \\ 0|0|\ldots|B \end{bmatrix}$$

2) Calculate the system output:

$$y(k) = W^T(k)U(k)$$

$$\tilde{y}(k) = W^T(k)S(k)$$

3) Calculate the system error for use in the adaptation equation:

$$e(k) = d(k)-y(k)$$

For most of the remainder of this report we will assume the pilot (or "desired") signal is formed as follows:

$$d(k) = \sum_{i=1}^{M} n_i(k)$$

Other possibilities will be discussed in a later section.

4) Perform the LMS adaptation:

$$W(k+1) = W(k) + 2\mu e(k)U(k)$$

where μ is the adaptation constant.

2.4 The Pilot Signal

To this point very little has been said about the pilot signal and its components, but the functioning of the ABWIN depends heavily on the pilot signal.

It was stated previously that the purpose of the pilot signal in the ABWIN is to place "soft" constants in the array's response to signals; the intent being to maintain an omnidirectional reception capability in the absence of signals. To attain the capability of omnidirectionality, the reference signal components ($n_i(k)$, $i=1$, M) are independent white noises.

Consider a signal received by the array from an external source. Such a signal appears identical to each sensor, with the exception that the signal may arrive at a different time. Thus there is a correlation between sensors for a signal with any spatial orientation.

The ABWIN pilot signal is different, however. Since the pilot signal components are independent noises, there is no correlation between the pilot signal components on different sensors. Thus, the pilot signal component from one sensor cannot be used to cancel any portion of the pilot signal component of another sensor. Because of this, the pilot signal does not appear to arrive from a specific direction. In other words, it does not exhibit any spatial orientation.

With a pilot signal created as described above, the ABWIN cannot place a lobe in a given direction to enhance reception of the pilot signal as in conventional adaptive arrays. At the same time, any change to the TF weights does affect the system's response to the pilot signal. Thus, in the absence of external signals, the ABWIN attains a reception pattern which will be referred to as the "quiescent" pattern. When an external signal is received, the ABWIN reacts to place a null in the reception pattern in the direction and frequency of the received signal. This null also decreases the gain of the pilot signal through the system; as a result, the ABWIN will adjust to a reception pattern that "balances" the amount of the pilot signal lost against the amount of the external signal that is allowed to pass. This, then, is the "soft constraint" capability of the ABWIN.

3. The Quiescent Pattern of the ABWIN

Let us now determine the quiescent pattern of the ABWIN, by examining the impulse response of the various transversal filters in the absence of signals received by the elements.

Consider the way in which the pilot signal is generated. The signal used for reference purposes is the sum of the individual components. Therefore, we see from Figure 1 that in the absence of any external signals, the adaptive array can obtain zero error if its output is simply the sum of the current inputs to the TF's. This is achieved when each TF has a weight vector which is zero except for the weight corresponding to the most recent input. This most recent input has a weight of 1 associated with it. Thus, the quiescent system weight vector is:

$$W(k)=\left[\begin{array}{c} 1 \\ 0 \\ \vdots \\ 0 \\ \hdashline 1 \\ 0 \\ \vdots \\ 0 \\ \hdashline \vdots \\ 1 \\ 0 \\ \vdots \\ 0 \end{array}\right]\begin{array}{l} \\ \\ n \text{ elements} \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \end{array}$$

In other words, the response of each TF to a unit impulse is a unit impulse.

Note that the zero weights arise from the fact that white noise sources are used for generating the pilot signal components. Since white noise has no time correlation, the pilot signal components from previous time samples are of no aid in "predicting" the reference signal at this time instant. Thus a zero weight is associated with all delayed samples.

The weights associated with the current inputs must be unity. This is a result of the statistical independence of the individual pilot signal components. Since the components are statistically independent, none of the components is any aid in "predicting" the value of another component. Thus to change a weight from unity would only add to the system output a quantity which could not be cancelled by the pilot signal, resulting in a non-zero error.

Thus we see that in the absence of external signals, the ABWIN can produce a zero error by selection of a unique weight vector, which has the effect of just summing the current inputs to produce the output, with no dependence on past inputs.

This "quiescent" weight vector determines the "quiescent" array reception pattern, in conjunction with the antenna array geometry. This quiescent pattern is simply the pattern obtained when the antenna element outputs are directly summed. Thus the sensor geometry has a direct effect on the quiescent pattern of the ABWIN, but does not effect the quiescent weight vector. A method of modifying the pilot signal to modify the quiescent weight vector, allowing a broader choice of quiescent reception pattern, is proposed in a later section.

4. Analysis of the Convergence Point of the ABWIN

In this section we will analytically determine the mean (expected value) of the weight vector, at congergence. Begin with the adaptation equation:

$$W(k+1) = W(k) + 2\mu e(k)U(k) = W(k) + 2\mu U(k)e(k)$$

Substituting for the error:

$$\begin{aligned} W(k+1) &= W(k) + 2\mu U(k)[d(k)-y(k)] \\ &= W(k) + 2\mu U(k)[d(k)-U^T(k)W(k)] \\ &= W(k) + 2\mu d(k)U(k) - 2\mu U(k)U^T(k)W(k) \end{aligned}$$

Now take the expectation:

$$E\{W(k+1)\} = E\{W(k)\} + 2\mu E\{d(k)U(k)\} - 2\mu E\{U(k)U^T(k)W(k)\}$$

Now we make the approximation that

$$E\{U(k)U^T(k)W(k)\} = E\{U(k)U^T(k)\}\, E\{W(k)\}$$

which is good for small μ. Under this approximation:

$$E\{W(k+1)\} = E\{W(k)\} + 2\mu E\{d(k)U(k)\} - 2\mu E\{U(k)U^T(k)\}\, E\{W(k)\}$$

At convergence, we have $E\{W(k+1)\} = E\{W(k)\}$, which is achieved when

$$E\{U(k)U^T(k)\}\, E\{W(k)\} = E\{d(k)U(k)\}$$

Therefore at convergence,

$$E\{W(k)\} = E\{U(k)U^T(k)\}^{-1}\, E\{d(k)U(k)\}$$

For ease of notation, let

$$R_{uu} \triangleq E\{U(k)U^T(k)\}$$

$$P = E\{d(k)U(k)\}$$

$$\overline{W} = E\{W(k)\}$$

Thus the converged weight vector satisfies

$$\bar{W} = R_{uu}^{-1} P.$$

Now, since

$$d(k) = \sum_{i=1}^{M} n_i(k)$$

we have

$$E\{d(k)U(k)\} = \sum_{i=1}^{M} E\{n_i(k)U(k)\}$$

However, the pilot signals are white and independent of one another and any external signals. Therefore,

$$P = \begin{bmatrix} \sigma^2 \\ 0 \\ \vdots \\ 0 \\ \hline \vdots \\ \hline \sigma^2 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

where σ^2 is the power of a pilot signal component: $\sigma^2 \triangleq E\{n_i^2\}$.

However, R_{uu} can be decomposed into a contribution from the external signals and one from the pilot signals.

$$R_{uu} = E\{[S(k)+N(k)]\ [S^T(k)+N^T(k)]\} .$$

But the $n_i(k)$'s were constructed to be independent of the $s_i(k)$'s. So R_{uu} can be broken into the sum of external signal and pilot signal covariance matrices.

$$\begin{aligned} R_{uu} &= E\{S(k)S^T(k)\} + E\{N(k)N^T(k)\} \\ &= R_{ss} + R_{nn} \end{aligned}$$

Thus, for the mean weight vector at convergence, we have

$$\bar{W} = (R_{ss} + R_{nn})^{-1} P$$

Now, since the n_i are independent white noises of variance σ^2, $R_{nn} = \sigma^2 I$ where I is an identity matrix. Thus

$$\bar{W} = (R_{ss} + \sigma^2 I)^{-1} P$$

The term R_{ss} itself may be the sum of a set of matrices, each dependent on signals received by the array.

We see that the converged weight vector is a function of the reference signal power σ^2, as well as a function of the signals in the tapped delay line (R_{ss}). R_{ss} is dependent upon both the time correlation characteristics of the external signals as well as the sensor geometry. A signal with non-zero correlation across a delay of one or more time samples will introduce non-zero off-diagonal terms in R_{ss}, due to the tapped delay lines. In addition, an external wavefront received by the sensor array arrives at each sensor delayed in time by an amount dependent upon the sensor geometry and the direction of arrival of the wave. This geometry dependent time delay affects correaltion between the contents of the tapped delay lines of different sensors. Thus we see that the cross-correlation matrix R_{ss} is a complicated function of the statistical characteristics of the received signals and the sensor geometry.

We see that in the absence of external signals, $R_{ss} = 0$, and the quiescent weight vector is:

$$\bar{W} = (\sigma^2 I)^{-1} P = \frac{1}{\sigma^2} P = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \\ \hdashline \vdots \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

as predicted in an earlier discussion.

5. Application of the ABWIN

5.1 Introduction

In this section the ABWIN is applied to a particular array configuration under several different signal environments. The simulation of the ABWIN is compared with results calculated from the theory presented earlier to demonstrate the validity of the theory. Then the theory is used to calculate the ABWIN behavior under different signal conditions to demonstrate the ABWIN response to a signal on the basis of its power.

5.2 The Array Configuration

Figure B.2 shows the sensor geometry used in this section. The speed of signal propagation is 1, the sampling rate is .125, and each tapped delay line contains 8 taps. Thus each tapped delay line will hold 1 cycle of a sine wave of frequency 1, and if the signal sine wave (with frequency 1) is arriving from the 0° direction, the signal at sensor 4 is shifted 180° in phase from the signal at sensor 1.

5.3 Simulation Results

This section presents an example where the converged value of the mean weight vector is computed from the theoretical results presented earlier, and compared with an actual weight vector obtained from a computer simulation of the ABWIN.

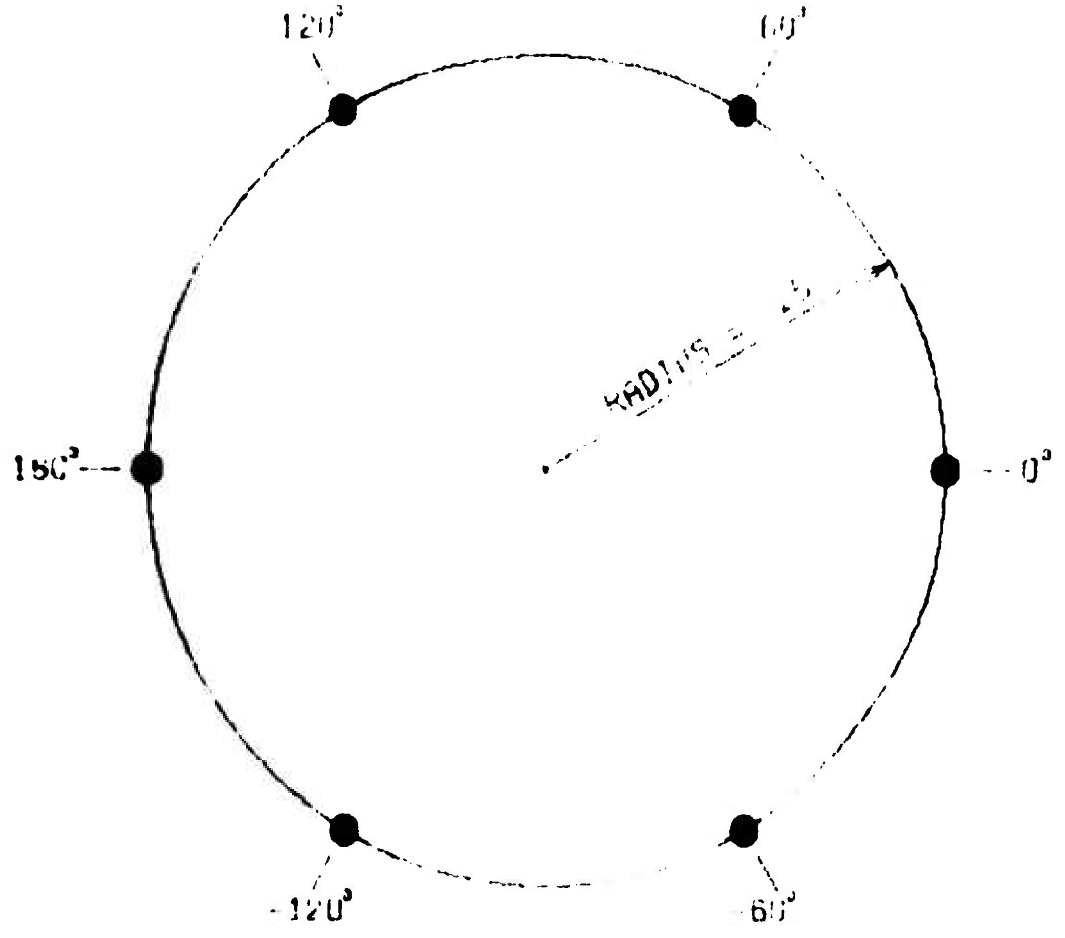


Figure B.2 Geometry of the antenna array used in the simulations.

In this example two signals are present. The first signal has a frequency of 1., a power of 1., and is arriving from a direction of 18°. The second signal has a frequency of 2., a power of 10., and is arriving from a direction of 108°. The pilot signal component power (σ^2) is 5. The calculated value of $\overline{W}$ is compared below with a weight vector obtained from a simulation of the ABWIN. Note that the simulation weight vector is an instantaneous weight vector; no averaging was used to obtain this vector.

The weight vector consists of 48 (6x8) elements. The first eight elements correspond to the TF associated with sensor 1 (TF_1). The next eight elements correspond to TF_2 and so on. The first weight of each set of eight corresponds to the weight associated with the most recent sample, the second to the next oldest sample, and so on.

The theoretical and measured weight vectors are presented in Table B.1.

TABLE B.1 THEORETICAL AND MEASURED WEIGHT VECTORS

$\bar{W}$ (theoretical)	ABWIN simulation
1.0358	1.0344
- .1375	- .1224
- .1407	- .1326
- .0003	.0020
.0508	.0559
.0109	.0015
.0541	.0332
.1269	.1244
.9228	.9224
- .0245	- .0231
- .0510	- .0582
- .1025	- .1103
- .0005	.0031
.1568	.1600
.1287	.1174
- .0297	- .0451
.8309	.8232
- .0380	- .0515
.1079	.0883
.0823	.0735
.0160	.0153
.0485	.0572
.0452	.0579
- .0929	- .0781
1.0358	1.0269
.1269	.1119
.0541	.0490
.0109	.0171
.0508	.0494
- .0003	- .0040
- .1407	- .1349
- .1375	- .1472
.9228	.9055
- .0297	- .0243
.1287	.1391
.1568	.1648
- .0005	.0048
- .1026	- .1075
- .0510	- .0565
- .0245	- .0365
.8309	.8271
- .0929	- .0978
.0452	.0443
.0485	.0560
.0160	.0288
.0823	.0895
.1079	.1173
- .0380	- .0400

As can be seen, there is a very good agreement between the theoretical and actual weight vectors. In all cases examined in this research a good agreement between actual and theoretical values were obtained.

5.4 Results Calculated form Theory

This section presents results in which the thoery presented earlier is used to calculate the response of an ABWIN to a set of situations which allow examination of the ABWIN performance.

5.4.1 Signal of Frequency = 1, Power = 1, Direction = 18°

In this section, a single signal of frequency 1 and power 1 is impinging on the array of sensors, arriving from a direction of 18°. Table B.2 below shown the gain of the ABWIN in the direction of the signal as a function of the power of the pilot signal components. Figures B.3 and B.4 show the entire antenna pattern at a frequency of 1 for two of the cases in Table 1. The crosshairs on the figures show the receiving array gain in the direction of signal arrival. The stronger the pilot signal, the relatively weaker the recieved signal is, the more like a signal and less like a powerful jammer it appears to the system. So, the stronger the pilot signal, the lower the notching effect seen by the actual signal of unit power.

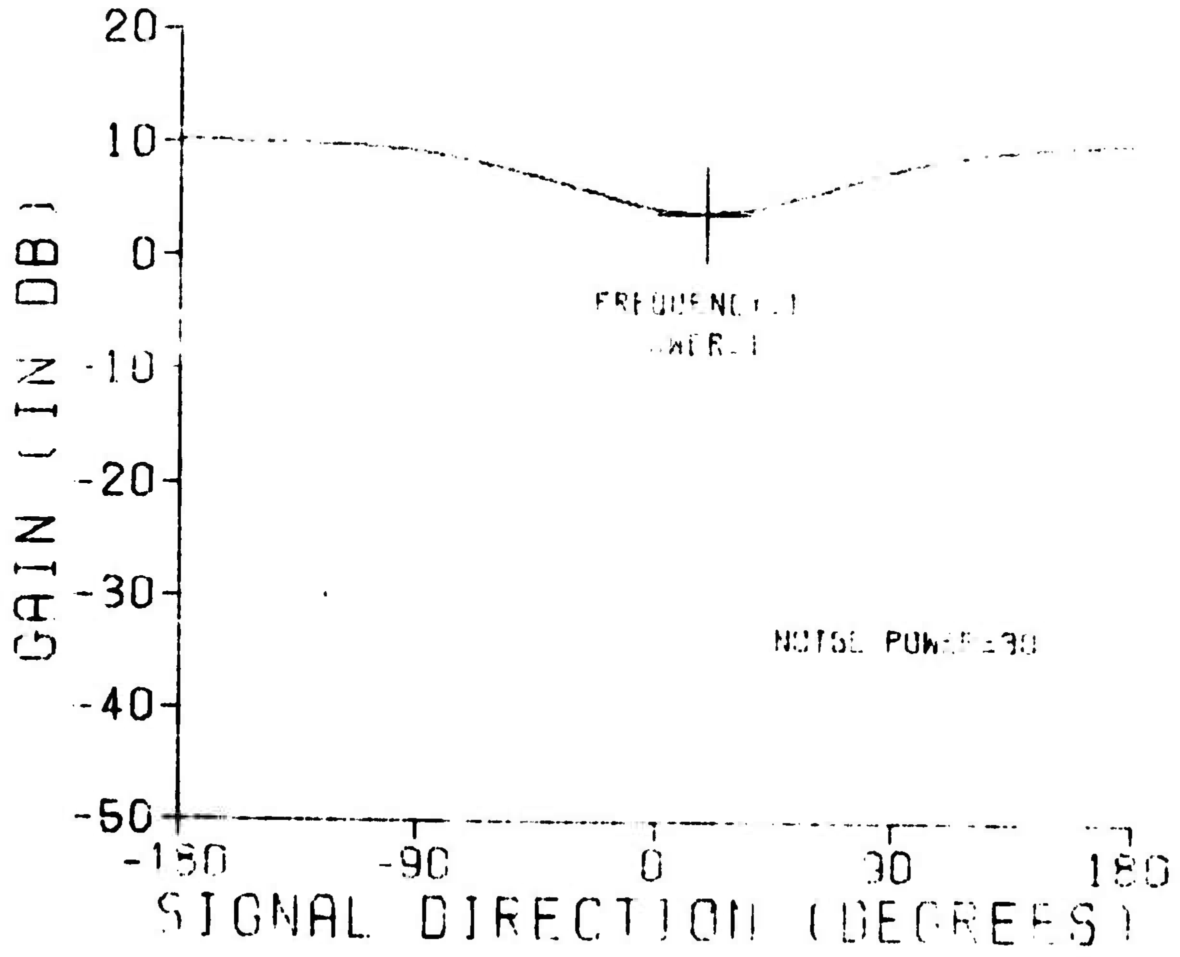


Figure B.3 Signal of Frequency = 1, Power = 1, Direction = 18° with Pilot Signal Component Power = 30.

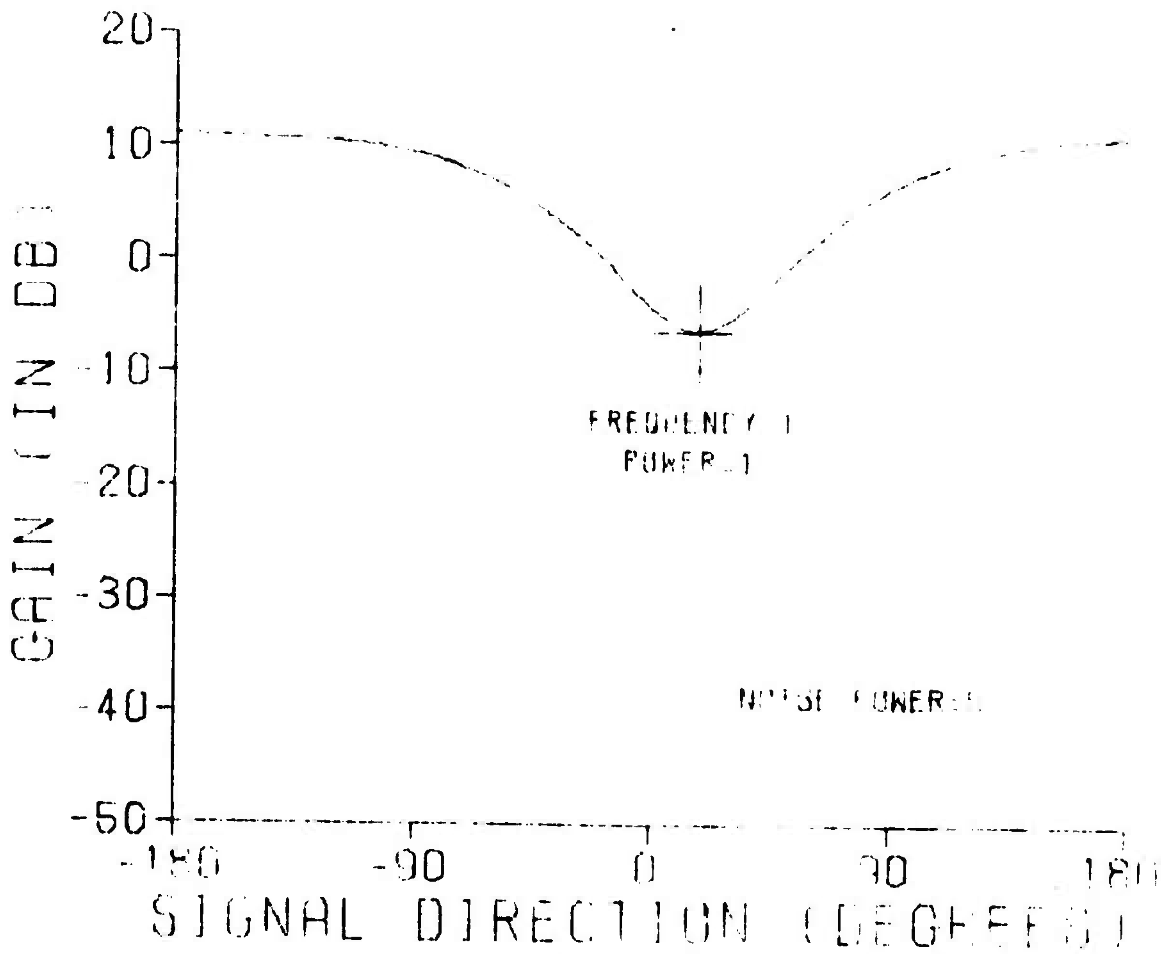


Figure B.4 **Signal of Frequency = 1., Power = 1., Direction = 18° with Pilot Signal Component Power = 5. (Conditions differ from Figure B.3 only in pilot signal component power).**

TABLE B.2

ANTENNA POWER GAIN AT FREQUENCY = 1, DIRECTION = 18°, SIGNAL POWER = 1

Pilot Signal Component Power (σ^2)	Signal Power Gain
100.	5.22
30.	2.48
10.	.694
5.	.239
1.	1.28×10^{-2}
.1	1.38×10^{-3}

5.4.2. Signal of Frequency = 1, Power -10, Direction = 18°

This is the same as the previous situation, except the power of the incoming signal has been increased from 1 to 10. Table B.3 and Figures B.5 and B.6 present the results. Once again, the weaker the pilot signal, the greater the rejection of the received signal. The more powerful recieved signal (of power 10) is more strongly rejected by the adaptive antenna than that of unit power.

5.4.5 Two Signals: Signal 1: Frequency = 1, Power = 1, Direction = 18°

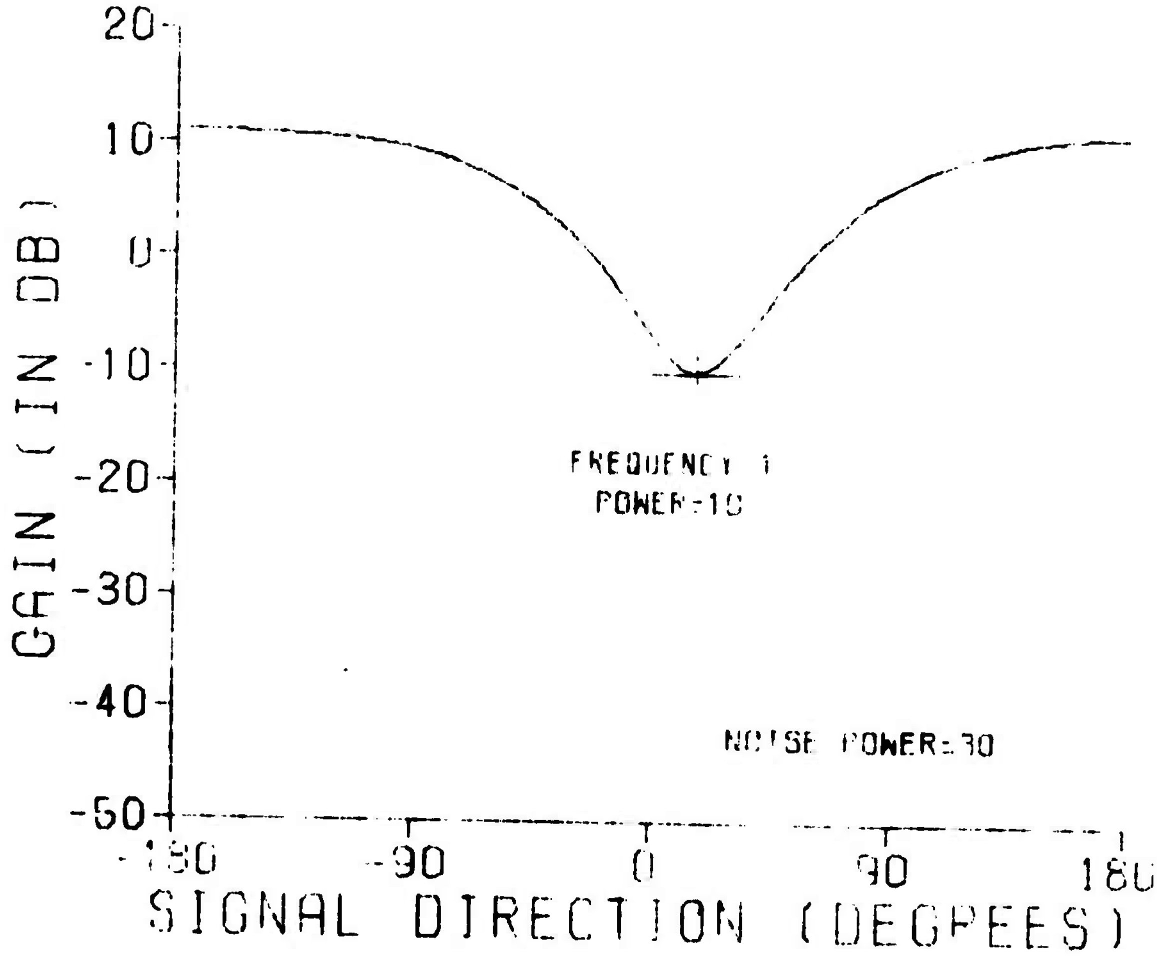


Figure B.5 Signal of Frequency = 1, Power = 10, Direction = 18°, with Pilot Signal Component Power = 30. (Same conditions as Figure B.3 except the received signal is of greater power here).

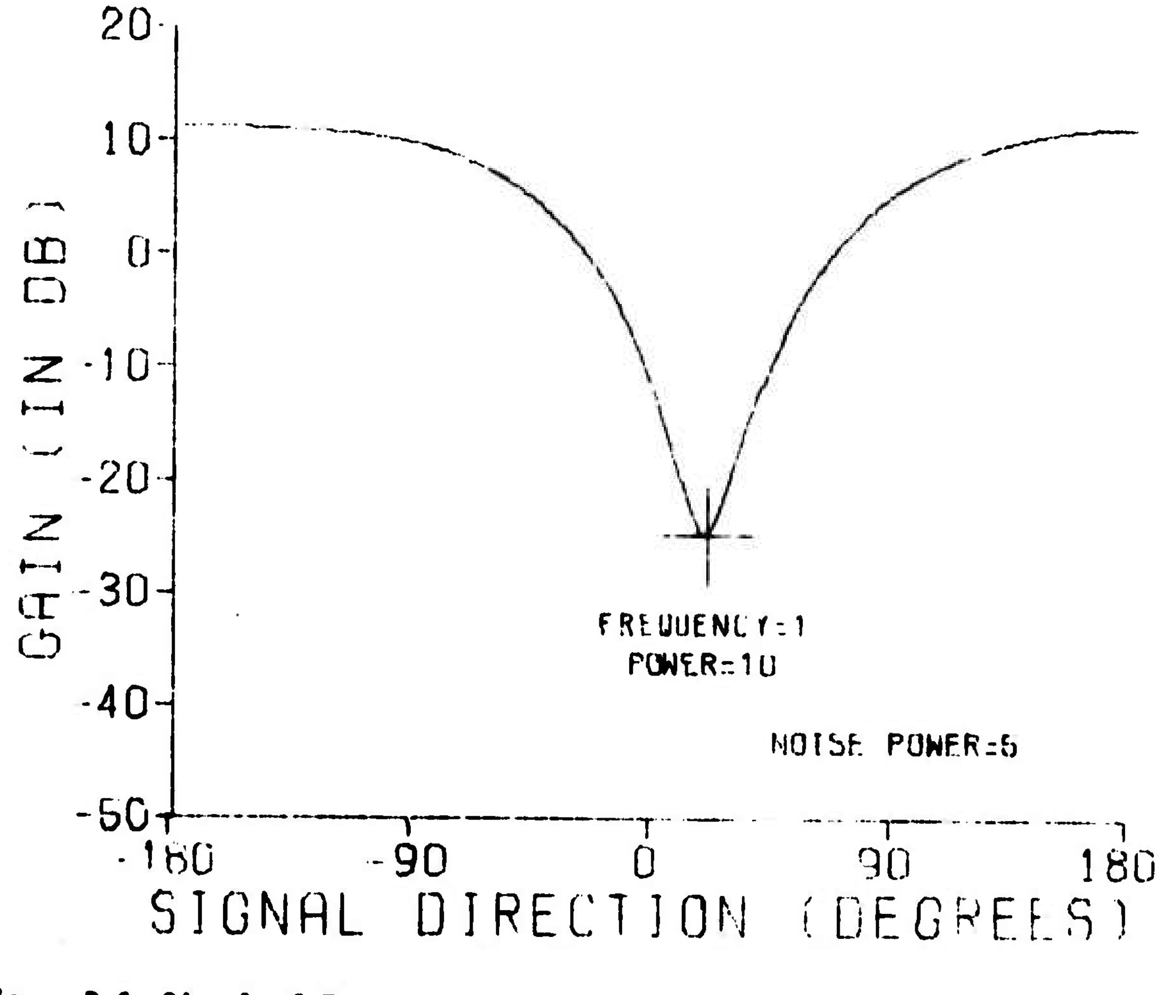


Figure B.6 Signal of Frequency = 1, Power = 10., Direction = 18°, with Pilot Signal Component Power = 5. (Same conditions as Figure B.4 except the received signal is of greater power here. Same conditions as Figure B.5 except pilot signal component power is less here).

TABLE B.3

ANTENNA POWER GAIN AT FREQUENCY = 1, DIRECTION = 18°, SIGNAL POWER = 10

Pilot Signal Component Power (σ^2)	Signal Power Gain
100.	.694
30.	$.991 \times 10^{-1}$
10.	$.128 \times 10^{-1}$
5.	$.334 \times 10^{-2}$
1.	$.138 \times 10^{-3}$
.1	$.139 \times 10^{-5}$

5.4.3 Signal of Frequency = 2, Power = 1, Direction = 108°

Table B.4 shows the gain of the ABWIN to a signal of frequency = 2, direction = 180°, and power = 1. Figures B.7 and B.8 show the entire antenna pattern for the cases of reference signal component power = 30 and 5. The loss of signal is roughly similar to the previous cases (Fig. B.3 and B.4)

5.4.6 Two Signals: Signal 1:

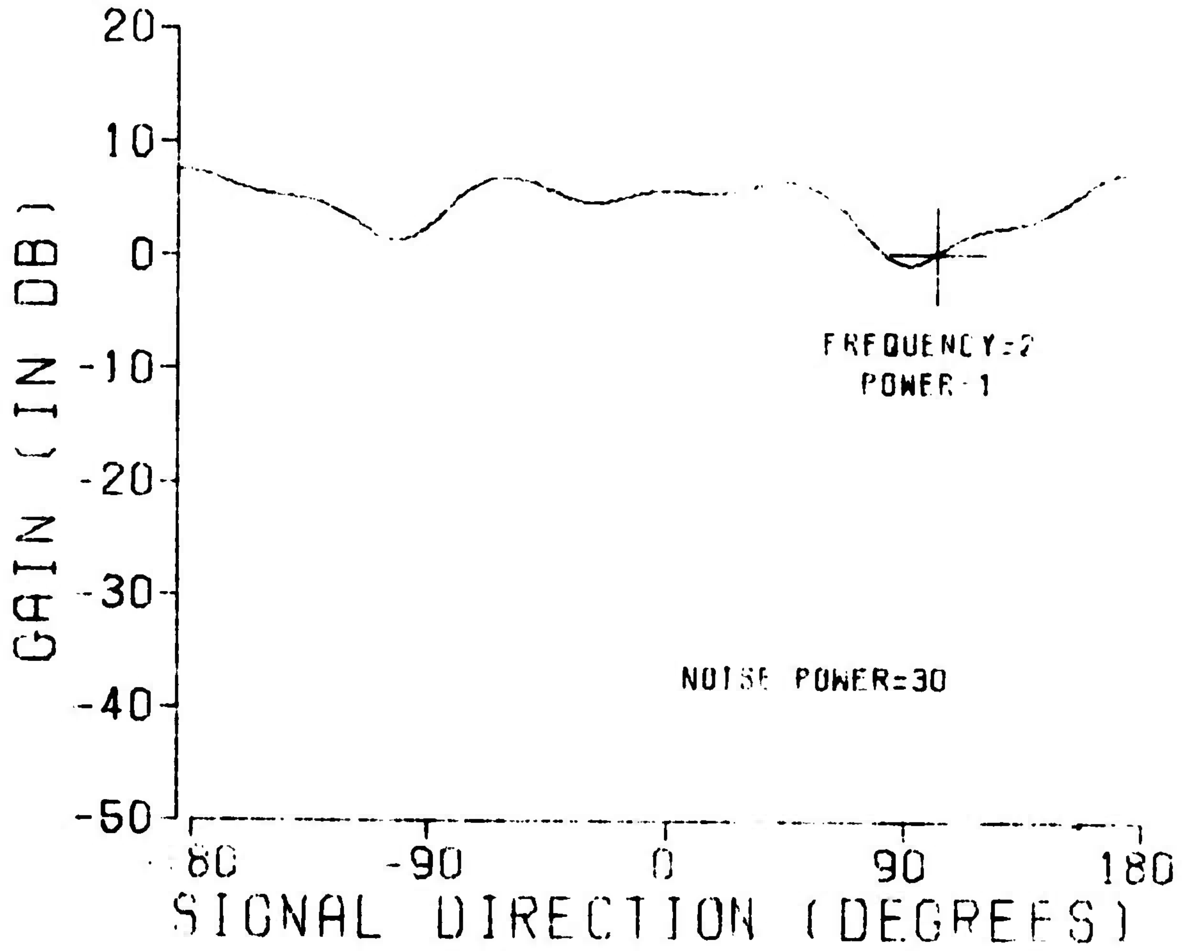


Figure B.7 Signal of Frequency = 2, Power = 1, Direction = 108°, with Pilot Signal Component Power = 30. (Similar conditions to Figure B.3 except signal is of different frequency and direction).

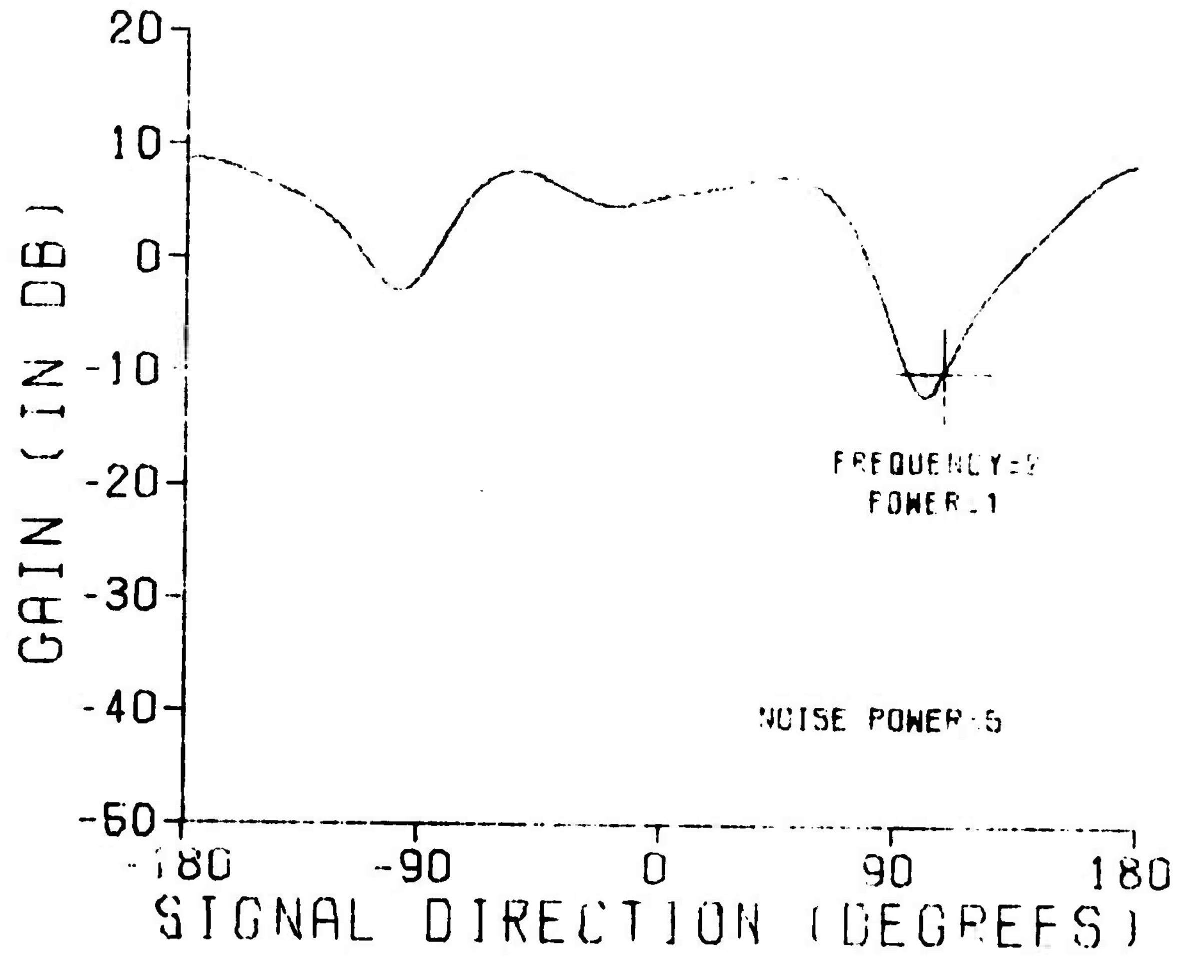


Figure B.8 Signal of Frequency = 2, Power = 1, Direction = 108°, with Pilot Signal Component Power = 5. (Similar conditions to Figure B.4 except signal is of different frequency and direction. Same conditions as Figure B.7 except the pilot signal component power is weaker here).

TABLE B.4

ANTENNA POWER GAIN AT FREQUENCY = 2, DIRECTION = 108°, POWER = 1

Pilot Signal Component Power (σ^2)	Signal Power Gain
100.	2.30
30.	1.09
10.	.306
5.	.105
1.	$.565 \times 10^{-2}$
.1	$.608 \times 10^{-4}$

5.4.4 Signal of Frequency = 2, Power = 10, Direction = 108°

This situation is the same as the previous case except the signal power has been increased from 1 to 10. Stronger signal losses result. Table B.5 and Figures B.9 and B.10 present the measured responses.

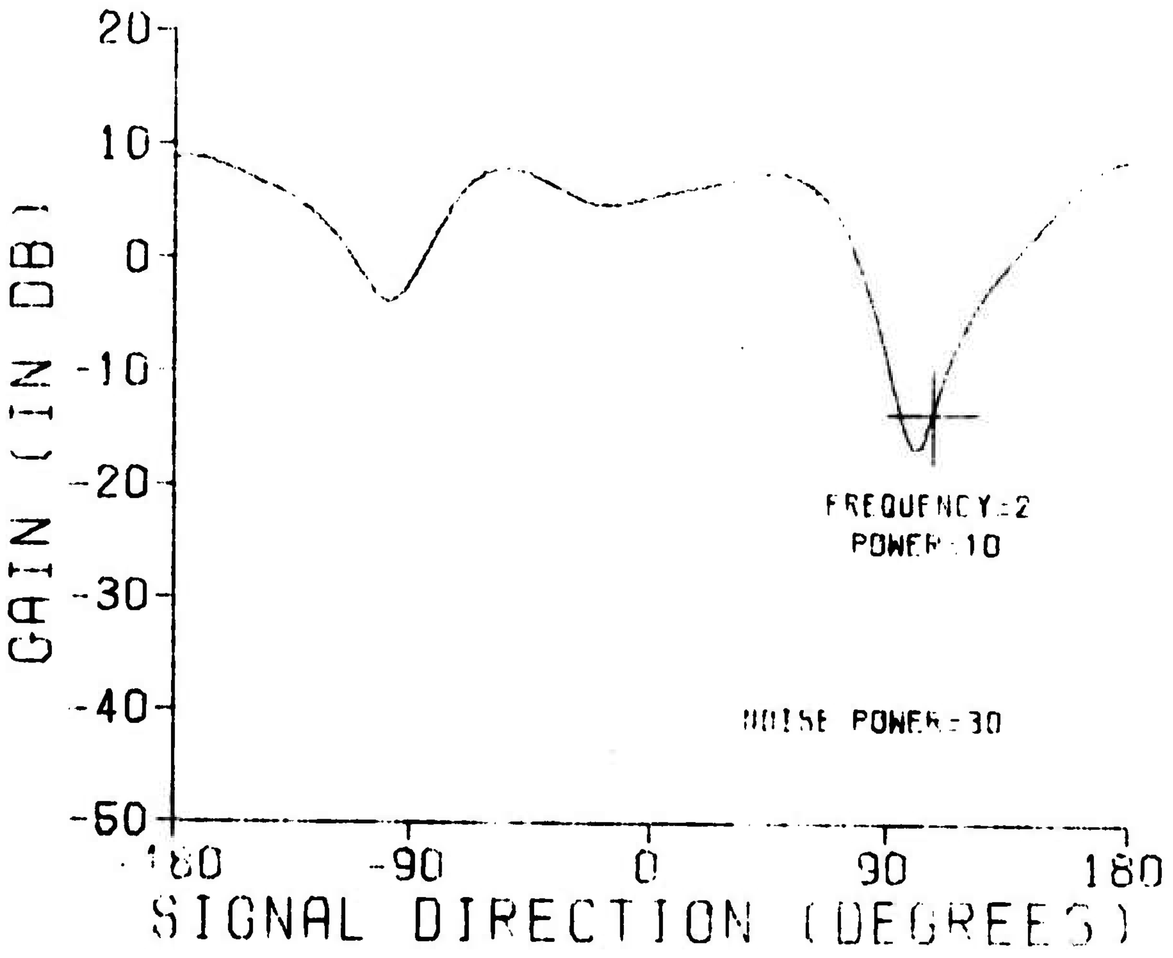


Figure B.9 Signal of Frequency = 2, Power =10, Direction = 108°, with Pilot Signal Component Power = 30. (Similar to conditions of Figure B.5, except signal is of different frequency and direction. Same conditions as Figure B.7 except signal power is greater here.

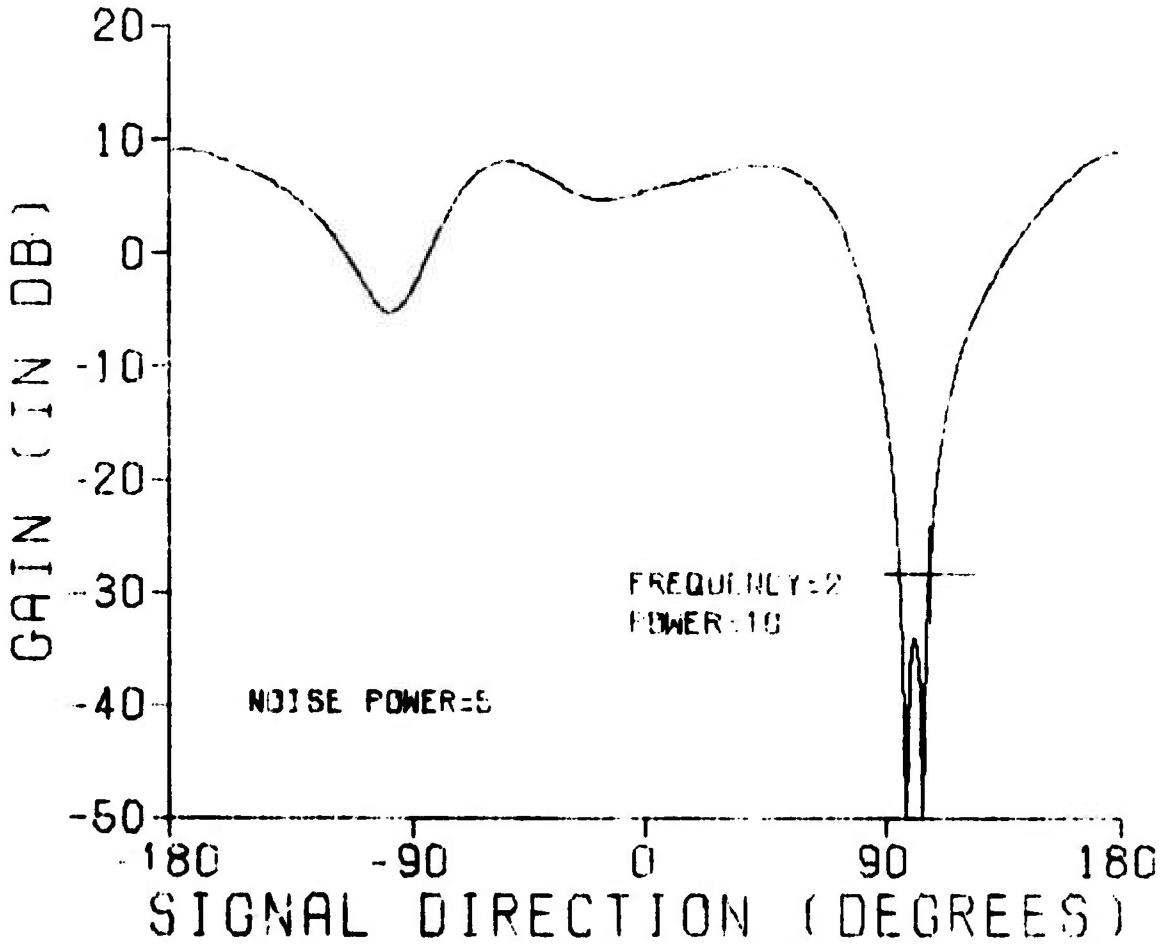
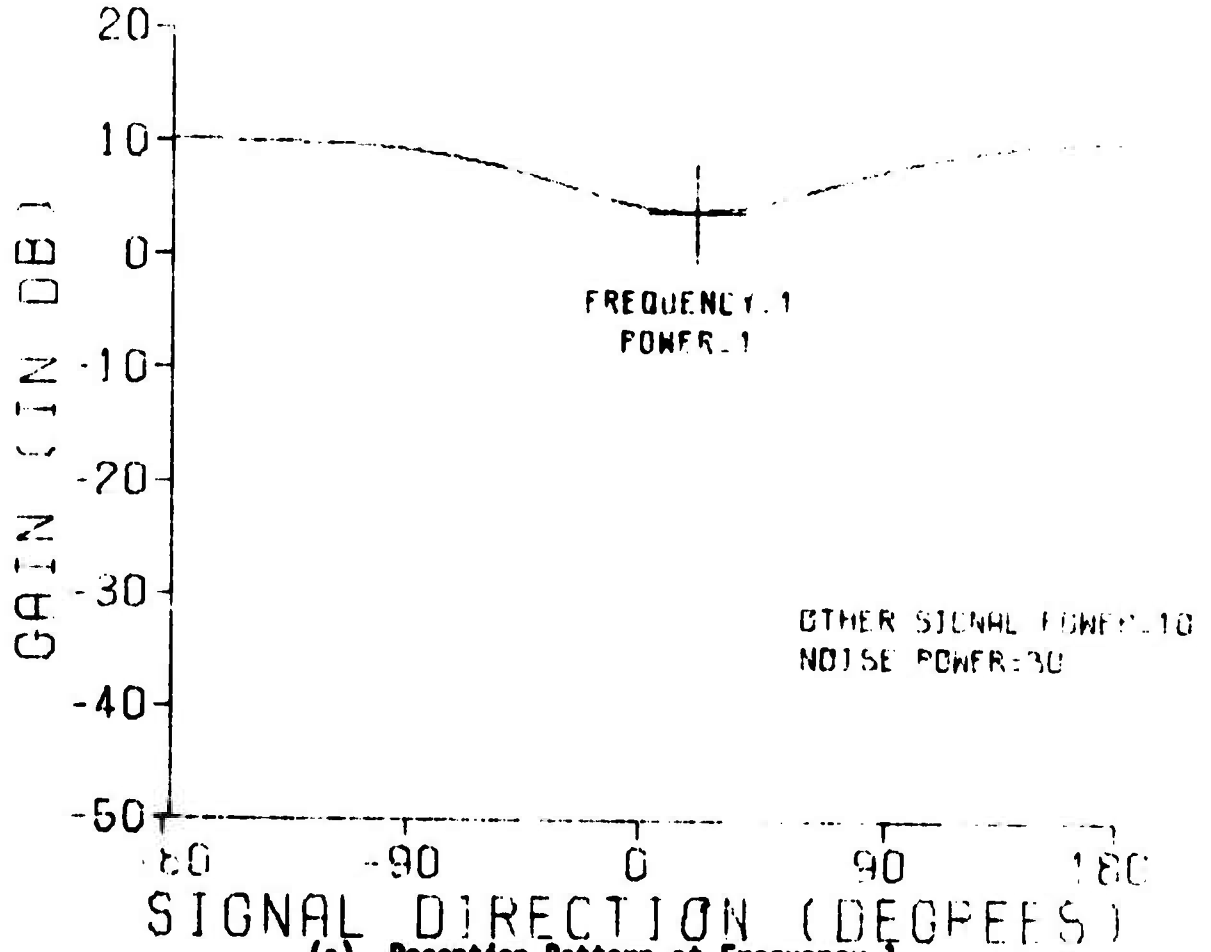
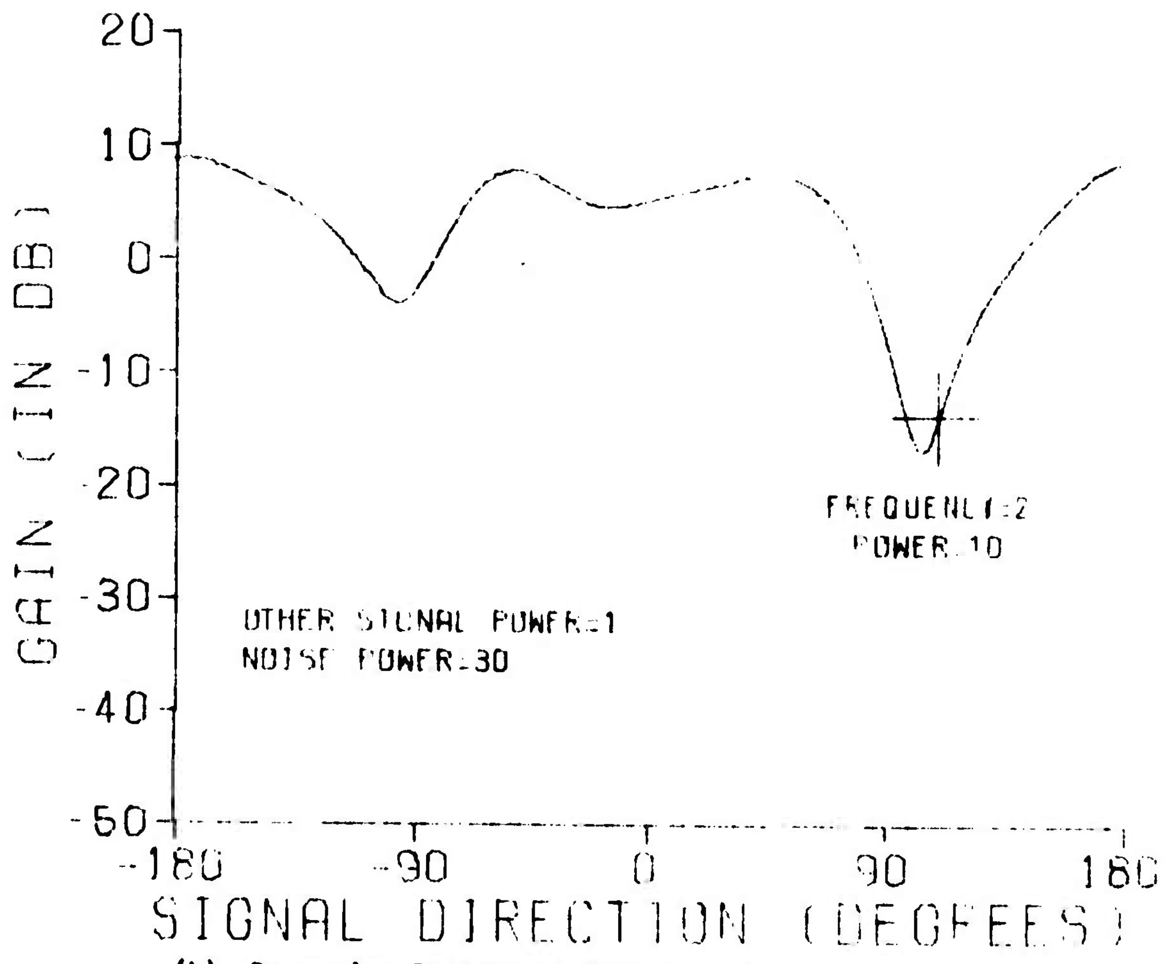


Figure B.10 Signal with Frequency = 2, Power = 10, Direction = 108°, with Pilot Signal Component Power = 5. (Similar to conditions of Figure B.6 except the signal power is greater here. Same conditions as Figure B.9 except the pilot signal component power is less here).



(a) Reception Pattern at Frequency 1

Figure B.11 Two signals
Signal 1: Frequency = 1., Power = 1., Direction = 18°
Signal 2: Frequency = 2., Power = 10., Direction = 108°
with Pilot Signal Component Power = 30.



(b) Reception Pattern at Frequency 2

Figure B.11 (continued)

TABLE B.5

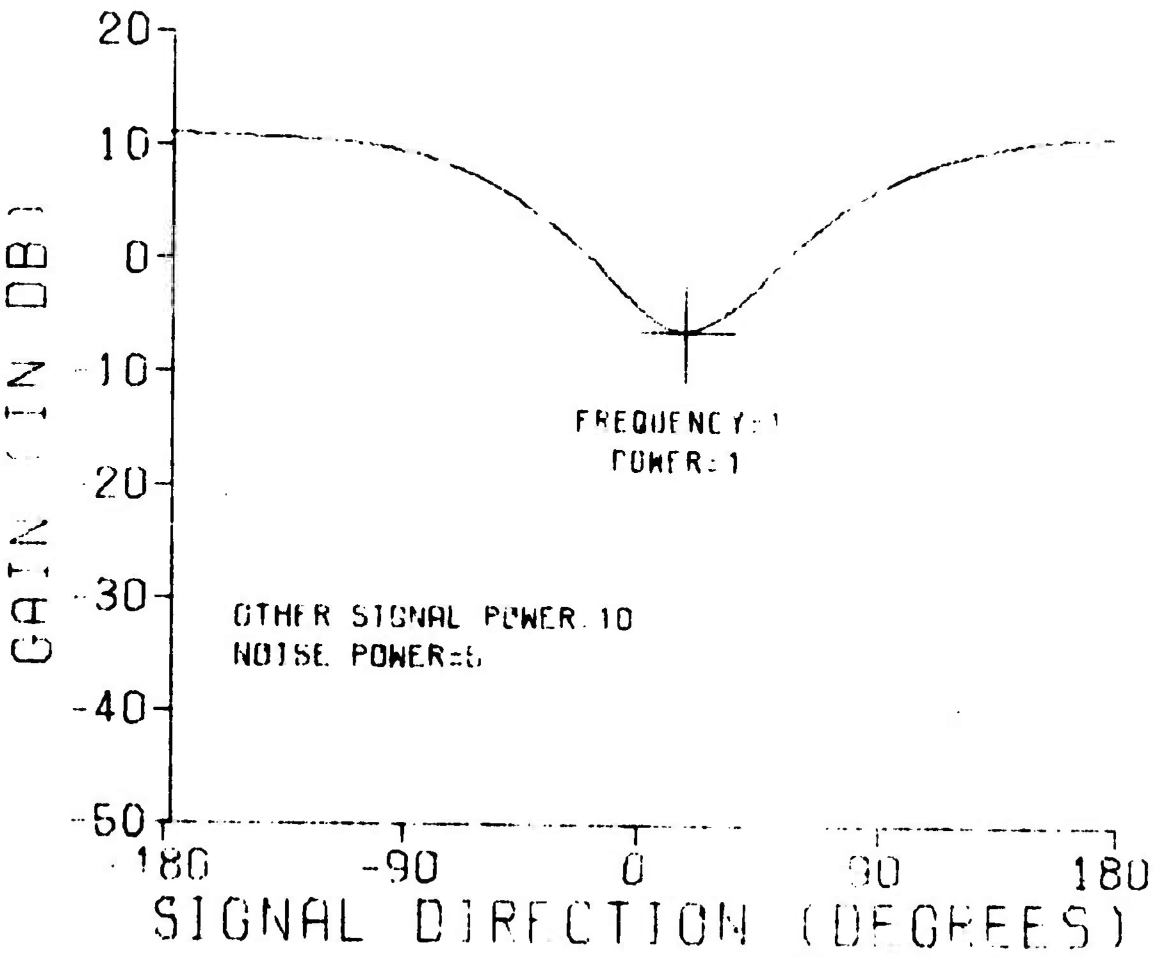
ANTENNA POWER GAIN AT FREQUENCY = 2, DIRECTION = 108°, POWER = 1

Pilot Signal Component Power (σ^2)	Signal Power Gain
100.	.306
30.	$.436 \times 10^{-1}$
10.	$.565 \times 10^{-2}$
5.	$.147 \times 10^{-2}$
1.	$.608 \times 10^{-4}$
.1	$.613 \times 10^{-6}$

5.4.5 Two Signals: Signal 1: Frequency = 1, Power = 1, Direction = 18°
Signal 2: Frequency = 2, Power = 10, Direction = 108°

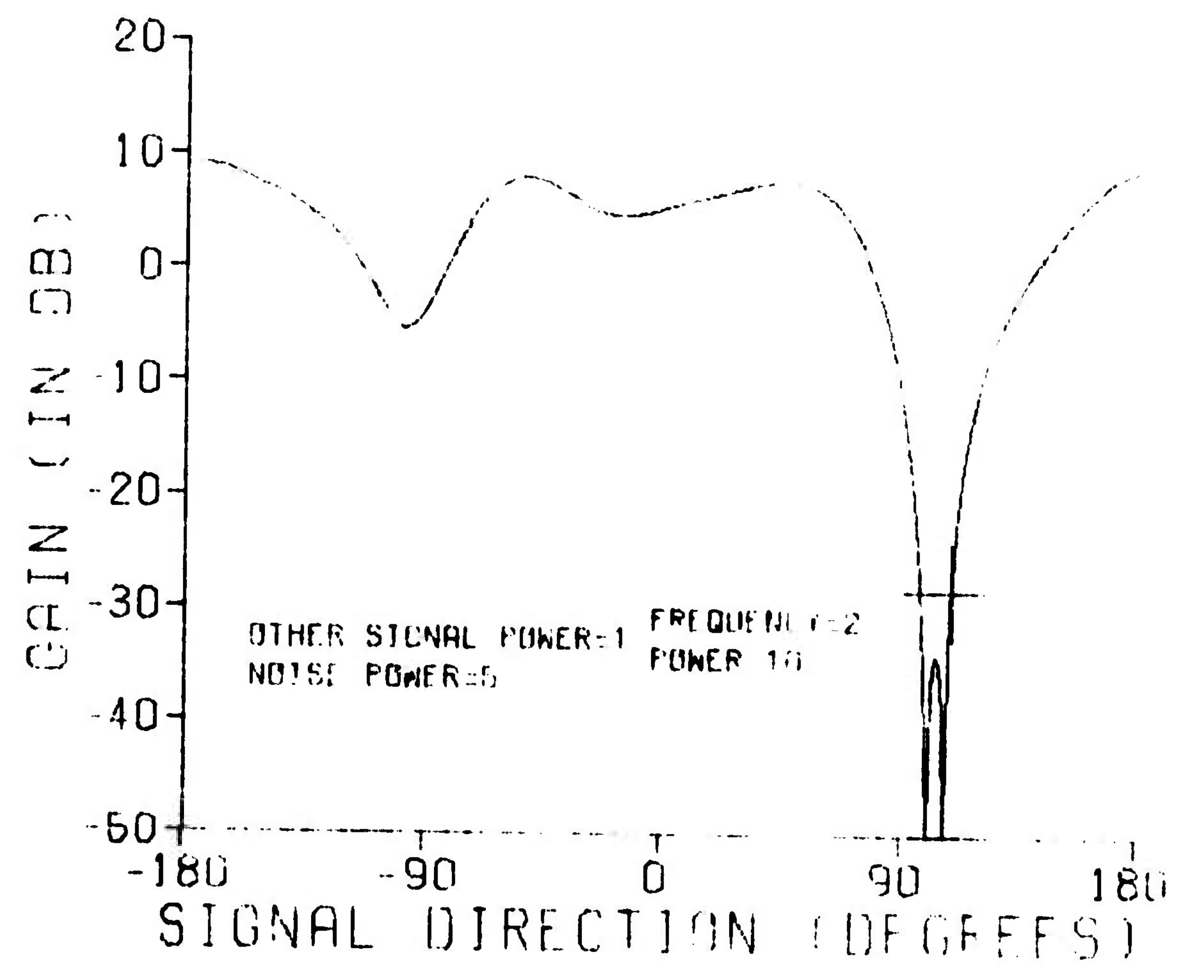
In this situation we have the case of two signals, of different frequency and different direction, where one signal is much stronger than the other. The ABWIN reacts in such a way that, at the output, signal 1 is stronger than signal 2, even though at the input it is the weaker of the two. The phenomenon is similar to "inversion of signal to noise ratios." Table B.6 shows the results, and Figure B.11 shows the antenna patterns at the two frequencies for the case where the pilot signal component power = 30, and Figure B.12 is the analagous figure for the case where the signal component power = 5.

The adaptive system handles the two signals essentially independently, somewhat attenuates the weak signal and strongly attenuates the strong signal.



(a) Reception Pattern at Frequency 1

Figure B.12 Two signals
Signal 1 = Frequency = 1., Power = 1., Direction = 18°
Signal 2 = Frequency = 2., Power = 10., Direction = 108°
with Pilot Signal Component Power = 5.
(Same conditions as Figure B.11 except pilot component power is less here).



(b) Reception Pattern at Frequency 2

Figure B.12 (continued)

TABLE B.6

ANTENNA POWER GAIN IN THE DIRECTION OF TWO SIGNALS:

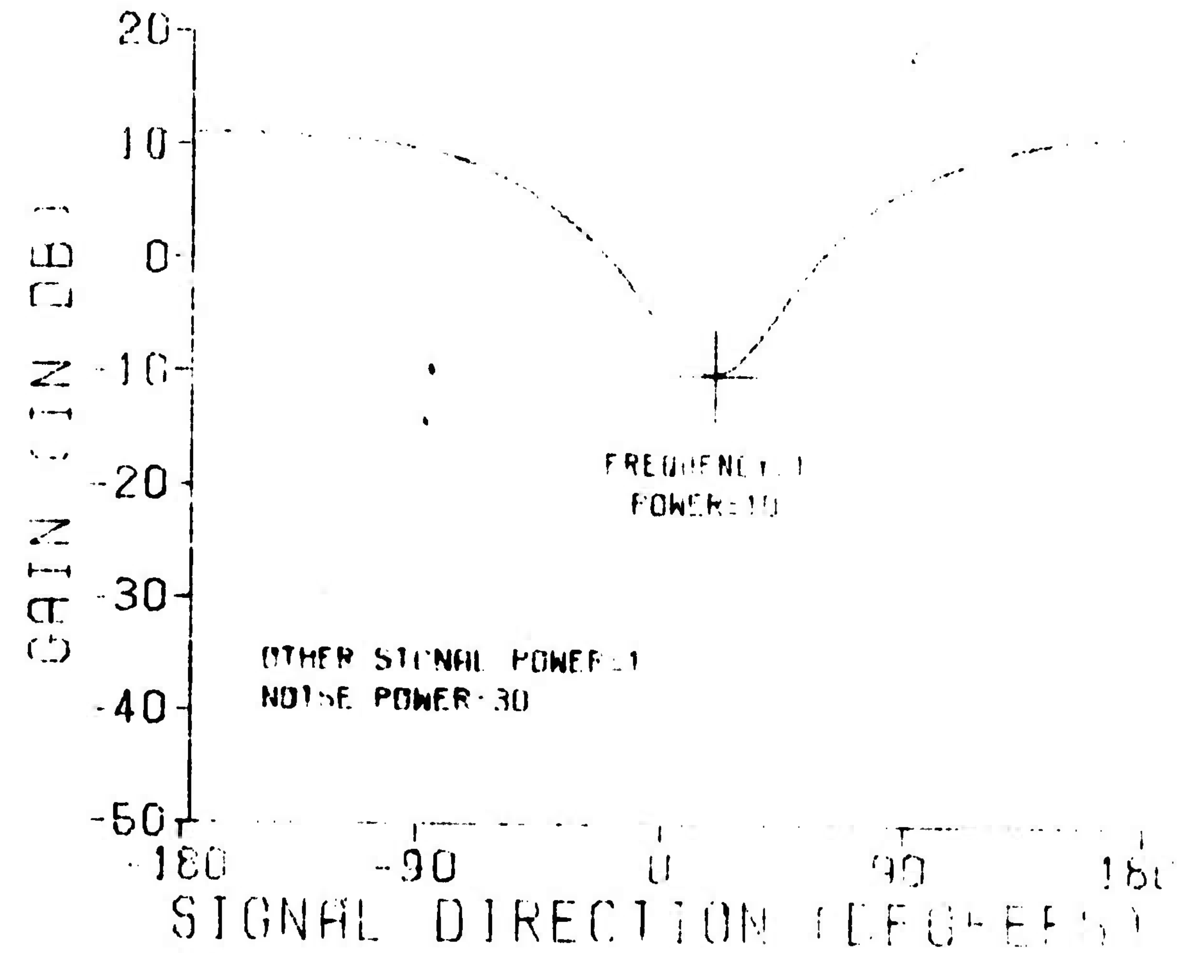
SIGNAL 1: Frequency = 1., Direction = 18°, Power = 1.

SIGNAL 2: Frequency = 2., Direction = 108°, Power = 10.

Pilot Signal Component Power (σ^2)	Signal 1 Power Gain	Signal 2 Power Gain
100.	5.22	.306
30.	2.48	$.436 \times 10^{-1}$
10.	.694	$.565 \times 10^{-2}$
5.	.239	$.147 \times 10^{-2}$
1.	$.128 \times 10^{-1}$	$.608 \times 10^{-4}$
.1	$.138 \times 10^{-3}$	$.614 \times 10^{-6}$

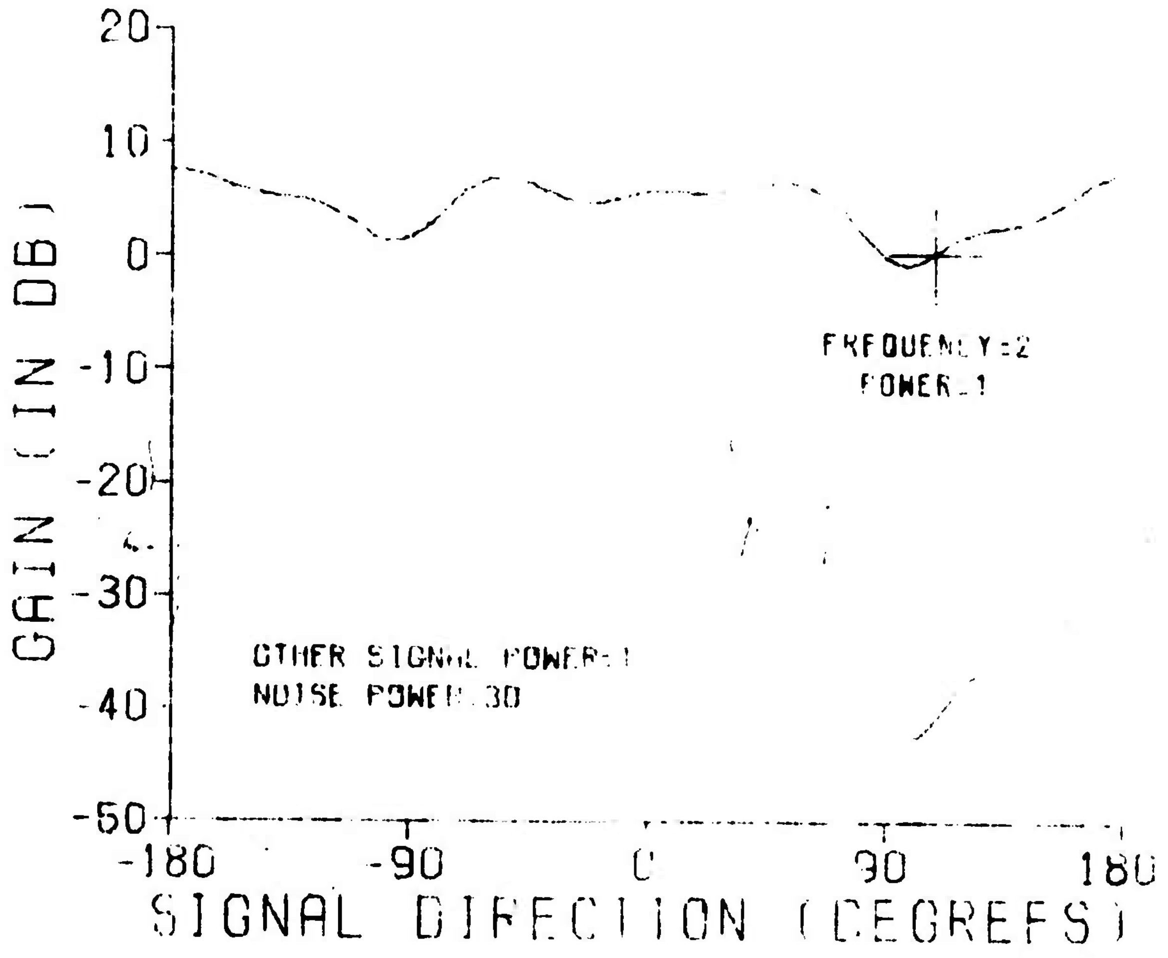
5.4.6 Two Signals: Signal 1: Frequency =1, Power = 10, Direction = 18°
Signal 2: Frequency =2, Power = 1, Direction = 108°

This situation is the same as the previous section except the power levels of the two signals have been interchanged. We see from Table B.7 that the gain of the antenna in the signal directions and frequencies have also switched, resulting once again in greater attenuation for the stronger signal. Figure B.13 shows the antenna patterns at the two frequencies for the case where the reference signal component power = 30., and Figure B.14 is the analgous case for the reference signal component power = 5. Figures B.13 and B.14 may be directly compared with Figures B.9 and B.10 respectively.



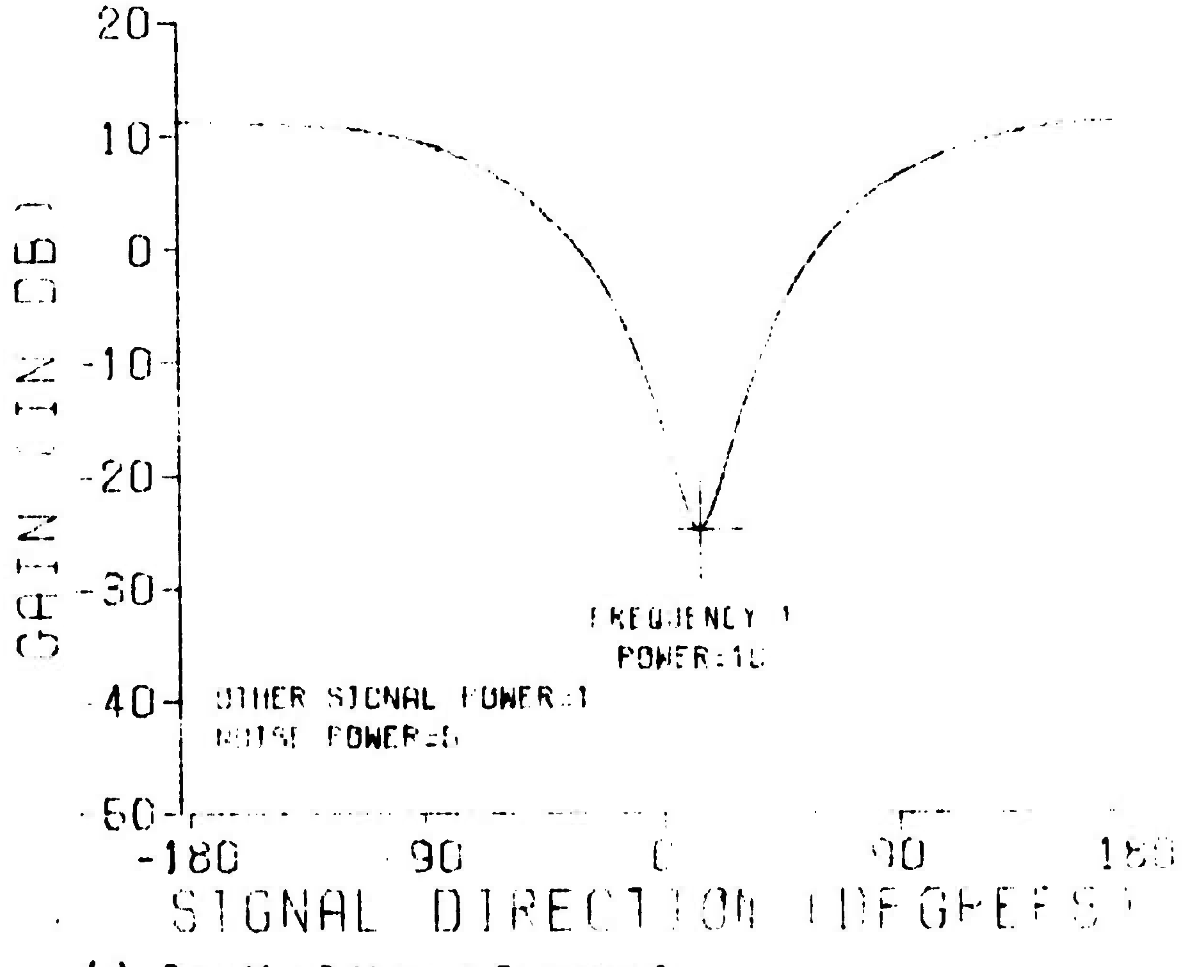
(a) Reception Pattern at Frequency 1

Figure B.13 Two Signals
Signal 1 = Frequency = 1, Power = 1∅, Direction = 18°
Signal 2 = Frequency = 2, Power = 1., Direction = 108°
with Pilot Signal Component Power = 3∅. (Same conditions as Figure B.11 except the signal powers have been interchanged).



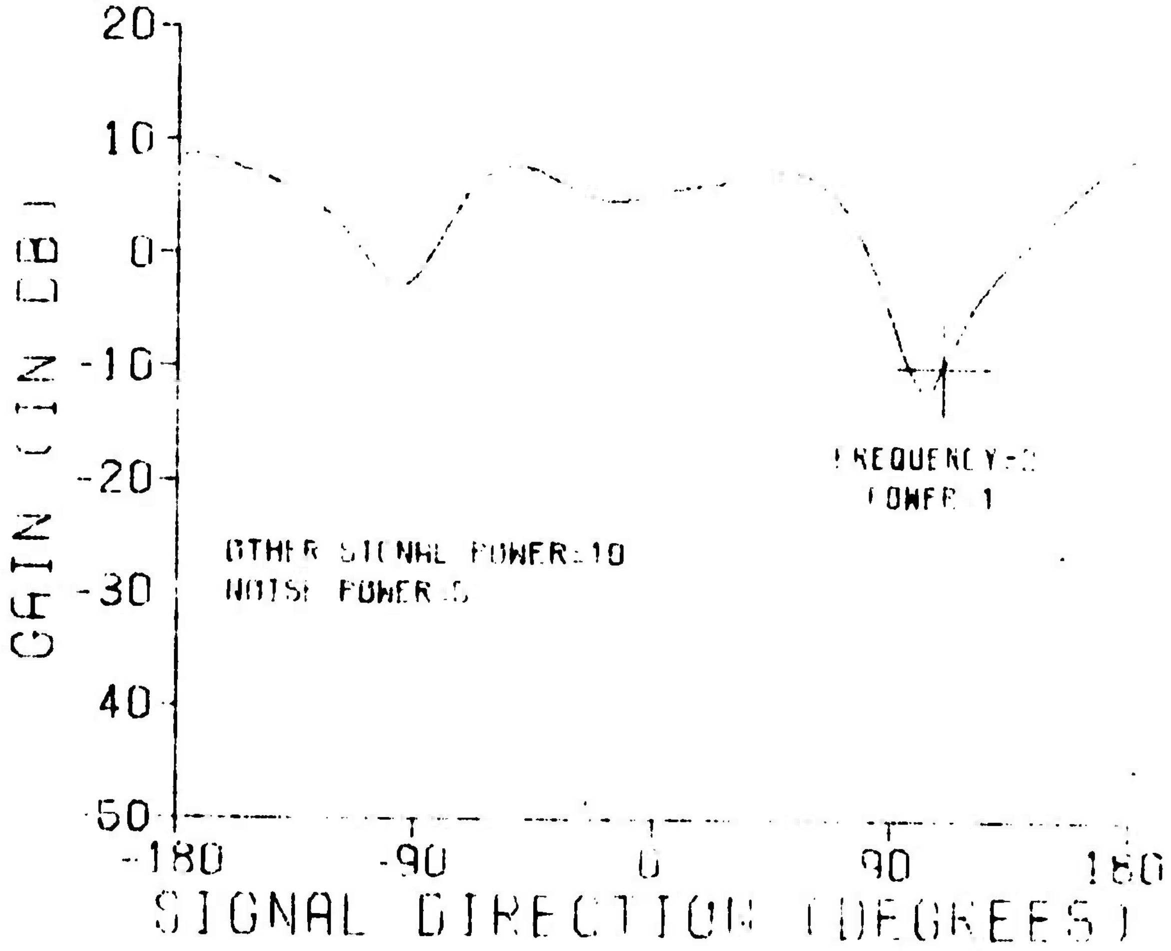
(b) Reception Pattern at Frequency 2

Figure B.13 (continued)



(a) Reception Pattern at Frequency 1

Figure B.14 Two Signals
Signal 1: Frequency = 1, Power = 10, Direction = 18°
Signal 2: Frequency = 2, Power = 1∅, Direction = 108°
with Pilot Signal Component Power = 5
(Same conditions as Figure B.12 execpt power of the two signals have been interchanged. Same conditions as Figure B.13 except the pilot signal component power is less here).



(b) Reception Pattern at Frequency 2

Figure B.14 (continued)

TABLE B.7

ANTENNA POWER GAIN IN THE DIRECTION OF TWO SIGNALS:

Signal 1: Frequency = 1., Direction = 18°, Power = 10.

Signal 2: Frequency = 2., Direction = 108°, Power = 1.

Pilot Signal Component Power (Noise Power)	Signal 1 Power Gain	Signal 2 Power Gain
100.	.694	2.30
30.	$.991 \times 10^{-1}$	1.09
10.	$.128 \times 10^{-1}$	.306
5.	$.334 \times 10^{-2}$	.105
1.	$.138 \times 10^{-3}$	$.505 \times 10^{-2}$
.1	$.139 \times 10^{-5}$	$.608 \times 10^{-4}$

5.5 Summary of the Simulations

From section 5.4 we see that the pilot signal component power has a direct effect on the attenuation an external signal experiences. An important parameter is ratio of signal power to pilot power. The two incident signals used in the simulations were handled essentially independently by the system. Figure B.15 illustrates how the recieved signals are attenuated as their power increases. Weak signals pass while strong jammers are attenuated.

The results of section 5.5 supports the goal of ABWIN response to a signal on the basis of its power level. In the examples given, the effect is so strong that the relative power levels of the signals at the output of the array system is reversed from that at the input.

Figure B.15 is a summary figure which shows the relationship between the gain of the ABWIN and the ratio of the signal power to the pilot signal power. From this figure we clearly see the effect the pilot signal power has on the ABWIN's response to a signal. (The data points for the signal of frequency 1 was extracted from Tables B.2, B.3, B.6, and B.7. Similarly, the data for the signal of frequency 2 was extracted from Tables B.4, B.5, B.6, and B.7).

6. Modifying the ABWIN Pilot Signal

In previous sections the assumption was made that the pilot signal was constructed by summing the individual pilot signal components (n_i). It was shown that this resulted in a particular quiescent weight vector, which, with the sensor geometry, determined the quiescent reception pattern for the system. A problem exhibited itself in that the quiescent pattern was

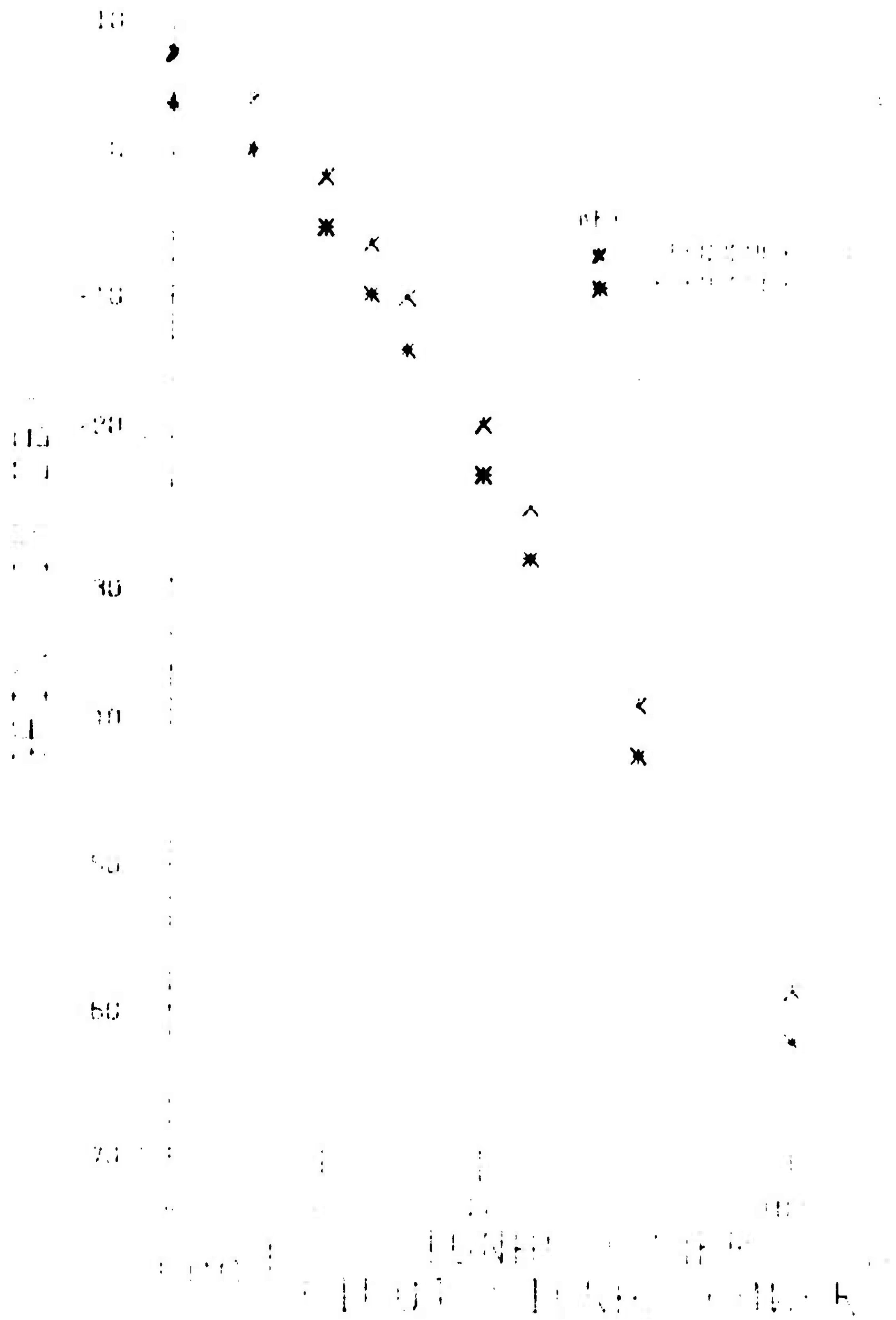


Figure B.15 ABWIN gain as a function of signal to pilot signal power ratio.

the same pattern obtained by summing the antenna element outputs. In many situations the pattern so obtained may be unacceptable for the application.

By modifying the formation of the pilot signal, the quiescent pattern may be modified, and thus the sensor geometry may be taken into consideration.

Consider the quiescent pattern as discussed in section 4. In that section, it was proven that $\overline{W} = \frac{1}{\sigma^2} P$, where $P = E\{d(k)U(k)\}$, the correlation between the pilot signal and the contents of the tapped delay lines. Since it was assumed the pilot signal components (n_i) were uncorrelated with the antenna element outputs, we see that the only significant contents of the tapped delay lines are delayed samples of the pilot signal components. Thus, by changing the pilot signal $d(k)$ to correlate differently with $U(k)$, the quiescent weight vector $\overline{W}$ can be altered.

The method proposed here for modifying $d(k)$ is to allow $d(k)$ to include delayed samples of the $n_i(k)$. This can be accomplished by passing the $n_i(k)$ through a set of transversal filters, where each weight has as a value the desired correlation of $d(k)$ with the corresponding element in $U(k)$.

As an example, let us take a case of two sensors, each with two taps. If the pilot signal is formed as originally described, we have

$$d(k) = n_1(k) + n_2(k)$$

$$U(k) = \begin{bmatrix} n_1(k) \\ n_1(-1) \\ \hdashline n_2(k) \\ n_2(k-1) \end{bmatrix}$$

Then $P = E\{d(k)U(k)\} = E \begin{bmatrix} n_1^2(k) + n_2(k)n_1(k) \\ n_1(k) + n_1(k-1) + n_2(k)n_1(k-1) \\ \hdashline n_2^2(k) + n_1(k)n_2(k) \\ n_1(k)n_2(k-1) + n_2(k)\ n_2(k-1) \end{bmatrix}$

Since n_1 and n_2 are independent, white noise sources of zero mean and variance σ^2 we obtain:

$$P = \begin{bmatrix} \sigma^2 \\ 0 \\ \hdashline \sigma^2 \\ 0 \end{bmatrix} \quad \text{, so } \overline{W} = \frac{1}{\sigma^2} P = \begin{bmatrix} 1 \\ 0 \\ \hdashline 1 \\ 0 \end{bmatrix} \quad \text{, as expected.}$$

Now suppose that form consideration of sensor geometry we wanted

$$\overline{W} = \begin{bmatrix} a \\ b \\ \hdashline c \\ d \end{bmatrix}$$

To do this, we would form the pilot signal as follows:

$$d(k) = an_1(k) + bn_1(k-1) + cn_2(k) + dn_2(k-1)$$

Then

$$P = E \begin{bmatrix} an_1^2(k) + bn_1(k-1)n_1(k) + cn_2(k)\ n_1(k) + dn_2(k-1)n_1(k) \\ an_1(k)n_1(k-1) + bn_1^2(k-1) + cn_2(k)n_1(k-1) + dn_2(k-1)n_1(k-1) \\ \hdashline an_1(k)n_2(k) + bn_1(k-1)n_2(k) + cn_2^2(k) + dn_2(k-1)n_2(k) \\ an_1(k)n_2(k-1) + bn_1(k-1)n_2(k-1) + cn_2(k)n_2(k-1) + dn_2^2(k-1) \end{bmatrix}$$

$$= \begin{bmatrix} a\sigma^2 \\ b\sigma^2 \\ c\sigma^2 \\ d\sigma^2 \end{bmatrix}$$

so $$\bar{W} = \frac{1}{\sigma^2} P = \begin{bmatrix} a \\ b \\ --- \\ c \\ d \end{bmatrix} ,$$ as desired.

Thus this method of formation of the pilot signal allows control over the quiescent pattern of the system by choice of a suitable quiescent weight vector.

7. A Proposal for Modification of the ABWIN Algorithm

In view of the preceding analysis of the ABWIN, particularly of the advantages of choosing a quiescent weight vector, and many similarities to the ALEWIN described in part A of this report, a modification to the ABWIN is proposed here.

In section A of this report, after the ALEWIN is introduced, a second type of line enhancer, the ALEWAIN is introduced. The similarities in performance between the ALEWIN and the ALEWAIN are demonstrated. The ALEWAIN (using the "leaky" LMS algorithm) has the characteristic that in the absence of any excitation (inputs), the weight vector collapses, of "relaxes" to zero. In the ABWIN, we see similar behavior in that in the absence of external excitation, the weight vector returns to its quiescent value. On the basis of this resemblance, the following algorithm is proposed: run the adaptive array as discussed before, but without the pilot signal noise components added to the sensor outputs. As the error signal, use the negative of the system output. Then use the following rule for updating the weight vector:

$$\bar{W}(k+1) = W(k) + 2\mu e(k)U(k) - 2\mu\gamma(\tilde{W}-W(k))$$

where γ is a constant to be adjusted, $\tilde{W}$ is the quiescent weight vector.

Now a term explictly causIng a relaxatlon effect is included, the constant γ controlling the magnitude of the relaxation effect.

Preliminary studies Indicate that this algorithm has the desired features of the ABWIN, but in addition does not require a "parallel" system for computation of the system output $\tilde{y}$ without the corrupting pilot signal does not require a complicated scheme for generating the desired quiescent pattern, and generates less adaptation noise in the weights (thus enablIng faster convergence).

8. Conclusions

The Adaptive Beamformer With Injected Noise has been introduced, and some analysis has been undertaken. Simulations of the ABWIN have been shown to agree wlth the theoretical results. The formation of the pilot signal to obtain a desired quiescent response has been discussed, and a new method to accomplish the same goals as the ABWIN with a simpler algorithm has been proposed for study.

The effect of sensor geometry on the capabilities of the ABWIN has not yielded to analysis at thls time, and is likely to be a problem in future studles of all antenna arrays, adaptive and otherwlse.

It is suggested that the study of the new algorlthm proposed above be pursued, with the intent of camparing its performance to that of the ABWIN, and extending analysis of both algorithms further than presented herein. In addltion, the study of the effect of sensor geometry on the capabilltles of these systems should be continued as a background activity.

A Comparison of Adaptive Algorithms Based on the Methods of Steepest Descent and Random Search

BERNARD WIDROW, FELLOW, IEEE, AND JOHN M. McCOOL, SENIOR MEMBER, IEEE

Abstract—This paper compares the performance characteristics of three algorithms useful in adjusting the parameters of adaptive systems: the differential (DSD) and least-mean-square (LMS) algorithms, both based on the method of steepest descent, and the linear random search (LRS) algorithm, based on a random search procedure derived from the Darwinian concept of "natural selection." The LRS algorithm is presented here for the first time. Analytical expressions are developed that define the relationship between rate of adaptation and "misadjustment," a dimensionless measure of the difference between actual and optimal performance due to noise in the adaptive process. For a fixed rate of adaptation it is shown that the LMS algorithm, which is the most efficient, has a misadjustment proportional to the number of adaptive parameters, while the DSD and LRS algorithms have misadjustments proportional to the square of the number of adaptive parameters. The expressions developed are verified by computer simulations that demonstrate the application of the three algorithms to system modeling problems, of the LMS algorithm to the cancelling of broadband interference in the sidelobes of a receiving antenna array, and of the DSD and LRS algorithms to the phase control of a transmitting antenna array. The second application introduces a new method of constrained adaptive beamforming whose performance is not significantly affected by element nonuniformity. The third application represents a class of problems to which the LMS algorithm in the basic form described in this paper is not applicable.

Manuscript received November 5, 1975; revised April 15, 1976.
B. Widrow is with the Information Systems Laboratory, Department of Electrical Engineering, Stanford University, Stanford, CA 94305.
J. M. McCool is with the Fleet Engineering Department, Naval Undersea Center, San Diego, CA 92132.

I. Introduction

THE APPLICATION of adaptive techniques has allowed development during the past fifteen years of high-performance receiving antennas with a capability of automatically eliminating sidelobe interference. In such antennas the main beam is steered in a predetermined direction in search of expected signals, while interference received outside the main beam causes the formation of nulls in the radiation pattern [1]–[10]. New types of adaptive antennas are also currently being designed that will automatically seek and track desired signals. This application promises a further significant enhancement of antenna capabilities.

Many adaptive antenna systems are configured by connecting the elements of an antenna array to a multichannel adaptive filter. In its general form an adaptive filter is a device that adjusts its internal parameters and optimizes its performance according to the statistical characteristics of its input and output signals. The internal filter adjustment is made through a series of variable settings controlled by an adaptive algorithm.

The purpose of this paper is to analyze and compare the properties of certain algorithms available for use with adaptive filters. Two basic methods of adaptation are considered, those of steepest descent and random search. Theoretical performance comparisons of algorithms based on these methods, including the Widrow–Hoff LMS algorithm and a new linear random search algorithm, are made by relating quality of solution to speed of adaptation. Results of computer simulations are presented to provide experimental confirmation of the theoretically predicted performance of the algorithms and to illustrate their use in adaptive antenna applications.

II. Characteristics and Terminology of the Adaptive Process

The theoretical analyses of this paper are based on the particular form of adaptive transversal filter illustrated in Fig. 1. This finite impulse response (FIR) filter consists of a tapped delay line connected to an adaptive linear combiner that adjusts the gain of (or "weights") the signals derived from the delay line and combines them to form an output signal.[1] All of the algorithms described in this paper can be used to govern the operation of the adaptive linear combiner; the LMS algorithm is restricted to this use.

The input signal vector X_j of the adaptive linear combiner is defined as

$$X_j^T \triangleq [x_{1j}\ x_{2j} \cdots x_{nj}]^T. \tag{1}$$

The input signal components are assumed to appear simultaneously on all input lines at discrete times indexed by the subscript j. The weighting coefficients or multiplying factors $w_1, w_2, \cdots, w_n$ are adjustable, as symbolized in Fig. 1 by circles with arrows through them. The weight vector W is

$$W^T \triangleq [w_1\ w_2 \cdots w_n]^T. \tag{2}$$

The output y_j is equal to the inner product of X_j and W:

$$y_j = X_j^T W = W^T X_j. \tag{3}$$

The error ε_j is defined as the difference between the desired response d_j (an externally supplied input sometimes called the "training signal") and the actual response y_j:

$$\varepsilon_j \triangleq d_j - X_j^T W = d_j - W^T X_j. \tag{4}$$

In adaptive antenna systems the desired response may be derived by various methods, one of which is to inject a "pilot signal" whose characteristics determine the "look" direction and frequency response of the main beam [4]. Other methods are illustrated in Section VI.

It is the purpose of the adaptive process to adjust the weights of the adaptive linear combiner to minimize the mean square of the error ε_j. Let the input signals X_j and desired response d_j be statistically stationary. During adaptation the weight vector varies, so that even with stationary inputs the output y_j and error ε_j will generally be nonstationary. Care must thus be taken in defining the mean square error for an adaptive system. The only possibility is an ensemble average, which can be established in the following manner.

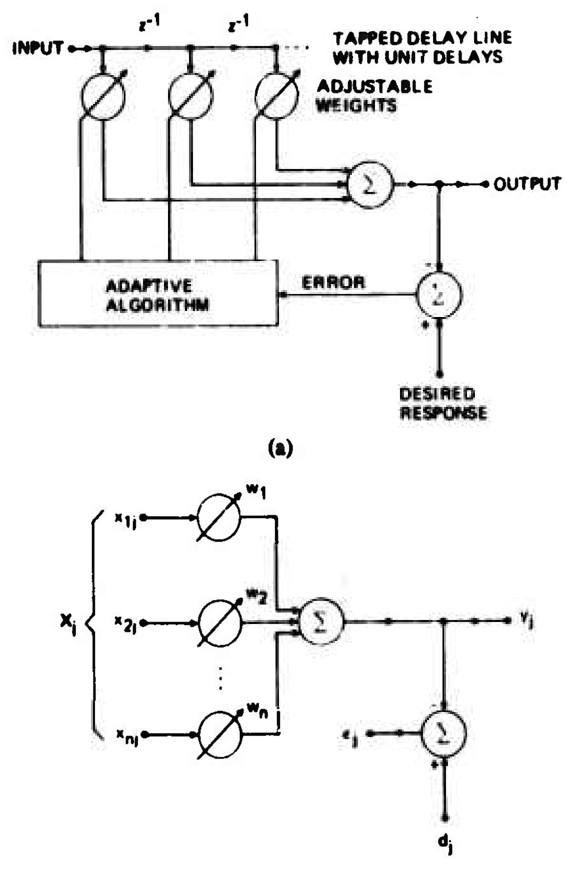


Fig. 1. Adaptive filter consisting of tapped delay line connected to adaptive linear combiner. (a) Adaptive filter configuration. (b) Adaptive linear combiner with input and output terminology.

[1] The adaptive linear combiner is "linear" only when the weighting coefficients are fixed; adaptive systems, like all systems whose characteristics change with those of their inputs, are by nature nonlinear.

The adaptive process progresses recursively or by iterative cycles. At the kth iteration let the weight vector be W_k. Squaring and expanding (4) and letting $W = W_k$ yields

$$\varepsilon_j^2 = d_j^2 - 2d_jX_j^TW_k + W_k^TX_jX_j^TW_k. \quad (5)$$

Now assume an ensemble of identical adaptive linear combiners, each having the same weight vector W_k at the kth iteration. Let each combiner have individual inputs X_j and d_j derived, respectively, from stationary ergodic ensembles. Each combiner will produce an individual error ε_j represented by (5). Averaging (5) over the ensemble yields

$$E[\varepsilon_j^2]_{W=W_k} = E[d_j^2] - 2E[d_jX_j^T]W_k + W_k^TE[X_jX_j^T]W_k. \quad (6)$$

Defining the vector P as the cross correlation between the desired response (a scalar) and the X-vector then yields

$$P^T \triangleq E[d_jX_j^T] = E[d_jx_{1j}\; d_jx_{2j} \cdots d_jx_{nj}]^T. \quad (7)$$

The input correlation matrix R is defined in terms of the ensemble average

$$R \triangleq E[X_jX_j^T] = E\begin{bmatrix} x_{1j}x_{1j} & x_{1j}x_{2j} & \cdots & \\ x_{2j}x_{1j} & x_{2j}x_{2j} & \cdots & \\ \vdots & \vdots & & \\ & & \cdots & x_{nj}x_{nj} \end{bmatrix}. \quad (8)$$

This matrix is real, symmetric, and positive definite, or in rare cases positive semi-definite. The mean square error ξ_k can thus be expressed as

$$\xi_k \triangleq E[\varepsilon_j^2]_{W=W_k} = E[d_j^2] - 2P^TW_k + W_k^TRW_k. \quad (9)$$

Note that the mean square error is a quadratic function of the weights that can be pictured as a concave hyperparaboloidal surface, a function that never goes negative. Adjusting the weights involves descending along this surface with the objective of reaching its unique minimum point ("the bottom of the bowl" [11]). Gradient methods are commonly used for this purpose.

The gradient ∇_k of the mean square error function with $W = W_k$ is obtained by differentiating (9):

$$\nabla_k \triangleq \left\{\begin{matrix} \dfrac{\partial E[\varepsilon_j^2]}{\partial w_1} \\ \vdots \\ \dfrac{\partial E[\varepsilon_j^2]}{\partial w_n} \end{matrix}\right\}_{W=W_k} = -2P + 2RW_k. \quad (10)$$

The optimal weight vector W^*, generally called the Wiener weight vector, is obtained by setting the gradient to zero:

$$W^* = R^{-1}P. \quad (11)$$

This equation is a matrix form of the Wiener–Hopf equation [12]–[14].

For the purposes of subsequent analysis it is convenient to reexpress the mean square error function (9) and the gradient function (10) in more compact form. Substituting (11) in (9) yields the minimum mean square error:

$$\xi_{\min} = E[d_j^2] - W^{*T}P. \quad (12)$$

Recombining (12) with (9) and (11) yields

$$\xi_k = \xi_{\min} + V_k^TRV_k \quad (13)$$

where

$$V_k \triangleq W_k - W^*. \quad (14)$$

The gradient may be expressed in terms of V_k as

$$\nabla_k = 2RV_k. \quad (15)$$

If one assumes that the R-matrix is positive definite, it may be expressed in normal form as follows

$$R = Q\Lambda Q^{-1} \quad (16)$$

where the columns of the square modal matrix Q are the eigenvectors of R and Λ is the diagonal matrix of eigenvalues. If Q is constructed to be orthonormal,[2] then one may write

$$Q^{-1} = Q^T. \quad (17)$$

Note further that the inverse of R is

$$R^{-1} = Q\Lambda^{-1}Q^{-1}. \quad (18)$$

The mean square error may thus be expressed as

$$\xi_k = \xi_{\min} + V_k^TQ\Lambda Q^TV_k. \quad (19)$$

A new set of coordinates may now be defined as follows:

$$V' = Q^TV = Q^{-1}V \quad (20)$$

and

$$V'^T = V^TQ. \quad (21)$$

Substituting (20) and (21) into (19) then yields

$$\xi_k = \xi_{\min} + V_k'^T\Lambda V_k'. \quad (22)$$

The transformation Q projects V into V'—that is, projects V into primed coordinates. It can be observed from (22) that, since Λ is diagonal, the primed coordinates must comprise the principal axes of the quadratic mean square error performance surface. The gradient expressed in primed coordinates then becomes

$$\nabla_k' = 2\Lambda V_k'. \quad (23)$$

III. The Method of Steepest Descent

The practical objective of the adaptive process is to find a solution to (11). One way of doing so would be by analytical means. An analytical solution, however, would present serious computational difficulties when the number of weights was large or when the input data rate was high. In addition to the inversion of an $n \times n$ matrix, it could require as many as $n(n + 3)/2$ autocorrelation and cross correlation measurements to obtain the elements of R and P. Furthermore, this process would have to be continually repeated in most circumstances, where the input

[2] This can always be done when R is positive definite.

signal statistics would be slowly varying. For these reasons it is more practicable to make use of other recursive statistical estimation methods in devising algorithms for use in adaptive filters.

A well known and proven method for adjusting the response of an adaptive system is that of steepest descent [15], [16]. Adaptation by this method starts with an arbitrary initial value W_0 for the weight vector. The gradient of the mean square error function is measured and the weight vector altered in accordance with the negative of the value obtained. This procedure is repeated, causing the error to be successively reduced and the weight vector to approach the optimal value.

The method of steepest descent can be described by the relation

$$W_{k+1} = W_k + \mu(-\nabla_k) \tag{24}$$

where μ is a parameter that controls stability and rate of convergence, and ∇_k is the value of the gradient at a point on the error surface corresponding to $W = W_k$. An expression for the gradient, a linear function of the weights, is given by (15). Substituting this expression into (24) yields

$$W_{k+1} = W_k - 2\mu R V_k. \tag{25}$$

Subtracting W^* from both sides of (25) yields

$$V_{k+1} = V_k - 2\mu R V_k = (I - 2\mu R)V_k. \tag{26}$$

Equation (26) is a linear homogeneous vector difference equation whose solution characterizes the dynamic behavior of the weight vector as it begins at W_0 and, if the process is convergent, relaxes toward W^*. The solution of (26) is given by

$$V_k = (I - 2\mu R)^k V_0. \tag{27}$$

This solution is stable (convergent) if

$$\lim_{k\to\infty} (I - 2\mu R)^k = 0. \tag{28}$$

Since

$$(I - 2\mu R) = Q(I - 2\mu\Lambda)Q^{-1} \tag{29}$$

and

$$(I - 2\mu R)^k = Q(I - 2\mu\Lambda)^k Q^{-1} \tag{30}$$

condition (28) will be satisfied if

$$\lim_{k\to\infty} (I - 2\mu\Lambda)^k = 0. \tag{31}$$

Condition (31) will be met when

$$|1 - 2\mu\lambda_p| < 1 \tag{32}$$

for $p = 1,2,\cdots,n$. Since all eigenvalues are positive,

$$\frac{1}{\lambda_{\max}} > \mu > 0 \tag{33}$$

where $\lambda_{\max}$ is the largest eigenvalue of R. Equation (33) gives the stable range for μ.

It is easily shown that in primed coordinates the method of steepest descent is represented by

$$V'_{k+1} = (I - 2\mu\Lambda)V_k' \tag{34}$$

whose solution is

$$V_k' = (I - 2\mu\Lambda)^k V_0'. \tag{35}$$

For the pth coordinate one may write

$$v_{pk}' = (1 - 2\mu\lambda_p)^k v_{p0}'. \tag{36}$$

Equation (36) represents a simple geometric progression for v_{pk}', starting from the initial condition v_{p0}'. The pth geometric ratio is

$$r_p = (1 - 2\mu\lambda_p). \tag{37}$$

An exponential envelope of time constant τ_p can be fitted to the geometric sequence represented by (36). If the unit of time is one iteration cycle, then

$$r_p = \exp(-1/\tau_p) = 1 - \frac{1}{\tau_p} + \frac{1}{2!\,\tau_p^2} - \cdots. \tag{38}$$

In practical adaptive processes μ is chosen so that τ_p is large compared to one; the series of (38) can thus be represented by its first two terms. Combining (38) with (37) gives a formula for the pth time constant of the method of steepest descent:

$$\tau_p = \frac{1}{2\mu\lambda_p}. \tag{39}$$

Transient phenomena in the weights, as seen from (35) and (36), are simple geometric sequences along the primed coordinates. Along the original unprimed coordinates, the same phenomena, represented by (27), are more complicated. Transients in the weights themselves thus consist of sums of geometric sequences, the number of time constants typically being equal to the number of weights.

While transients are occurring in the weights as they relax toward the optimal Wiener solution, the mean square error undergoes changes. The expected error, for $W = W_k$, is given by (22). The weight transients, expressed in terms of V_k', are given by (35). A "learning curve" showing mean square error as a function of number of iterations k can be computed by substituting (35) into (22):

$$\xi_k = \xi_{\min} + V_0'^T(I - 2\mu\Lambda)^{2k}\Lambda V_0'. \tag{40}$$

As long as conditions (31) and (33) are met, the adaptive process will converge on the minimum point of the mean square error surface:

$$\lim_{k\to\infty} \xi_k = \xi_{\min}. \tag{41}$$

The mean square error solution starts at $k = 0$ with an initial value $\xi_{\min} + V_0^T\Lambda V_0'$, corresponding to $V_k' = V_0'$, and relaxes toward $\xi_{\min}$. The relaxation process is a sum of geometric sequences whose pth mode has a geometric ratio of $(1 - 2\mu\lambda_p)^2$. Thus the mean square error learning curve has a pth mode time constant of

$$\tau_{p\,\mathrm{mse}} = \frac{1}{4\mu\lambda_p} = \frac{\tau_p}{2}. \tag{42}$$

Learning curves of computer simulated adaptive processes will be presented below.

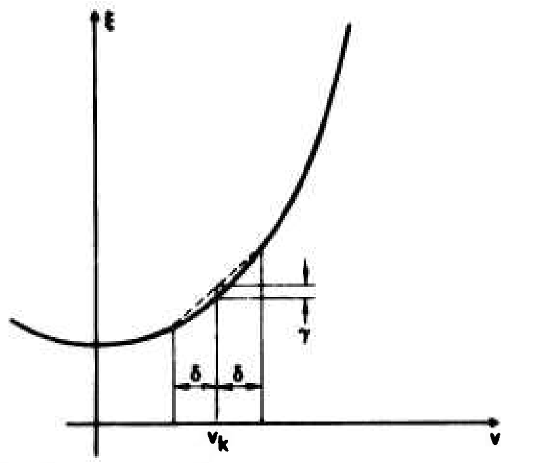


Fig. 2. Gradient estimation by derivative measurement.

If exact gradient measurements could be made each iteration, the adaptive weight vector would converge to the Wiener optimal weight vector. In reality, however, exact gradient measurements are not possible, and the gradient vector must be estimated from a limited statistical sample. The following sections describe two algorithms based on the method of steepest descent that use different techniques to obtain the necessary gradient estimates. The first uses differentiation and requires that finite perturbations be made in the weight vector. The second, the LMS algorithm, obtains gradient estimates directly and without perturbing or "dithering" the nominal weight vector adjustment.

A. Differential Algorithm

One way of estimating gradient vectors is by the direct measurement of derivatives. Although this technique is straightforward and easy to implement, it has been largely overlooked in the literature and is here analyzed in detail. For convenience the resulting algorithm is designated the DSD ("differential steepest descent") algorithm.

1) *Gradient estimation by derivative measurement:* A single component of the gradient vector can be measured in the manner illustrated in Fig. 2. The curve representing the parabolic mean square error function of a single variable is defined by

$$\xi(v_k) \triangleq \xi_k = \lambda v_k^2 + \xi_{\min}. \tag{43}$$

Its first and second derivatives are

$$\left(\frac{d\xi}{dv}\right)_{v=v_k} = 2\lambda v_k \tag{44}$$

$$\left(\frac{d^2\xi}{dv^2}\right)_{v=v_k} = 2\lambda. \tag{45}$$

The derivatives are numerically estimated by taking "symmetric differences":

$$\left(\frac{d\xi}{dv}\right)_{v=v_k} = \frac{\xi(v_k+\delta) - \xi(v_k-\delta)}{2\delta} \tag{46}$$

$$\left(\frac{d^2\xi}{dv^2}\right)_{v=v_k} = \frac{\xi(v_k+\delta) - 2\xi(v_k) + \xi(v_k-\delta)}{\delta^2}. \tag{47}$$

These finite differences are exact for the quadratic ξ-function.

The procedure illustrated in Fig. 2 requires that the weight adjustment be altered while the gradient measurement is being made. It is assumed that no time is spent at the nominal adjustment v_k but that equal time[3] is spent at $v_k + \delta$ and $v_k - \delta$. The result is that on the average the mean square error is greater by an amount γ than it would have been if the adjustment had remained at v_k. A performance penalty thus results from the weight vector alteration.

The quantity γ can be calculated for the one-dimensional quadratic ξ-function as follows:

$$\gamma = \frac{\lambda(v_k+\delta)^2 + \lambda(v_k-\delta)^2 + 2\xi_{\min}}{2} - \lambda(v_k)^2 - \xi_{\min}$$

$$= \lambda\delta^2. \tag{48}$$

Notice that the value of γ depends only on λ and δ and not on v_k. A dimensionless measure of how much the adaptive system is perturbed each time the gradient is measured, a parameter that may be called the "perturbation" P, is defined as follows:

$$P \triangleq \frac{\gamma}{\xi_{\min}} = \frac{\lambda\delta^2}{\xi_{\min}}. \tag{49}$$

This is the average increase in mean square error normalized with respect to the minimum achievable mean square error.

The estimation of two-dimensional gradients may now be considered. In this case the $\boldsymbol{R}$-matrix is given by

$$\boldsymbol{R} = \begin{bmatrix} r_{00} & r_{01} \\ r_{10} & r_{11} \end{bmatrix} \tag{50}$$

and the ξ-function is

$$\xi = r_{00}v_1^2 + r_{11}v_2^2 + 2r_{01}v_1v_2 + \xi_{\min}. \tag{51}$$

When the partial derivative of the error surface along coordinate v_1 is measured, the perturbation is

$$P = r_{00}\delta^2/\xi_{\min}. \tag{52}$$

The perturbation for measurement along coordinate v_2 is

$$P = r_{11}\delta^2/\xi_{\min}. \tag{53}$$

Assuming that equal time is required for the measurement of each gradient component (that is, that $2N$ data samples are used for each measurement), the average perturbation during the measurement is given by

$$P_{av} = \frac{\delta^2}{\xi_{\min}}\,\frac{r_{00} + r_{11}}{2}. \tag{54}$$

If one now defines a general perturbation for n dimensions as the average of the perturbations of the individual gradient component measurements, one obtains

$$P = \frac{\delta^2}{\xi_{\min}}\,\frac{\operatorname{tr}\boldsymbol{R}}{n}. \tag{55}$$

[3] The time required to take N data samples.

Since the trace of the R-matrix is equal to the sum of its eigenvalues, and since the sum divided by the number of eigenvalues is equal to the average of the eigenvalues, the perturbation may be conveniently expressed as

$$P = \frac{\delta^2 \lambda_{av}}{\xi_{min}}. \qquad (56)$$

Other means of gradient measurement have been used in practical systems. A weight can be perturbed or dithered sinusoidally, and the cross correlation between the weight value and the value of the performance function determined. All weights can be simultaneously dithered at individual frequencies and the gradient components obtained by cross correlation. The procedure of Fig. 2 corresponds to square-wave dithering.

2) *Gradient measurement noise:* Gradients measured in the manner shown in Fig. 2 are noisy because they are based on differences in ξ-measurements that are noisy. Each ξ-measurement is an estimate based on N error samples:

$$\xi = \frac{1}{N} \sum_{j=1}^{N} \varepsilon_j^2. \qquad (57)$$

It is well known that the variance in an estimate of the mean square obtained from N independent samples is equal to the difference divided by N between the mean fourth and the square of the mean square. The variance in the estimate of ξ may accordingly be expressed as

$$\operatorname{var}[\xi] = \frac{E[\varepsilon_j^4] - (E[\varepsilon_j^2])^2}{N}. \qquad (58)$$

If ε_j is normally distributed with zero mean and variance of σ^2, its mean fourth is $3\sigma^4$, and the square of its mean square is σ^4. The variance in the estimate of ξ is thus

$$\operatorname{var}[\xi] = \frac{1}{N}(3\sigma^4 - \sigma^4) = \frac{2\sigma^4}{N} = \frac{2\xi^2}{N}. \qquad (59)$$

Note that the variance is proportional to the square of ξ and inversely proportional to the number of data samples. It can thus in general be expressed as

$$\operatorname{var}[\xi] = \frac{\kappa \xi^2}{N} \qquad (60)$$

where κ has a value of 2 for an unbiased Gaussian probability density. If the probability density is other than Gaussian, the value of κ is generally less than but close to two. It is thus assumed for the purposes of subsequent analysis that

$$\operatorname{var}[\xi] = \frac{2\xi^2}{N}. \qquad (61)$$

The derivatives required by the gradient estimation technique of Fig. 2 are measured in accordance with (46). The error in the derivative estimate will be a sum of two components that, since the samples of the error ε_j are assumed to be independent, will also be independent. The variance of each component is determined by (61). If it is assumed that the perturbation P is small, that the adaptive process is close to convergence, and that the weight vector remains near the minimum point of the mean square error surface, then the two components will have essentially the same variances, and these variances will be additive. The variance in the estimate of the derivative, using (46) and (61), may be expressed as

$$\operatorname{var}\left[\frac{d\xi}{dv}\right]_{v=v_k} = \frac{1}{4\delta^2}\left[\frac{2\xi^2(v_k+\delta)}{N} + \frac{2\xi^2(v_k-\delta)}{N}\right] \cong \frac{\xi_{min}^2}{N\delta^2}. \qquad (62)$$

When a gradient vector is measured, the errors in each component are independent. The gradient noise vector N_k may thus be defined in terms of the true gradient ∇_k and the estimated gradient $\hat{\nabla}_k$:

$$\hat{\nabla}_k \triangleq \nabla_k + N_k. \qquad (63)$$

Under the assumed conditions the covariance of the gradient noise vector is thus given by

$$\operatorname{cov}[N_k] = \frac{\xi_{min}^2}{N\delta^2} I. \qquad (64)$$

It is also useful to obtain an expression for the covariance of the gradient noise vector in primed coordinates:

$$N_k' = Q^{-1} N_k. \qquad (65)$$

Since the covariance matrix of $\mathcal{N}_k$ is scalar, projecting into primed coordinates through the orthonormal transformation Q^{-1} yields the same covariance for N_k':

$$\operatorname{cov}[N_k'] = E[Q^{-1} N_k N_k^T Q] = \frac{\xi_{min}^2}{N\delta^2} I. \qquad (66)$$

Near the minimum point of the mean square error surface the covariance of the gradient noise is essentially constant and not a function of W_k.

3) *Noise in the weight vector:* Adaptation based on noisy gradient estimates results in noise in the weight vector. The method of steepest descent with ideal gradients is represented by (26). With estimated gradients this equation may be rewritten as

$$V_{k+1} = V_k + \mu(-\hat{\nabla}_k) = V_k + \mu(-\nabla_k - N_k). \qquad (67)$$

Substituting (15) and combining terms yields

$$V_{k+1} = (I - 2\mu R)V_k - \mu N_k \qquad (68)$$

a first-order vector difference equation with a stochastic driving function of $-\mu N_k$. Projection into primed coordinates may be accomplished by premultiplying both sides of (68) by Q^{-1}:

$$V_{k+1}' = (I - 2\mu\Lambda)V_k' - \mu N_k'. \qquad (69)$$

In steady state, after initial adaptive transients have died out, V_k' undergoes a stationary random process in response to the stationary driving function $-\mu N_k'$. Since there is no cross coupling between terms and the components of N_k' are mutually uncorrelated, the components of V_k'

will also be mutually uncorrelated, and the covariance matrix of N_k' will be diagonal. To find this matrix one first multiplies both sides of (69) by their own transposes:

$$\begin{aligned} V'_{k+1}V'^T_{k+1} &= (I - 2\mu\Lambda)V_k'V_k'^T(I - 2\mu\Lambda) \\ &\quad + \mu^2 N_k'N_k'^T - \mu(I - 2\mu\Lambda)V_k'N_k'^T \\ &\quad - \mu N_k'V_k'^T(I - 2\mu\Lambda). \end{aligned} \tag{70}$$

Taking expected values of both sides yields[4]

$$\begin{aligned} \text{cov}\,[V_k'] &= (I - 2\mu\Lambda)\,\text{cov}\,[V_k'](I - 2\mu\Lambda) \\ &\quad + \mu^2\,\text{cov}\,[N_k']. \end{aligned} \tag{71}$$

Combining terms further yields

$$\text{cov}\,[V_k'] = \mu^2(4\mu\Lambda - 4\mu^2\Lambda^2)^{-1}\,\text{cov}\,[N_k']. \tag{72}$$

In practical circumstances the method of steepest descent is implemented with a small value of μ, so that

$$\mu\Lambda \ll I. \tag{73}$$

Neglecting the squared terms in (72) thus yields

$$\text{cov}\,[V_k'] = \frac{\mu}{4}\Lambda^{-1}\,\text{cov}\,[N_k']. \tag{74}$$

Using (66) one may now write

$$\text{cov}\,[V_k'] = \frac{\mu\xi^2_{\min}}{4N\delta^2}\Lambda^{-1}. \tag{75}$$

The components of V_k' are mutually uncorrelated but not all of the same variance. The covariance of V_k can be obtained from (75) by using (18) and (20):

$$\text{cov}\,[V_k] = E[QV_k'V_k'^TQ^{-1}] = \frac{\mu\xi^2_{\min}}{4N\delta^2}R^{-1}. \tag{76}$$

4) *Misadjustment:* Without noise in the weight vector, adaptation by the method of steepest descent would converge to a steady-state solution at the minimum point of the mean square error surface. The mean square error would therefore be $\xi_{\min}$. Noise in the weight vector, however, tends to cause the steady-state solution to vary randomly about the minimum point—that is, to "climb the sides of the bowl." The result is an "excess" mean square error, a mean square error that is greater than $\xi_{\min}$.

An expression for mean square error in terms of V' is given by (22), where the excess mean square error is $V_k'^T\Lambda V_k'$. The average excess mean square error is

$$E[V_k'^T\Lambda V_k'] = \sum_{p=1}^{n}\lambda_p E[(v_{pk}')^2]. \tag{77}$$

From (75) one may write

$$E[(v_{pk}')^2] = \frac{\mu\xi^2_{\min}}{4N\delta^2}\left(\frac{1}{\lambda_p}\right). \tag{78}$$

[4] Note that since V_k' is affected only by gradient noise from previous adaptive cycles, V_k' and N_k' are uncorrelated.

Thus (77) can be rewritten

$$E[V_k'^T\Lambda V_k'] = \frac{n\mu\xi^2_{\min}}{4N\delta^2}. \tag{79}$$

A useful parameter in the design of adaptive processes is the misadjustment M, which is defined as the average excess mean square error divided by the minimum mean square error:

$$M \triangleq \frac{E[V_k'^T\Lambda V_k']}{\xi_{\min}}. \tag{80}$$

The misadjustment is a dimensionless measure of the difference between adaptive performance and optimal Wiener performance as a result of gradient noise. In other words, it is a measure of the cost of adaptability.

Using (79) one can express the misadjustment for the DSD algorithm as follows:

$$M = \frac{n\mu\xi_{\min}}{4N\delta^2}. \tag{81}$$

This formula is simple and clear but can be more usefully expressed in terms of time constants of the learning process and the perturbation of the gradient estimation process.

Each gradient component measurement uses $2N$ samples of data. Each iteration involves n gradient component measurements and therefore requires $2Nn$ data samples. The time constant $\tau_{p_{\text{mse}}}$ is given by (42) in number of iterations, a basic "unit of time." If one now defines a new time constant $T_{p_{\text{mse}}}$ whose basic unit is the data sample and whose value is expressed in number of data samples, then for the DSD algorithm

$$T_{p_{\text{mse}}} \triangleq 2nN\tau_{p_{\text{mse}}}. \tag{82}$$

The new time constant is easily related to real time if the sampling rate is known.

Using the perturbation formula (56) one can reexpress the misadjustment for the DSD algorithm (81) as

$$M = \frac{n\mu\lambda_{\text{av}}}{4NP}. \tag{83}$$

Using (42) the time constant defined by (82) can also be reexpressed as

$$T_{p_{\text{mse}}} = \frac{nN}{2\mu\lambda_p} \tag{84}$$

which is equivalent to

$$\lambda_p = \frac{nN}{2\mu}\left(\frac{1}{T_{p_{\text{mse}}}}\right) \tag{85}$$

or

$$\lambda_{\text{av}} = \frac{nN}{2\mu}\left(\frac{1}{T_{\text{mse}}}\right)_{\text{av}}. \tag{86}$$

Combining (86) with (83) shows the misadjustment to be

$$M = \frac{n^2}{8P}\left(\frac{1}{T_{\text{mse}}}\right)_{\text{av}}. \tag{87}$$

For the DSD algorithm, misadjustment is thus proportional to the square of the number of weights and inversely proportional to the perturbation. It is also inversely proportional to the speed of adaptation; that is, fast adaptation results in a high misadjustment. More specifically, the misadjustment is dependent on the average reciprocal time constant of the learning curve whose time base is calibrated in number of data samples. Note that very fast modes may dominate this average and cause an increase in misadjustment, while the rate of convergence will remain limited by the slowest mode. In other words, with disparate eigenvalues in the $\boldsymbol{R}$-matrix, the adaptive process may be afflicted with the misadjustment of its fastest modes but may converge only at the rate of its slowest modes. With equal or closely similar eigenvalues, the process is more efficient, and the misadjustment is given by

$$M = \frac{n^2}{8PT_{\text{mse}}}. \tag{88}$$

In this case the learning curve has only one time constant, T_{mse}.

Misadjustment as defined here is a normalized performance penalty resulting from noise in the weight vector and is a stochastic effect. In an actual adaptive system, where the weight vector is deterministically perturbed to measure the gradient, another penalty accrues, the perturbation, also a ratio of excess mean square error to minimum mean square error. The total excess mean square error can be shown to be the sum of the "stochastic" and "deterministic" components. The total misadjustment is thus

$$M_{\text{tot}} \triangleq M + P. \tag{89}$$

Adding these components yields

$$M_{\text{tot}} = \frac{n^2}{8P}\left(\frac{1}{T_{\text{mse}}}\right)_{\text{av}} + P. \tag{90}$$

The perturbation is a design parameter. Its choice is optimized by differentiating (90) with respect to P and setting the derivative to zero. The result is to make the two right-hand terms of (90) equal. The optimal perturbation is thus

$$P_{\text{opt}} = \tfrac{1}{2}M_{\text{tot}} \tag{91}$$

and the minimum total misadjustment is

$$(M_{\text{tot}})_{\text{min}} = \frac{n^2}{4P_{\text{opt}}}\left(\frac{1}{T_{\text{mse}}}\right)_{\text{av}} = \left[\frac{n^2}{2}\left(\frac{1}{T_{\text{mse}}}\right)_{\text{av}}\right]^{1/2}. \tag{92}$$

The use of the above misadjustment formulas in the design of adaptive systems will be illustrated in Section V below.

B. LMS Algorithm

The LMS algorithm is an implementation of the method of steepest descent that employs a gradient estimation technique more efficient than derivative measurement. This algorithm, however, is not universally applicable, and its use is restricted to the adaptive linear combiner of Fig. 1, where inputs X_j and d_j are given.

1) *Gradient estimation, convergence, time constants:* The error ε_j of the adaptive linear combiner of Fig. 1 is given by (4). A gradient estimate may be obtained by squaring the single value of ε_j and differentiating it as if it were the mean square error:

$$\hat{\nabla}_j = \begin{Bmatrix} \dfrac{\partial \varepsilon_j^2}{\partial w_1} \\ \vdots \\ \dfrac{\partial \varepsilon_j^2}{\partial w_n} \end{Bmatrix} = 2\varepsilon_j \begin{Bmatrix} \dfrac{\partial \varepsilon_j}{\partial w_1} \\ \vdots \\ \dfrac{\partial \varepsilon_j}{\partial w_n} \end{Bmatrix} = -2\varepsilon_j X_j. \tag{93}$$

Substituting (93) into (24) yields the LMS algorithm:

$$W_{j+1} = W_j + 2\mu\varepsilon_j X_j. \tag{94}$$

Since a new gradient estimate is obtained with each data sample, an adaptive iteration is effected with the arrival of each sample. The index k is thus replaced with the index j.

The gradient estimate of (93) may be implemented in a practical system without further squaring, averaging, or differentiation and is elegant in its simplicity and efficiency. All components of the gradient vector are obtained from a single data sample without perturbation of the weight vector. Since the estimate is obtained without averaging, it contains a large component of noise. The noise, however, is averaged and attenuated by the adaptive process, which acts as a low-pass filter in this respect. It is important to note also that for a fixed value of W the estimate is unbiased:

$$E[\hat{\nabla}_j] = -2E[\varepsilon_j X_j] = -2E[d_j X_j - X_j X_j^T W]. \tag{95}$$

From (10), the formula for the true gradient, this expression can be rewritten as

$$E[\hat{\nabla}_j] = -2(P - RW) = \nabla. \tag{96}$$

Proofs of convergence of the LMS algorithm have appeared in the literature [4], [11], [17]–[20].[5] These proofs show that the algorithm is stable when

$$1/\lambda_{\max} > \mu > 0 \tag{97}$$

which is the same as the condition for stability of the method of steepest descent in general, given by (33). It is also shown in [4] and [19] that the time constants of the LMS algorithm are

$$\tau_{p_{\text{mse}}} = \tfrac{1}{2}\tau_p = \frac{1}{4\mu\lambda_p} \tag{98}$$

which are similarly identical to the time constants for the method of steepest descent, given by (42). Once again, τ_p is the time constant of the pth mode for transient phenomena in the weights, while $\tau_{p_{\text{mse}}}$ is the corresponding time constant of the learning curve. Since only one data sample per itera-

[5] For input vectors X_j mutually uncorrelated over time; proofs for correlated input vectors have been developed in [21] and [22].

tion is used, the time constant expressed in number of data samples is

$$T_{p_{\text{mse}}} = \tau_{p_{\text{mse}}}. \tag{99}$$

2) *Gradient measurement noise:* Let it be assumed that the adaptive process, using a small value of the adaptive constant μ, has converged to a steady state near the minimum point of the mean square error surface defined by (9). The gradient estimation noise of the LMS algorithm at the minimum point, where the true gradient is zero, is the gradient estimate itself:

$$N_j = \hat{\nabla}_j = -2\varepsilon_j X_j. \tag{100}$$

The covariance of this noise is given by

$$\text{cov}\,[N_j] = E[N_j N_j^T] = 4E[\varepsilon_j^2 X_j X_j^T]. \tag{101}$$

It is well known from Wiener filter theory that, when the weight vector is optimized (that is, when $W_j = W^*$), the error ε_j is uncorrelated with the input vector X_j. If one assumes that ε_j and X_j are Gaussian, not only are they uncorrelated at the minimum point of the error surface but also statistically independent. Under these conditions (101) becomes

$$\text{cov}\,[N_j] = 4E[\varepsilon_j^2]E[X_j X_j^T] = 4\xi_{\min} R. \tag{102}$$

In primed coordinates the covariance is

$$\text{cov}\,[N_j'] = Q^{-1}\,\text{cov}\,[N_j]Q = 4\xi_{\min}\Lambda. \tag{103}$$

3) *Noise in the weight vector:* Equations (67)–(74) above apply to the method of steepest descent with any means of gradient estimation that results in a diagonal covariance matrix for N_j'—that is, to both the DSD algorithm and the LMS algorithm. For the LMS algorithm, using (74) and (103), one may write

$$\text{cov}\,[V_j'] = \frac{\mu}{4}\Lambda^{-1}(4\xi_{\min}\Lambda) = \mu\xi_{\min} I. \tag{104}$$

The covariance of the steady state noise in the weight vector (at or near the minimum point of the mean square error surface) is

$$\text{cov}\,[V_j] = \mu\xi_{\min} I. \tag{105}$$

4) *Misadjustment:* For the LMS algorithm the misadjustment M, defined by (80), may be found as follows. The average excess mean square error, given by (77), may be written as

$$\begin{aligned} E[V_j'^T \Lambda V_j'] &= \sum_{p=1}^{n} \lambda_p E[(v_{pj}')^2] = \mu\xi_{\min} \sum_{p=1}^{n} \lambda_p \\ &= \mu\xi_{\min}\,\text{tr}\,R \end{aligned} \tag{106}$$

where, according to (104), $E[(v_{pj}')^2] = \mu\xi_{\min}$ for all p. The misadjustment is thus given by

$$M = \frac{E[V_j'^T \Lambda V_j']}{\xi_{\min}} = \mu\,\text{tr}\,R. \tag{107}$$

This useful formula may be reexpressed in a manner that allows one to perceive the relationship between misadjustment and rate of adaptation. According to (98) one may write

$$\mu\lambda_p = \frac{1}{4\tau_{p_{\text{mse}}}} \tag{108}$$

and

$$\mu\,\text{tr}\,R = \frac{1}{4}\sum_{p=1}^{n}\frac{1}{\tau_{p_{\text{mse}}}} = \frac{n}{4}\left(\frac{1}{\tau_{p_{\text{mse}}}}\right)_{\text{av}}. \tag{109}$$

The misadjustment may thus be written

$$M = \frac{n}{4}\left(\frac{1}{\tau_{p_{\text{mse}}}}\right)_{\text{av}}. \tag{110}$$

It is interesting to compare (110) with (87), the misadjustment formula for the DSD algorithm. Once again it is apparent that misadjustment is reduced by slow adaptation, by making the values of $\tau_{p_{\text{mse}}}$, where $p = 1,\cdots,n$, large. With the LMS algorithm, however, for a given value of misadjustment, the adaptive time constants increase linearly with the number of weights rather than with the square of the number of weights. Furthermore, there is no perturbation. In typical circumstances much faster adaptation is thus possible than with the DSD algorithm, as will be borne out by the numerical examples presented in Section VI.

It may also be observed from (110) that the LMS algorithm, since it is based on the method of steepest descent, suffers like the DSD algorithm when there is a great disparity in the eigenvalues of R. Under such conditions misadjustment once again can be dominated by the fastest modes (those with the smallest time constant $\tau_{p_{\text{mse}}}$), while rate of convergence can be limited by the slowest modes.

When the eigenvalues are equal, a useful formula for the misadjustment of the LMS algorithm is

$$M = \frac{n}{4}\left(\frac{1}{\tau_{\text{mse}}}\right). \tag{111}$$

Experience has shown this formula to be a good approximation of the relationship between misadjustment, time constant of the learning curve, and number of weights even when the eigenvalues are not equal. Such a relationship is needed in designing an adaptive system when the eigenvalues are unknown.

Since trace R is the total power of the inputs to the weights, which is generally known, one can use (107) in choosing a value of μ that will produce a desired value of M. One can accordingly combine (111) and (107) to obtain a general formula for time constant of the learning curve with equal eigenvalues:

$$\tau_{\text{mse}} = \frac{n}{4\mu\,\text{tr}\,R}. \tag{112}$$

This formula is also a good approximation in many cases when the eigenvalues of R are unequal.

IV. Random Search

The method of steepest descent is a systematic surface-searching procedure. Although randomness enters in practice through gradient estimation noise, adaptation by this method is basically a deterministic process. Random search, by contrast, seeks to improve performance by making random changes in system parameters. A simple algorithm based on this method, inspired by the Darwinian concept of evolution, may be called random search by "natural selection." Though derived from a natural model this algorithm appears to offer a practical approach to the adaptive process that may have engineering merit [23].

In random search by natural selection a random change is made in the weight vector of an adaptive processor, such as the linear combiner of Fig. 1. The mean square error is measured before and after the change and the measurements compared. If the change causes the error to be lower, it is accepted. If it does not, it is rejected, and a new random change is tried. This procedure can be described algebraically as follows:

$$W_{k+1} = W_k + \tfrac{1}{2}[1 + \operatorname{sgn}\{\hat{\xi}(W_k) - \hat{\xi}(W_k + U_k)\}]U_k \tag{113}$$

where U_k is a random vector; $\hat{\xi}(W_k)$ is an estimate of mean square error based on N samples of ε_j with $W = W_k$; $\hat{\xi}(W_k + U_k)$ is an estimate of mean square error based on N samples of ε_j with $W = W_k + U_k$; and sgn $\{z\}$ is $+1$ for $z \geq 0$ and -1 for $z < 0$.

This algorithm, though easy to implement, has the drawback that nothing is learned when a trial change is rejected and forgotten. For this reason a more efficient "linear" random search algorithm, hereafter called the "LRS algorithm," has been devised. In this algorithm, first described here, a small random change U_k is tentatively added to the weight vector at the beginning of each iteration. The corresponding change in mean square error performance is observed. A permanent weight vector change, proportional to the product of the change in performance and the initial tentative change, is then made. This procedure can be expressed algebraically as follows:

$$W_{k+1} = W_k + \beta[\hat{\xi}(W_k) - \hat{\xi}(W_k + U_k)]U_k \tag{114}$$

where U_k is a random vector from a random vector generator designed to have a covariance of $\sigma^2 I$; $\hat{\xi}(W_k)$ and $\hat{\xi}(W_k + U_k)$ are defined as in (113); and the terms β and σ^2 are design constants affecting stability and rate of adaptation.

The LRS algorithm is "linear" because the weight change is proportional to the change in mean square error, and in this respect it differs from random search by natural selection as described in (113). The latter algorithm is simpler to implement but does not perform as well. It is also difficult to treat mathematically, and a performance analysis is not attempted in this paper.

For the purpose of analyzing the LRS algorithm, the following definitions are useful. The true change in mean square error resulting from the addition of U_k to W_k is given by

$$(\Delta\xi)_k \triangleq \xi(W_k + U_k) - \xi(W_k). \tag{115}$$

The corresponding estimated change in mean square error is

$$(\widehat{\Delta\xi})_k \triangleq \hat{\xi}(W_k + U_k) - \hat{\xi}(W_k). \tag{116}$$

The error in the estimated change is

$$\zeta_k \triangleq (\Delta\xi)_k - (\widehat{\Delta\xi})_k \tag{117}$$

whose variance, from (59), is given by

$$\begin{aligned}\operatorname{var}[\zeta_k] &= \operatorname{var}[(\widehat{\Delta\xi})_k]\\ &= \operatorname{var}[\hat{\xi}(W_k + U_k)] + \operatorname{var}[\hat{\xi}(W_k)]\\ &= \frac{2}{N}[\xi^2(W_k + U_k) + \xi^2(W_k)].\end{aligned} \tag{118}$$

In steady state operation near the minimum point of the mean square error surface, (118) can be expressed as

$$\operatorname{var}[\zeta_k] \cong \frac{4}{N}\xi_{\min}^2. \tag{119}$$

A perturbation is caused by the tentative changes in the weight vector that are a part of the LRS algorithm. At each iteration, N samples of data are used to obtain $\hat{\xi}(W_k)$, with the weight vector set at its nominal value, and N samples to obtain $\hat{\xi}(W_k + U_k)$. The next nominal value is chosen immediately after the two $\hat{\xi}$ measurements are made. During a given cycle the average excess mean square error is thus given by

$$E\left[\xi(W_k) - \frac{\xi(W_k) + \xi(W_k + U_k)}{2}\right] = \tfrac{1}{2}E[\xi(W_k) - \xi(W_k + U_k)]. \tag{120}$$

Since U_k has zero mean and is uncorrelated with W_k, and since $\operatorname{cov}[U_k] = \operatorname{cov}[U_k'] = \sigma^2 I$, the average excess mean square error can also be expressed as

$$\tfrac{1}{2}E[U_k^T R U_k] = \tfrac{1}{2}E[U_k'^T \Lambda U_k'] = \tfrac{1}{2}\sigma^2 \operatorname{tr} R. \tag{121}$$

The perturbation P is defined as the ratio of the average excess mean square error (resulting from tentative changes in the weight vector) to the minimum mean square error. It may thus be expressed as

$$P = \frac{\sigma^2 \operatorname{tr} R}{2\xi_{\min}}. \tag{122}$$

1) *Stability, time constants of LRS algorithm:* Equation (114) may be rewritten, using (115), (116), and (117), as follows:

$$W_{k+1} = W_k + \beta[-(\Delta\xi)_k + \zeta_k]U_k \tag{123}$$

or

$$V_{k+1} = V_k + \beta[-(\Delta\xi)_k + \zeta_k]U_k. \tag{124}$$

If one lets σ^2 be small by design, so that U_k is always small, one can write

$$(\Delta\xi)_k = U_k^T \nabla_k = 2U_k^T R V_k. \tag{125}$$

Substituting (125) into (124) then yields

$$\begin{aligned}V_{k+1} &= V_k + \beta U_k[-2U_k^T R V_k + \zeta_k]\\ &= (I - 2\beta U_k U_k^T R)V_k + \beta\zeta_k U_k.\end{aligned} \tag{126}$$

Equations (114) and (126) are equivalent representations of the LRS algorithm, the former more useful for implementation and the latter for analysis. Equation (126) shows that the weight vector is the solution of a first-order linear vector difference equation having a randomly time-variable coefficient $-2\beta U_k U_k^T R$ and a random driving function $\beta \zeta_k U_k$.

Both sides of (126) may be premultiplied by Q^{-1} to obtain an equivalent expression in primed coordinates:

$$V'_{k+1} = (I - 2\beta U_k' U_k'^T \Lambda) V_k' + \beta \zeta_k U_k'. \qquad (127)$$

Though this expression is simpler than (126), it remains difficult to solve because of cross coupling and randomness in the matrix coefficient. It is thus necessary to derive stability conditions for the LRS algorithm without an explicit solution to (127). One may begin by studying the behavior of the mean of the weight vector.

By taking expected values of both sides of (127) and observing that U_k' is a random vector uncorrelated with ζ_k and V_k, one obtains

$$\begin{aligned} E[V'_{k+1}] &= E[(I - 2\beta U_k' U_k'^T \Lambda) V_k'] + \beta E[\zeta_k U_k'] \\ &= (I - 2\beta E[U_k' U_k'^T] \Lambda) E[V_k'] + 0 \\ &= (I - 2\beta \sigma^2 \Lambda) E[V_k']. \end{aligned} \qquad (128)$$

This equation is analogous to (34) for the method of steepest descent. Its solution is

$$E[V_k'] = (I - 2\beta\sigma^2\Lambda)^k V_0'. \qquad (129)$$

Equation (129) gives, for an initial condition of $V_k' = V_0'$, the expected value of the weight vector's transient response. Stability of (128) assures convergence of the mean of V_k'. The stability condition is

$$1/\lambda_{\max} > \beta\sigma^2 > 0. \qquad (130)$$

When $\beta\sigma^2$ is so chosen, the following condition is fulfilled:

$$\lim_{k\to\infty} E[V_k'] = 0. \qquad (131)$$

By analogy with the method of steepest descent, whose transient behavior is characterized by (34) through (39), the time constant of the pth mode of the expected value of the weight vector is

$$\tau_p = \frac{1}{2\beta\sigma^2\lambda_p}. \qquad (132)$$

The time constant of the pth mode of the mean square error learning curve is half this value:

$$\tau_{p_{\mathrm{mse}}} = \frac{1}{4\beta\sigma^2\lambda_p}. \qquad (133)$$

2) *Noise in the weight vector of the LRS algorithm:* If one lets $\beta\sigma^2$ be chosen so that (130) is satisfied, then the mean of the weight vector will converge according to (131). Convergence of the mean, however, does not necessarily imply boundedness of the covariance of the weight vector. For the purpose of obtaining an expression for the noise in the weight vector, such boundedness is here assumed without proof. It is also assumed that the weight vector undergoes a stationary stochastic process after initial adaptive transients have died out.

The assumed steady state covariance of the weight vector may be calculated as follows. Multiplying both sides of (127) by their own transposes yields

$$\begin{aligned} V'_{k+1} V'^T_{k+1} &= (I - 2\beta U_k' U_k'^T \Lambda) V_k' V_k'^T (I - 2\beta\Lambda U_k' U_k'^T) \\ &\quad + \beta^2 \zeta_k^2 U_k' U_k'^T \\ &\quad + (I - 2\beta U_k' U_k'^T \Lambda) V_k' \beta \zeta_k U_k'^T \\ &\quad + \beta \zeta_k U_k' V_k'^T (I - 2\beta\Lambda U_k' U_k'^T). \end{aligned} \qquad (134)$$

Noting that ζ_k and U_k' are stationary processes of zero mean uncorrelated with each other and taking expected values of both sides of (134) yields

$$\begin{aligned} &E[V'_{k+1} V'^T_{k+1}] \\ &\quad = E[(I - 2\beta U_k' U_k'^T \Lambda) V_k' V_k'^T (I - 2\beta\Lambda U_k' U_k'^T)] \\ &\qquad + \beta^2 E[\zeta_k^2] E[U_k' U_k'^T] + 0 \\ &\quad = E[(I - 2\beta U_k' U_k'^T \Lambda) V_k' V_k'^T (I - 2\beta\Lambda U_k' U_k'^T)] \\ &\qquad + \beta^2 \frac{4}{N} \xi^2_{\min} \sigma^2 I. \end{aligned} \qquad (135)$$

Since in steady state V_k' is also a stationary process of zero mean uncorrelated with U_k', one may write

$$\begin{aligned} &E[V'_{k+1} V'^T_{k+1}] \\ &\quad = E[(I - 2\beta U_k' U_k'^T \Lambda) E[V_k' V_k'^T] (I - 2\beta\Lambda U_k' U_k'^T)] \\ &\qquad + \beta^2 \frac{4}{N} \xi^2_{\min} \sigma^2 I \end{aligned} \qquad (136)$$

and

$$\begin{aligned} &\mathrm{cov}\,[V_k'] \\ &\quad = E[(I - 2\beta U_k' U_k'^T \Lambda)\,\mathrm{cov}\,[V_k'] (I - 2\beta\Lambda U_k' U_k'^T)] \\ &\qquad + \beta^2 \frac{4}{N} \xi^2_{\min} \sigma^2 I \\ &\quad = \mathrm{cov}\,[V_k'] - 2\beta E[U_k' U_k'^T] \Lambda\,\mathrm{cov}\,[V_k'] \\ &\qquad - 2\beta\,\mathrm{cov}\,[V_k'] \Lambda E[U_k' U_k'^T] \\ &\qquad + 4\beta^2 E[U_k' U_k'^T \Lambda\,\mathrm{cov}\,[V_k'] \Lambda U_k' U_k'^T] \\ &\qquad + \beta^2 \frac{2}{N} \xi^2_{\min} \sigma^2 I \\ &\quad = \mathrm{cov}\,[V_k'] - 2\beta\sigma^2 \Lambda\,\mathrm{cov}\,[V_k'] - 2\beta\sigma^2\,\mathrm{cov}\,[V_k'] \Lambda \\ &\qquad + 4\beta^2 E[U_k' U_k'^T \Lambda\,\mathrm{cov}\,[V_k'] \Lambda U_k' U_k'^T] \\ &\qquad + \beta^2 \frac{4}{N} \xi^2_{\min} \sigma^2 I. \end{aligned} \qquad (137)$$

Solving (137) to find the covariance of V_k' is difficult because the matrices cannot be factored. After reexamining (130), however, one could argue heuristically that in steady state the covariance matrix should be diagonal. All components of the driving function of (127) are uncorrelated with each other and uncorrelated over time. The random coefficient $I - 2\beta U_k' U_k'^T \Lambda$ is furthermore diagonal on the average, though generally not for each value of k, and uncorrelated with V_k' and with itself over time. Though this argument does not constitute a proof that the covariance of V_k' is diagonal, it makes such an assumption plausible.

If the covariance matrix of V_k' is thus assumed to be diagonal, then with some rearranging of terms (137) becomes

$$4\beta\sigma^2\Lambda \operatorname{cov}[V_k'] - 4\beta^2 E[U_k'U_k'^T\Lambda \operatorname{cov}[V_k']\Lambda U_k'U_k'^T] = \beta^2 \frac{4}{N}\xi_{\min}^2\sigma^2 I. \quad (138)$$

For slow adaptation, the case of greatest interest, it may be noted that

$$\beta\sigma^2\Lambda \ll I \quad (139)$$

which is analogous to (75) for the method of steepest descent.[6] One may note further that

$$\beta^2 E[U_k'U_k'^T\Lambda \operatorname{cov}[V_k']\Lambda U_k'U_k'^T] \cong (\beta\sigma^2\Lambda)^2 \operatorname{cov}[V_k'] \quad (140)$$

and from (139) that

$$(\beta\sigma^2\Lambda)^2 \operatorname{cov}[V_k'] \ll \beta\sigma^2\Lambda \operatorname{cov}[V_k']. \quad (141)$$

The term $-4\beta^2 E[\ ]$ of (138) is thus small and can be neglected. Equa ion (138) accordingly becomes

$$\operatorname{cov}[V_k'] = \frac{\beta}{N}\xi_{\min}^2\Lambda^{-1}. \quad (142)$$

Though this expression has not been rigorously derived, experience has shown it to lead to misadjustment formulas that are generally accurate.

3) *Misadjustment of LRS algorithm:* The average excess mean square error due to noise in the weight vector is given by (77). Using (142) one may write for the LRS algorithm

$$E[V_k'^T\Lambda V_k'] = \sum_{p=1}^{n}\lambda_p\left(\frac{\beta}{N}\xi_{\min}^2\frac{1}{\lambda_p}\right) = \frac{n\beta}{N}\xi_{\min}^2. \quad (143)$$

According to the definition of (80) the misadjustment of the LRS algorithm is thus

$$M = \frac{n\beta}{N}\xi_{\min}. \quad (144)$$

This result can be usefully expressed, using (121), in terms of the perturbation of the LRS process:

$$M = \frac{n\beta\sigma^2 \operatorname{tr} R}{2NP} = \frac{n^2\beta\sigma^2\lambda_{\mathrm{av}}}{2NP}. \quad (145)$$

It can also be expressed in terms of time constants of the adaptive process. The time constant of the pth mode of the learning curve, expressed in number of iterations, is given by (132). Since $2N$ samples of data are used per iteration, this time constant expressed in number of data samples is

$$T_{p_{\mathrm{mse}}} \triangleq 2N\tau_{p_{\mathrm{mse}}} = \frac{N}{2\beta\sigma^2\lambda_p}. \quad (146)$$

Note the difference between (146) and the equivalent expression (82) for the DSD algorithm, reflecting the difference in utilization of data per adaptive cycle by the two algorithms.

According to (146) one may write

$$\lambda_p = \frac{N}{2\beta\sigma^2}\left(\frac{1}{T_{p_{\mathrm{mse}}}}\right) \quad (147)$$

and

$$\lambda_{\mathrm{av}} = \frac{N}{2\beta\sigma^2}\left(\frac{1}{T_{p_{\mathrm{mse}}}}\right)_{\mathrm{av}}. \quad (148)$$

Inserting (148) into (145) yields

$$M = \frac{n^2}{4P}\left(\frac{1}{T_{p_{\mathrm{mse}}}}\right)_{\mathrm{av}}. \quad (149)$$

This formula closely resembles its counterpart (87) for the DSD algorithm.

According to (89) the total misadjustment must include the effects of perturbation. One may thus write

$$M_{\mathrm{tot}} = \frac{n^2}{4P}\left(\frac{1}{T_{p_{\mathrm{mse}}}}\right)_{\mathrm{av}} + P. \quad (150)$$

Optimal choice of P requires that both right-hand terms of (150) be equal and that P, therefore, be one-half the total misadjustment (91). One may thus further write

$$(M_{\mathrm{tot}})_{\min} = \frac{n^2}{2P_{\mathrm{opt}}}\left(\frac{1}{T_{p_{\mathrm{mse}}}}\right)_{\mathrm{av}} = n\left[\left(\frac{1}{T_{p_{\mathrm{mse}}}}\right)_{\mathrm{av}}\right]^{1/2}. \quad (151)$$

This formula once again closely resembles its counterpart (92) for the DSD algorithm and is further indicative of the fact that many behavioral properties of the LRS algorithm resemble those of steepest descent algorithms despite the difference in search procedure.

Other random search algorithms applicable to adaptive control and pattern recognition systems have been described in the literature [24]–[31]. These algorithms are capable of taking advantage of performance measurements from previous iterations in determining current parameter changes and are useful in searching multimodal performance surfaces. They tend to be complicated in implementation and mathematical description, however, and have not been analyzed to determine their misadjustment as a function of rate of adaptation. It is conjectured in this regard that their behavior may be somewhat similar to that of the LRS algorithm and that their convergence close to optimal points is relatively slow in high dimensional spaces.

V. Summary of Analytical Results

In the foregoing sections analytical expressions have been derived that characterize the performance of the DSD and LMS algorithms, based on the method of steepest descent, and the LRS algorithm, based on a random search

[6] The role of μ in the method of steepest descent is the same as that of $\beta\sigma^2$ in the LRS algorithm. Independent control over β and σ^2 is necessary, however, because σ^2 controls in addition the perturbation.

TABLE I
PERFORMANCE CHARACTERISTICS OF ADAPTIVE ALGORITHMS

	DSD algorithm	LMS algorithm	LRS algorithm
Misadjustment, M	$\frac{\mu n}{4N\delta^2}\xi_{\min} = \frac{n^2}{8P}\left(\frac{1}{T_{p_{mse}}}\right)_{av}$	$\mu \operatorname{tr} \boldsymbol{R} = \frac{n}{4}\left(\frac{1}{T_{p_{mse}}}\right)_{av}$	$\frac{\mu\beta}{N}\xi_{\min} = \frac{n^2}{4P}\left(\frac{1}{T_{p_{mse}}}\right)_{av}$
Perturbation, P	$\frac{\delta^2 \lambda_{av}}{\xi_{\min}}$		$\frac{\sigma^2 n \lambda_{av}}{2\xi_{\min}}$
Total misadjustment, M_{tot}	M + P	M	M + P
Time constant of pth mode:			
In number of adaptive iterations, $\tau_{p_{mse}}$	$\frac{1}{4\mu\lambda_p}$	$\frac{1}{4\mu\lambda_p}$	$\frac{1}{4\sigma^2\beta\lambda_p}$
In number of data samples, $T_{p_{mse}}$	$\frac{Nn}{2\mu\lambda_p}$	$\frac{1}{4\mu\lambda_p}$	$\frac{N}{2\sigma^2\beta\lambda_p}$

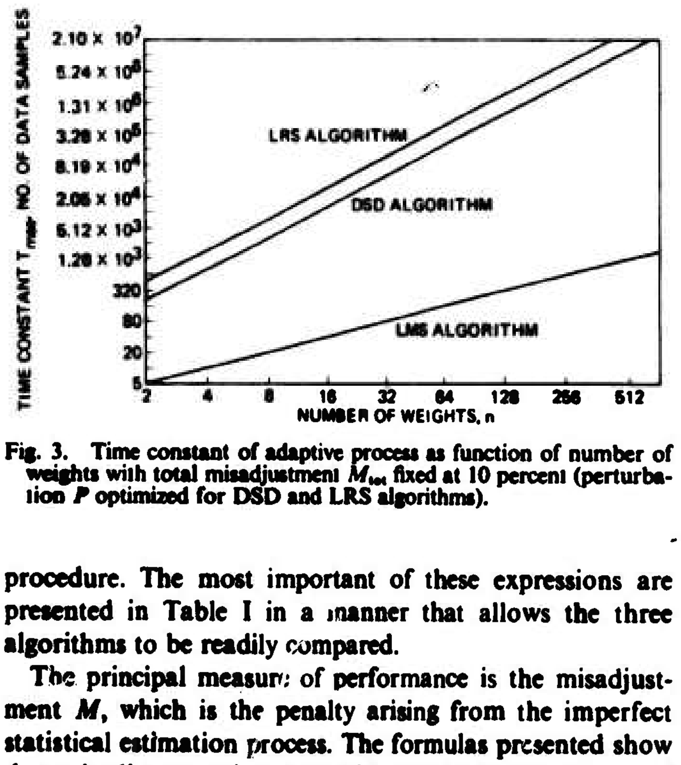


Fig. 3. Time constant of adaptive process as function of number of weights with total misadjustment M_{tot} fixed at 10 percent (perturbation P optimized for DSD and LRS algorithms).

procedure. The most important of these expressions are presented in Table I in a manner that allows the three algorithms to be readily compared.

The principal measure of performance is the misadjustment M, which is the penalty arising from the imperfect statistical estimation process. The formulas presented show that misadjustment increases with speed of adaptation, and this result can be taken as a general rule of adaptive processing. For a given real-time speed of adaptation[7] and given number of adaptive parameters, however, misadjustment varies considerably among the three algorithms. The most efficient in this respect is the LMS algorithm. The DSD and LRS algorithms, whose misadjustment expressions are nearly equivalent, are considerably less efficient.

Fig. 3 shows the relative efficiency of the three algorithms by plotting the required adaptive time constant as a function of number of adaptive weights with total misadjustment M_{tot} fixed at 10 percent. The eigenvalues of the $\boldsymbol{R}$-matrix are assumed to be equal, and the value of the total misadjustment for the DSD and LRS algorithms is minimized according to (92) and (151). It is readily seen that for a large number of weights the DSD and LRS algorithms have similar time constants. The LMS algorithm, on the other hand, has a much smaller time constant.

The formulas presented in Table I and the curves of Fig. 3 provide a practical tool for use in the design of adaptive filters. For the purposes of illustration let us assume that an adaptive digital filter with 10 weights is needed for a particular application. Let us further assume that a total misadjustment of 10 percent would be acceptable and that the eigenvalues of the $\boldsymbol{R}$-matrix are essentially equal. For the DSD algorithm, a total misadjustment of 10 percent, according to (91), yields an optimal perturbation of 5 percent. Thus the misadjustment M is

$$M = \frac{(n+1)^2}{8P}\left(\frac{1}{T_{p_{mse}}}\right)_{av} = 5 \text{ percent.} \tag{152}$$

This equation can be solved by substituting the appropriate values of n and P to obtain the average reciprocal time constant in number of data samples:

$$\left(\frac{1}{T_{p_{mse}}}\right)_{av} = 2(10)^{-4}. \tag{153}$$

Since all eigenvalues are assumed to be equal, there is only one time constant associated with the mean square error curve, and (153) can be rewritten as

$$T_{mse} = \frac{10^4}{2} = 5000 \text{ data samples.} \tag{154}$$

This is a large adaptive time constant for a 10-weight filter.

If the LMS algorithm is used instead of the DSD algorithm, then there is no perturbation and the misadjustment is

$$M = \frac{n+1}{4}\left(\frac{1}{T_{p_{mse}}}\right)_{av} = 10 \text{ percent} \tag{155}$$

[7] The basic unit of time in digital systems is the sampling period; in analog systems it is the equivalent Nyquist sampling period corresponding to the bandwidth of the error signal.

which yields a time constant of

$$T_{mse} = 25 \text{ data samples.} \tag{156}$$

This is a much mo~e favorable value. Within about four time constants adaptive transients would essentially die out. Settling time would be about 100 sampling periods or iterations.

For the LRS algorithm one must once again allocate one-half the total misadjustment to the perturbation P. The misadjustment M is thus

$$M = \frac{(n+1)^2}{4P}\left(\frac{1}{T_{p_{mse}}}\right)_{av} = 5 \text{ percent} \tag{157}$$

which yields a value of the time constant of

$$T_{mse} = 10\,000 \text{ data samples.} \tag{158}$$

The LRS algorithm thus would require twice the settling time required for the DSD algorithm. Note that the perturbation is set as follows:

$$P = 0.05 = \frac{\sigma^2 \operatorname{tr} \boldsymbol{R}}{2\xi_{min}} \tag{159}$$

which is equivalent to

$$\sigma^2 = 0.1\xi_{min}/\operatorname{tr} \boldsymbol{R}. \tag{160}$$

To set σ^2 for the random vector generator one would need to know the values of ξ_{min} and trace $\boldsymbol{R}$. Approximate values would be adequate in most practical circumstances.

These results illustrate the efficiency of the LMS algorithm, which has been shown to approach a theoretical limit for adaptive algorithms when the eigenvalues of the $\boldsymbol{R}$-matrix are equal or close to equal in value [32].[8] There are circumstances, however, where the LMS algorithm cannot be used and where the DSD and LRS algorithms provide a valuable option. An example is included in the applications described in the next section.

VI. Experimental Results

In this section the results of experiments performed by computer simulation are presented. These results show the relative performance of the DSD, LMS, and LRS algorithms in practical circumstances of varying complexity. They also provide a means of verifying the expressions for misadjustment and adaptive time constant derived in the preceding sections.

A. Modeling Experiments

Two modeling or system identification problems were simulated by computer to demonstrate the convergence of the three algorithms and the degree of correspondence between actual and theoretical performance. In these simulations an adaptive transversal filter with four weights was used. In the first the algorithms were required to converge to a weight vector solution that modeled the impulse response of a "digital" filter with a single fixed delay Δ of z^{-2}, where z^{-1} is the transfer function of the unit delay. In the second they were required to converge to a solution that best approximated the infinite impulse response of a one-pole recursive digital filter.

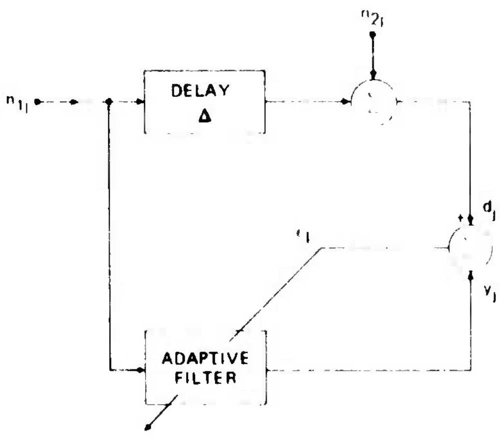


Fig. 4. Modeling a fixed delay wilh an adaptive filter.

1) *Modeling a fixed delay:* Fig. 4 shows the experimental configuration used to test convergence of the algorithms to model the fixed delay. An input signal n_1, composed of independent samples of white noise of unit power, was routed in parallel to the delay filter and the adaptive filter. The output of the delay filter was corrupted by a second input n_2, composed of independent additive white noise with a power of 0.5, to form the output of the system to be modeled. This output, the desired response d_j of the adaptive process, was compared with the adaptive filter output y_j in the normal way to form the error signal ε_j.

The optimal weight vector solution $\boldsymbol{W}^*$ for this experiment is zero for all weights except that whose tap delay corresponds to the delay Δ. The value of this weight is one. Thus, when the adaptive process has converged, the error ε_j is the noise n_2, which is uncorrelated over time. The minimum mean square error ξ_{min} is not zero but has a value equal to the power of the noise n_2. In addition, because the input n_1 is white and of unit power, all inputs to the weights are mutually uncorrelated and of unit power. The input correlation matrix $\boldsymbol{R}$ is thus equal to the unit matrix $\boldsymbol{I}$, and all eigenvalues of $\boldsymbol{R}$ are equal to one. These circumstances are the simplest that could be devised to test the three adaptive algorithms.

Fig. 5 shows learning curves of the adaptive process when the three algorithms were implemented with a fixed theoretical time constant T_{mse} of 2048 data samples. An individual learning curve and an ensemble average of 32 independent learning curves are presented for each algorithm. The averaged curves allow the misadjustment of the adaptive process to be experimentally measured.[9] The

[8] The gradient and performance estimation methods used in the DSD and LRS algorithms involve taking the difference between two large, noisy ξ-quantities. Some of this difference is due to statistical fluctuation (that is, to a change in data statistics from one sample to the next), an undesirable effect, and some to the actual weight change, a desirable effect. If the data could be repeated and the difference confined to the latter effect, the result would be a reduction in the amount of data required and a much better estimate. The gradient estimation technique of the LMS algorithm is equivalent to such "data repeating," which accounts for its inherent efficiency.

[9] The measurement is made by dividing by ξ_{min} the difference between the average value of asymptotic mean square error and ξ_{min}.

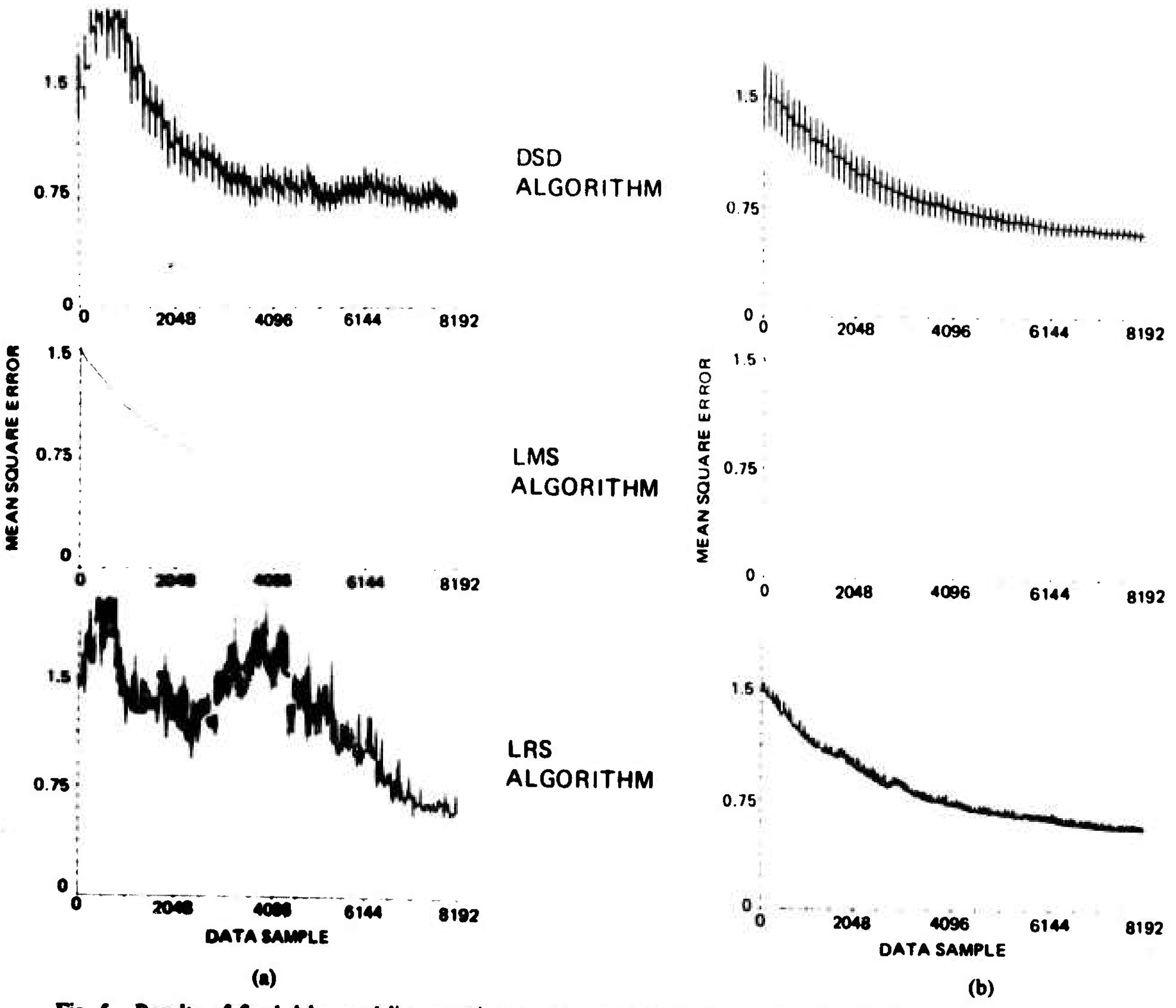


Fig. 5. Results of fixed delay modeling experiment with theoretical time constant T_{mse} fixed at 2048 data samples. (a) Individual learning curves. (b) Ensemble averages of 32 learning curves.

TABLE II
RESULTS OF FIXED DELAY MODELING EXPERIMENT WITH THEORETICAL TIME CONSTANT T_{mse} FIXED AT 2048 DATA SAMPLES

Algorithm	Convergence constants			Perturbation P, percent		Misadjustment M, percent		Total misadjustment M_{tot}, percent		Theoretical time constant T_{mse}, no. of data samples
	$\mu \times 10^{-3}$	β	$\sigma^2 \times 10^{-3}$	Theor.	Meas.	Theor.	Meas.	Theor.	Meas.	
DSD	15.625	–	–	2.21	2.19	4.42	5.70	6.63	7.89	2048
LMS	0.12207	–	–	–		0.0488	0.05	0.0488	0.05	2048
LRS	–	0.5	7.8125	3.125	3.12	6.25	8.08	9.375	11.20	2048

"high-frequency" variations of the curves representing the DSD and LRS algorithms are due to the required perturbation of the weight vector at each iteration. At the beginning of each experiment all adaptive weights were set to zero.

Table II presents the theoretical and measured values of perturbation and misadjustment for the learning curves of Fig. 5. Also shown are the values of the parameters μ, β, and σ^2. It is readily seen that the theoretical and measured values are in close agreement for all three algorithms.

Fig. 6 presents individual learning curves and ensemble averages of 32 learning curves showing convergence of the three algorithms with a fixed theoretical total misadjustment M_{tot} of 9.375 percent. Table III shows the values of perturbation, misadjustment, and time constant together with the values of the parameters μ, β, and σ^2. Once again close agreement between the theoretical and experimental results is observed.

2) *Modeling a one-pole recursive filter:* Fig. 7 shows the experimental configuration for the second modeling experiment. An input n, composed once again of independent samples of white noise of unit power, is routed in parallel to an adaptive transversal filter and a one-pole recursive

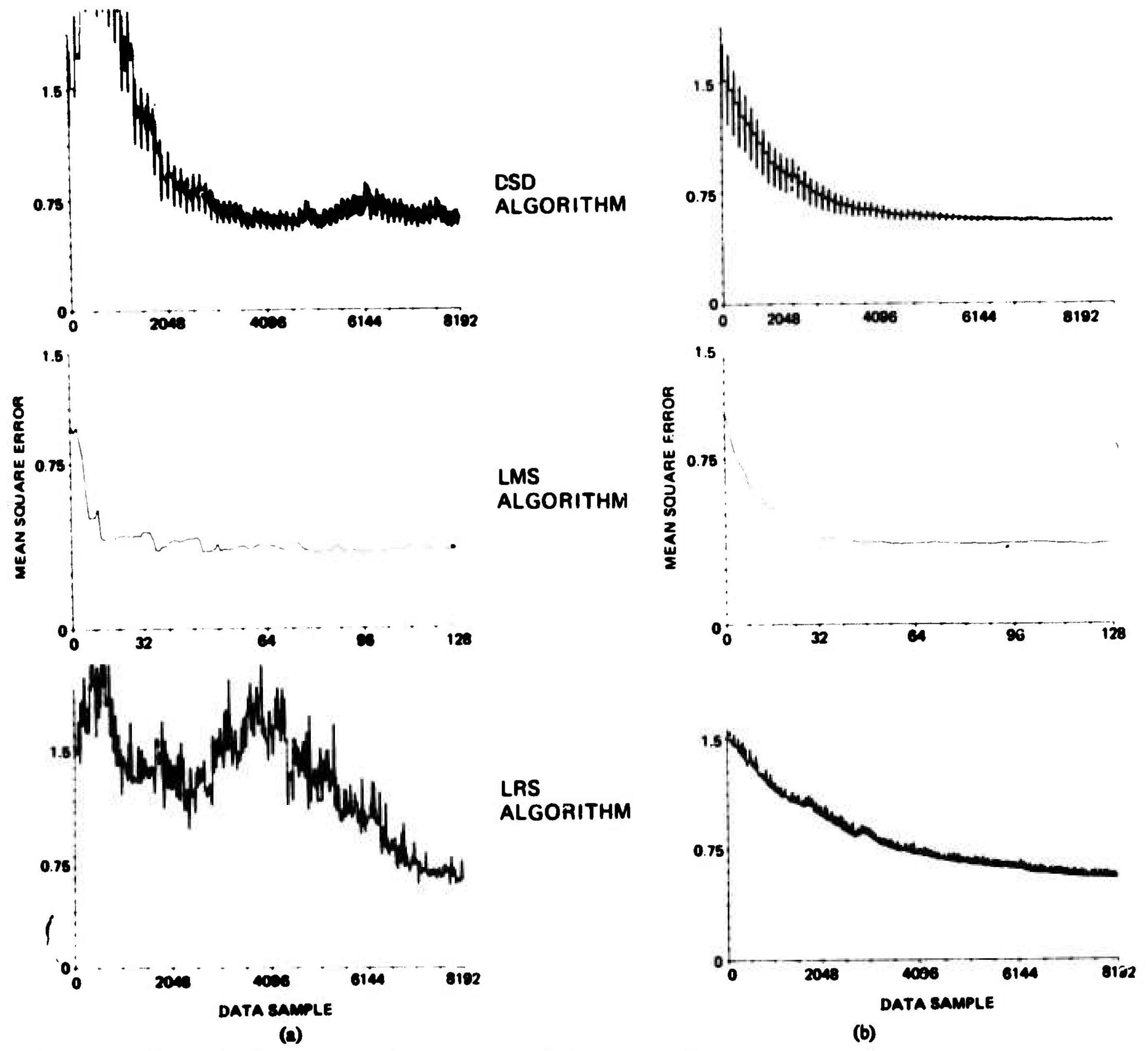


Fig. 6. Results of fixed delay modeling experiment with theoretical total misadjustment M_{tot} fixed at 9.375 percent. (a) Individual learning curves. (b) Ensemble averages of 32 learning curves.

TABLE III
RESULTS OF FIXED DELAY MODELING EXPERIMENT WITH THEORETICAL TOTAL MISADJUSTMENT M_{tot} FIXED AT 9.375 PERCENT

	Convergence constants			Perturbation P, percent		Misadjustment M, percent		Total misadjustment M_{tot}, percent		Theoretical time constant T_{mse},
Algorithm	$\mu \times 10^{-2}$	β	$\sigma^2 \times 10^{-3}$	Theor.	Meas.	Theor.	Meas.	Theor.	Meas.	no. of data samples
DSD	3.125	–	–	3.125	3.11	6.25	8.26	9.375	11.37	1024
LMS	2.34	–	–	–	–	9.375	10.35	9.375	10.35	10.7
LRS	–	0.5	7.8125	3.125	3.12	6.25	8.08	9.375	11.22	2048

digital filter whose transfer function is $1/(1 - az^{-1})$. The output of the one-pole filter is the desired response d_j, which is combined with the adaptive filter output y_j to produce the error ε_j.

In this experiment the four-weight adaptive filter is attempting to model a one-pole filter with an infinite impulse response. Since the input n is white noise, the optimal solution is to cause the adaptive filter's impulse response to match the one-pole filter's geometrical impulse response to the extent allowed by the length of the adaptive tapped delay line. A residual mean square error will be present because the best match attainable is imperfect.

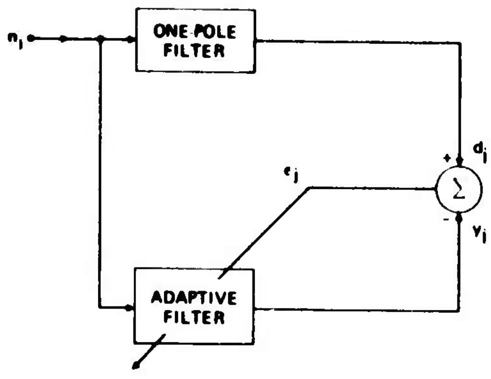


Fig. 7. Modeling a one-pole recursive filter with an adaptive filter.

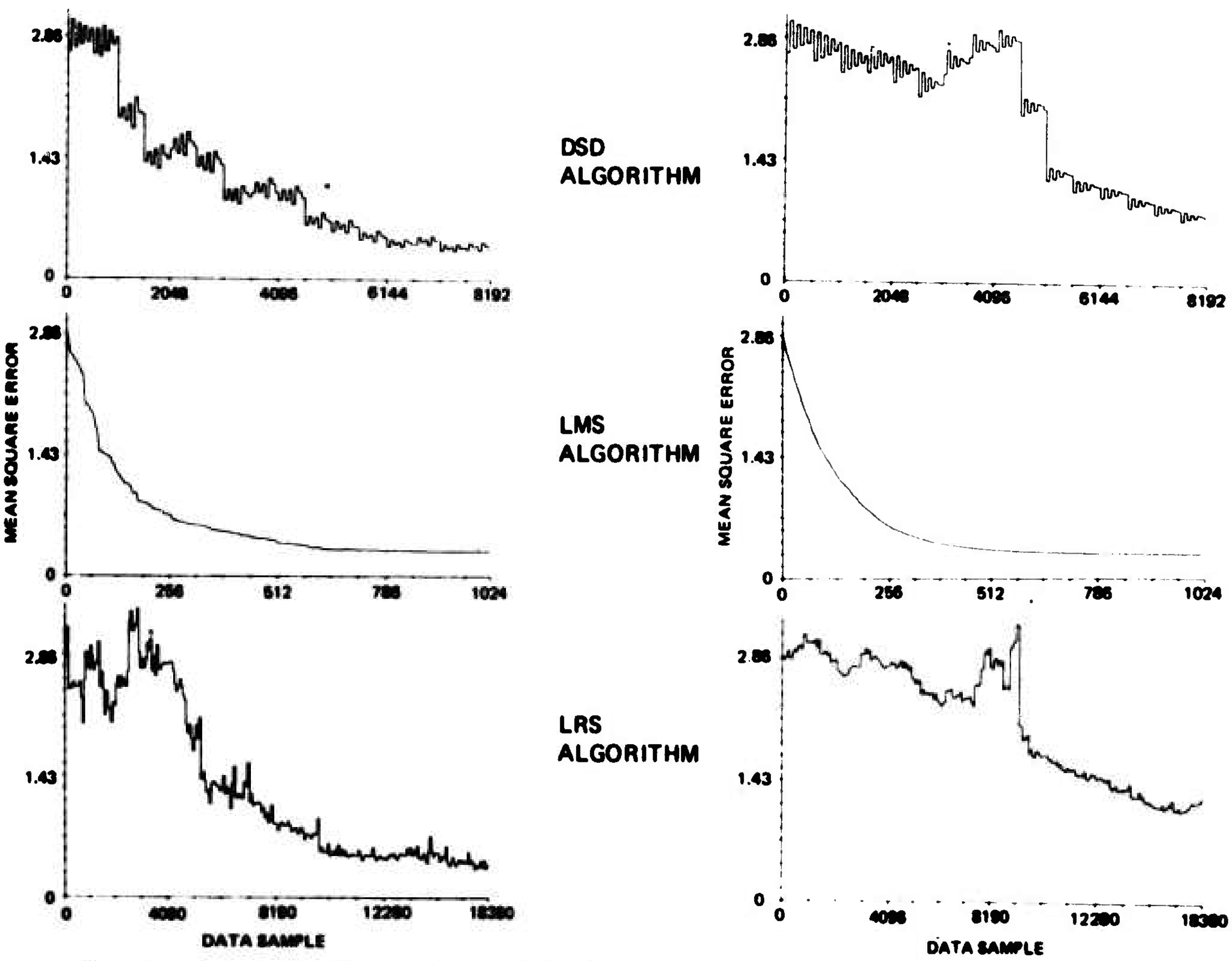


Fig. 8. Results of one-pole filter modeling experiment with theoretical total misadjustment M_{tot} fixed at 7.5 percent for DSD and LRS algorithms and at 0.75 percent for LMS algorithm. (a) Individual learning curves. (b) Ensemble averages of 32 learning curves.

In this case, when the adaptive filter has converged to the optimal solution, the error ϵ_j will be correlated over time. This latter condition violates one of the assumptions on which the previous derivations of misadjustment and time constant were based and can be expected to affect the agreement between theoretical and measured misadjustment and time constant.

Fig. 8 shows individual and averaged learning curves of the adaptive process with a fixed theoretical total misadjustment M_{tot} of 7.5 percent for the DSD and LRS algorithms and of 0.75 percent for the LMS algorithm. Note the difference in time scales and the rapid convergence of the LMS algorithm. Table IV presents the values of perturbation, misadjustment, and time constant and of the convergence parameters. It may be seen that the measured misadjustment is approximately twice the theoretical misadjustment for the DSD and LRS algorithms. For the LMS algorithm, however, measured and theoretical mis-

TABLE IV
RESULTS OF ONE-POLE FILTER MODELING EXPERIMENT WITH THEORETICAL TOTAL MISADJUSTMENT M_{tot} FIXED AT 7.5 PERCENT FOR DSD AND LRS ALGORITHMS AND 0.75 PERCENT FOR LMS ALGORITHM

	Convergence constants			Perturbation P, percent		Misadjustment M, percent		Total misadjustment M_{tot}, percent		Theoretical time constant T_{mse},
Algorithm	$\mu \times 10^{-3}$	β	$\sigma^2 \times 10^{-3}$	Theor.	Meas.	Theor.	Meas.	Theor.	Meas.	no. of data samples
DSD	40			2.5	2.51	5	10.31	7.5	12.82	1600
LMS	1.875					0.75	0.77	0.75	0.77	133
LRS		1.7467	2.5675	2.5	2.67	5	9.23	7.5	11.90	3200

adjustment are in close agreement. The results for the DSD and LRS algorithms are expected and can be attributed to the fact that the correlation in the error ε_j over time makes the effective statistical sample size less than the actual number of error samples. The reason that the LMS algorithm is not sensitive in this respect and does not experience a loss in performance is not understood at the present time and is a subject under investigation.

This experiment and the foregoing fixed delay experiment demonstrate that, in accordance with the theoretical expectation, the performance of the LMS algorithm is superior to that of the DSD and LRS algorithms, whose performance is approximately equivalent. The LMS algorithm converges more rapidly for a given level of misadjustment or is less noisy (produces less misadjustment) for a given rate of adaptation. For the DSD and LRS algorithms the relationship between rate of adaptation and misadjustment is known approximately for a wide variety of input statistical conditions. For the LMS algorithm the relationship under the same variety of input conditions is known to a closer approximation.

B. Adaptive Cancelling of Sidelobe Interference in a Receiving Antenna Array

The objective of this experiment is to demonstrate one of the ways in which adaptive filtering can be applied to reduce interference received by the sidelobes of an antenna array. Results are presented only for the LMS algorithm. The DSD and LRS algorithms could also be used with this problem, but their performance would not equal that of the LMS algorithm, as indicated by the formulas and experimental results already presented. An experiment where the DSD and LRS algorithms are applied to a problem that cannot be solved by the LMS algorithm is presented in the next section.

A number of adaptive beamforming methods capable of reducing interference in the sidelobes of an antenna array have been described in the literature [1]–[10]. These methods have the disadvantage that, unless the adaptive process is constrained, strong signal components in the main beam are rejected. When the adaptive process is constrained the signal is preserved, but there may be a loss in array performance caused by gain or phase errors due to nonuniformity in element placement, transfer function, or near-field effects.

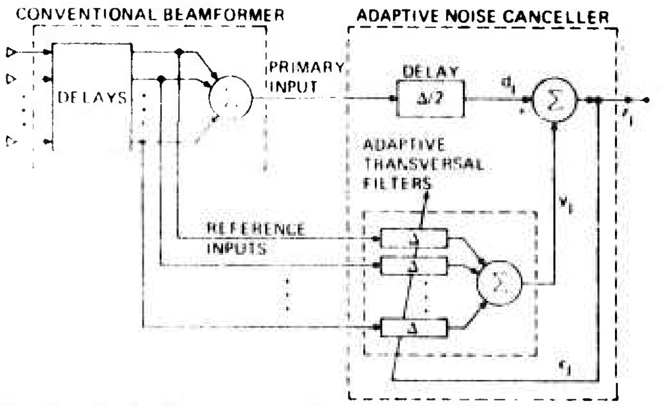


Fig. 9. Block diagram of null-constrained adaptive beamformer tolerant of array element gain and phase errors.

By the use of adaptive noise cancelling techniques[10] it is possible to realize a constrained adaptive beamformer that does not suffer a significant loss in performance when array element properties are not uniform. This beamformer, described here for the first time, is capable of reducing broadband and narrowband interference in the sidelobes of an antenna array without rejecting broadband signal components in the main beam, regardless of their strength. It is also simple and easy to implement.

Fig. 9 is a block diagram of the constrained adaptive beamformer. An array of receiving elements is connected to a conventional time delay and sum beamformer, which is steered in the direction of the signal. The conventional beamformer's output, containing signal and interference, forms the primary input to an adaptive noise canceller. This input is delayed by an amount $\Delta/2$, where Δ is defined below, to form the desired response d_j of the adaptive process. Multiple reference inputs to the noise canceller are derived by taking the delayed element outputs from the conventional beamformer before summation. These inputs are routed to a bank of adaptive transversal filters, each comprising a tapped delay line with a total delay of Δ. The filter outputs are summed to form a single output y_j, which is subtracted from d_j to obtain the canceller output z_j.

[10] Adaptive noise cancelling [33] is a form of optimal filtering that makes use of two inputs, a "primary" input consisting of signal and noise and a "reference" input consisting of noise correlated in some unknown way with that in the primary input. The reference input is adaptively filtered and subtracted from the primary input to obtain a signal estimate in many cases superior to that obtainable by other forms of adaptive or conventional filtering.

$$\begin{bmatrix} w & \cdots & w & 0 & w & \cdots & w \\ w & \cdots & w & 0 & w & \cdots & w \\ & & & \vdots & & & \\ w & \cdots & w & 0 & w & \cdots & w \end{bmatrix}$$

(a)

$$\begin{bmatrix} w & \cdots & w & 0 & 0 & 0 & w & \cdots & w \\ w & \cdots & w & 0 & 0 & 0 & w & \cdots & w \\ & & & & \vdots & & & & \\ w & \cdots & w & 0 & 0 & 0 & w & \cdots & w \end{bmatrix}$$

(b)

$$\begin{bmatrix} w & \cdots & w & 0 & 0 & 0 & 0 & 0 & w & \cdots & w \\ w & \cdots & w & w & 0 & 0 & 0 & w & w & \cdots & w \\ & & & & & 0 & & & & & \\ & & & & & \vdots & & & & & \\ & & & & 0 & 0 & 0 & & & & \\ w & \cdots & w & 0 & 0 & 0 & 0 & 0 & w & \cdots & w \end{bmatrix}$$

(c)

Fig. 10. Weighting coefficient matrices for null-constrained adaptive beamformer. (a) Single column-of-zeros constraint. (b) Triple column-of-zeros constraint. (c) "Hourglass" constraint.

This output also provides the "error" signal ε_j for the adaptive process.

The operation of the adaptive beamformer of Fig. 9 is constrained by constraining the weighting coefficients (gains) of the adaptive filter taps. Fig. 10 shows three forms of constraint, each suitable for a different purpose. Fig. 10(a) represents the matrix of coefficients appropriate for an ideal line array with a plane-wave signal incident in the "look" direction of the conventional beamformer. The gain of the central taps is constrained to be zero. The gains w of each of the other taps are independently controlled by the adaptive process. Note that the matrix has as many rows as there are reference inputs.

In this problem the signal appearing at the central tap of each adaptive filter is identical except in scale to d_j. If one assumes that the received signal is "white" and has an impulsive autocorrelation function, the signals appearing at the other taps will be uncorrelated with d_j. It is thus apparent that the signal components in y_j will be uncorrelated with those in d_j and that the adaptive process will have no tendency to cancel the received broadband signal. Interference components arriving from other than the "look" direction, on the other hand, will be correlated with the interference components in d_j at one or more of the unconstrained taps. These components will thus be cancelled by the adaptive process, which adjusts the gain of the unconstrained taps to minimize the mean square of the error ε_j (in this case, output power).

In practical applications arrays with ideal properties cannot be realized because perfect receiving elements, perfect element placement, and freedom from near-field irregularities cannot be achieved. Fig. 10(b) shows a form of constraint proposed to desensitize the behavior of the adaptive sidelobe canceller to imperfections in the properties of the receiving elements. This constraint consists of inserting an additional column of zeros on either side of the central column. Fig. 10(c) shows a configuration of the weighting coefficients that would allow the reception of strong broadband signals over a finite and controllable angular sector; in this configuration the zeros are arranged in the form of an "hourglass."

Fig. 11 shows directional response patterns obtained by computer simulation that indicate the performance of the adaptive beamformer of Fig. 9 with an ideal and a nonideal array using the single and triple "column-of-zeros" constraints. The ideal array consists of ten elements in a linear configuration and with half-wavelength spacing at the sampling frequency; for the nonideal array the single elements at each end of the array are moved forward one-quarter of a wavelength. The simulated received signal has a power of one, a white spectrum, and originates from a point source. The simulated interference is isotropic, with a power of 0.01 and a white spectrum. The directional response of the conventional time delay and sum beamformer is shown as a dotted line for purposes of comparison.

Fig. 11(a) represents the adaptive beamformer's performance with the ideal array and the single column-of-zeros constraint, while Fig. 11(b) represents performance with the nonideal array and single column-of-zeros constraint. Note that the beam formed is "super-directive"—that is, much narrower than the conventional beam—but severely reduced in sensitivity when array properties are not ideal.

Fig. 11(c) and Fig. 11(d) show beamformer performance with the triple column-of-zeros constraint. In this case the adaptive beam is close in width to the conventional beam, and its sensitivity is not affected by element irregularity. Even at high signal-to-noise ratios sensitivity is sustained over a finite range of angles, an unusual result since adaptive beamformers generally lose signals not incident exactly in the "look" direction.

C. Adaptive Phase Control of a Transmitting Antenna Array

This experiment illustrates the use of the DSD and LRS algorithms to solve a problem that cannot be solved with the LMS algorithm.[11] The problem selected, adaptive phase control of a transmitting array, is representative of a class of problems more general than those heretofore treated in this paper. Other problems of a similar nature include adaptive adjustment of the parameters of microwave resonators, waveguides, and coaxial transmission lines. A related problem at optical frequencies is adaptive adjustment by controlled warping of laser mirrors.

It should be noted that the formulas for time constant, perturbation, and misadjustment of the DSD and LRS algorithms given in Table I were derived by assuming stationary stochastic inputs to an adaptive system so configured that mean square performance is a quadratic

[11] In the form described in this paper the LMS algorithm can be used only to adjust variable weights. The DSD and LRS algorithms do not suffer from this limitation.

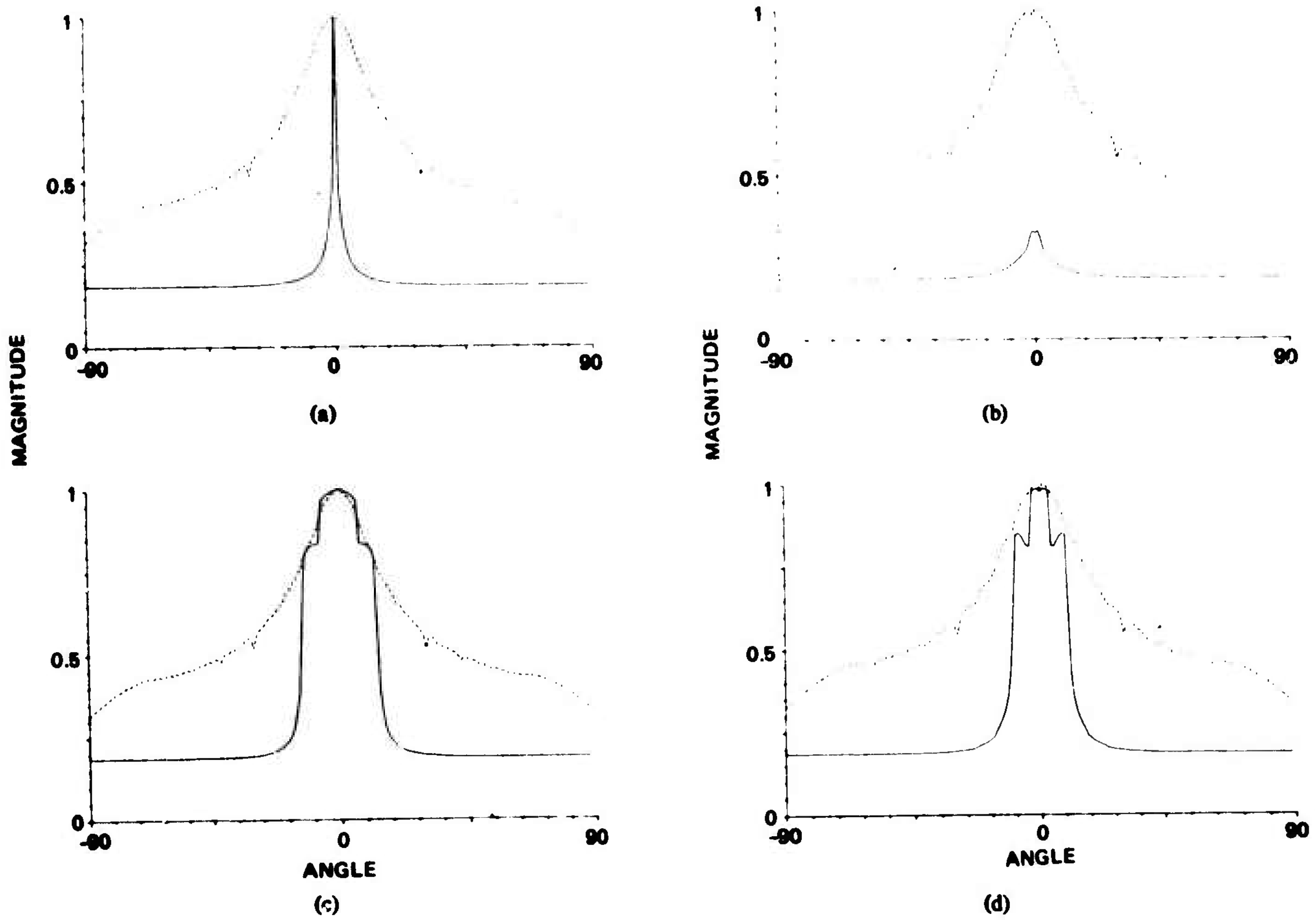


Fig. 11. Results of adaptive beamforming experiment. (a) Single column-of-zeros constraint, ideal array. (b) Single column-of-zeros constraint, nonideal array. (c) Triple column-of-zeros constraint, ideal array. (d) Triple column-of-zeros constraint, nonideal array.

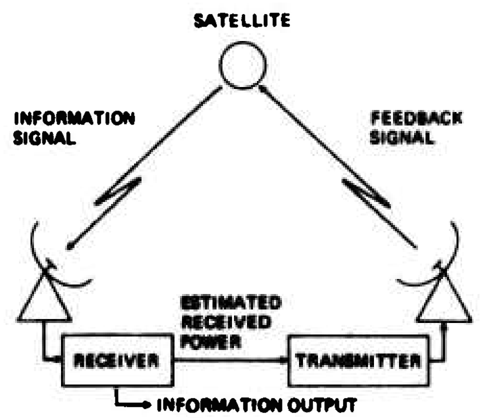


Fig. 12. Satellite transmitting information to receiver on earth.

function of the adjustable parameters. The conditions on which these formulas and proof of convergence are based are not satisfied in the adaptive phase control problem examined here. If one ignores the deterministic nature of the sinusoidal input signals and treats input power as in the stochastic case, however, the expressions of Table I provide predictions borne out well by experimental simulation.

Fig. 12 shows a typical application for a transmitting array with adaptive phase control. A satellite is relaying information over a large distance to a receiver on the earth. The power available to drive the transmitter is limited, and it is desirable for maximum power transfer to keep the main beam of the transmitting antenna optimized and steered toward the receiving station, whose position with respect to the satellite changes with the earth's rotation and the satellite's orientation. The array's elements need not be ideal. It is assumed that the power of the received signal can be measured or estimated and transmitted via a feedback link to the satellite for use as an input to an adaptive beamforming process. To avoid a loss of signal power that would partially or wholly offset the directional gain, the beamforming process must control the output phase rather than the gain of the satellite antenna's elements.

Fig. 13 is a block diagram showing the model used to simulate an adaptive transmitting antenna array of n elements. The signal is represented by a sine wave produced by a signal generator. An array of n phase compensators governed by an adaptive algorithm represents the adaptive processor. A corresponding array of n phase shifters provides a means of simulating the unknown phase shifts between the antenna elements and the receiver. The outputs of the phase shifters are summed and injected with "receiver" noise to simulate a weak received signal. This signal is sampled, squared, and averaged, providing a power estimate for the adaptive algorithm. The algorithm adjusts the phase compensators to maximize measured power. It is clear that

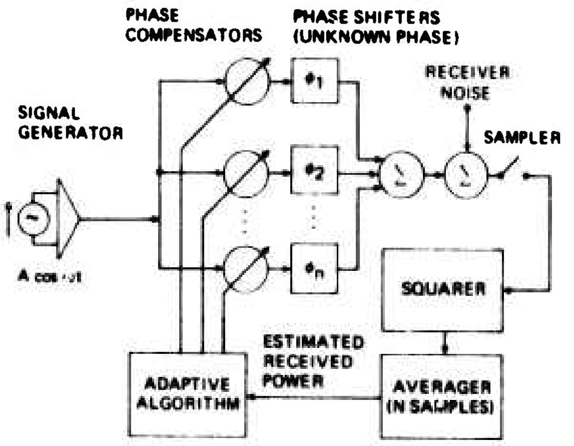


Fig. 13. Digital simulation of adaptive transmitting antenna.

maximum power will be transmitted when the combined phase shifts on each branch of the block diagram are integral multiples of 360 degrees relative to each other. Although there is no unique solution to the problem, there are families of equivalent solutions that provide maximum power transfer.

This model comprises all aspects of the satellite transmission example described above except the two-way time delay of the transmission path. This delay would affect the rate of adaptation of the processor and would have to be taken into account in designing a real system.

Fig. 14 shows learning curves of the adaptive process for the DSD and LRS algorithms when the injected noise of Fig. 13 is set equal to zero. The transmitting antenna was composed of 16 isotropic elements in a line array. Note that the curves rise to an asymptote representing maximum power rather than decaying toward a minimum. Note further that they are not exponential except as the optimal solution is approached. Exponential learning curves occur only when the algorithms are applied to quadratic performance surfaces. The performance surface for the simulated problem is a representation of output power as a function of phase and is not quadratic except near stationary points, where it can be represented by first- and second-degree terms of a Taylor expansion.[12] For this application the method of steepest descent might better be designated the "method of steepest ascent." It is described by (24) with the sign of μ reversed. A corresponding reversal of sign is also required in applying the LRS algorithm to this problem.

The "theoretical" time constant of both learning curves of Fig. 14 is 128 data samples. This value is based on the characteristics of the performance surface (that is, its "R-matrix") in the vicinity of the global optimum.[13] Visual inspection indicates that the actual time constants of the two curves are similar and agree well with the above value. The convergence parameter μ for the DSD algorithm was 8×10^{-3}. The convergence parameters β and σ^2 for the LRS algorithm were 1 and 8×10^{-3}, respectively. The maximum transmitted power $\xi_{\max}$ was equal to 32. The "perturbation" P for both algorithms was 5 percent, and the value of N was one.

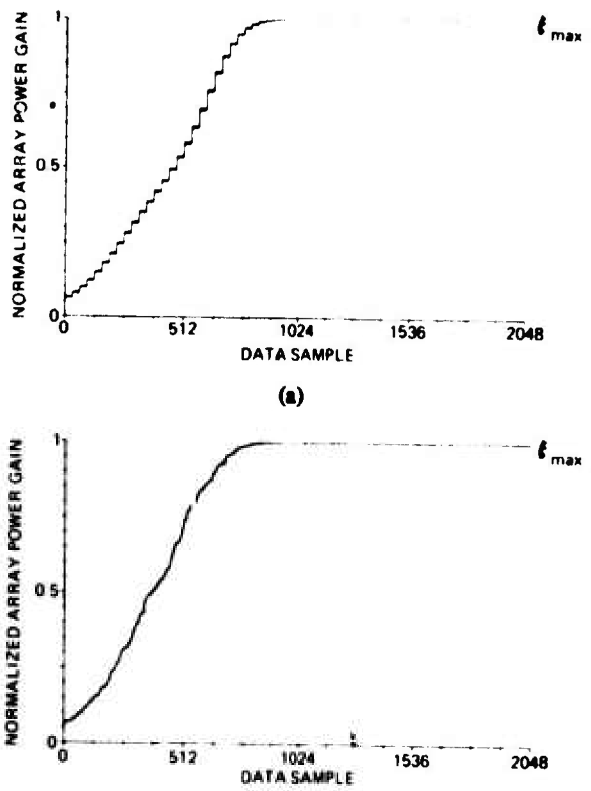


Fig. 14. Learning curves of simulated adaptive transmitting antenna without noise. (a) DSD algorithm. (b) LRS algorithm.

Fig. 15 shows sequences of radiation patterns corresponding to the learning curves of Fig. 14. Real time is indicated in terms of data samples equivalent to sampling periods of the digital system of Fig. 13. The simulated receiving site was located at a relative angle of 20 degrees. The initial setting of the phase compensators was zero. The unknown phase settings of the phase shifters were chosen at random. Note the rapid formation of the main lobe at 20 degrees and the suppression of sidelobes.

Fig. 16 shows learning curves of the adaptive process when independent samples of white noise with a power of 0.01 were injected into the simulated received signal. Array configuration and adaptive parameters are the same as in the noiseless case represented by Fig. 14. As well as can be determined by visual inspection, the actual time constants

[12] It has been shown by M. K. Leavitt, in a June 1975 term paper for the course EE 373, Adaptive Systems, in the Department of Electrical Engineering at Stanford University, that the performance surface is a sum of terms containing sums of cosines of differences in the adaptive phase settings. This surface has many global and relative optima and many saddle points where the gradient goes to zero. Only the global optima, however, are stable. Leavitt further shows that the presence of saddle points may result in slow convergence for algorithms based on the method of steepest descent. The LRS and other random search algorithms, on the other hand, may have an advantage on such irregular performance surfaces, though not enough experience has yet been gained to confirm this expectation.

[13] In the vicinity of a global optimum (when all phases are aligned), the "R-matrix" of the performance surface can be shown to be

$$R = -\xi_{\max} I.$$

The assumptions are that n is large and that equal power flows through all phase shifters. The maximum output power is $\xi_{\max}$. Note that all eigenvalues are equal and negative.

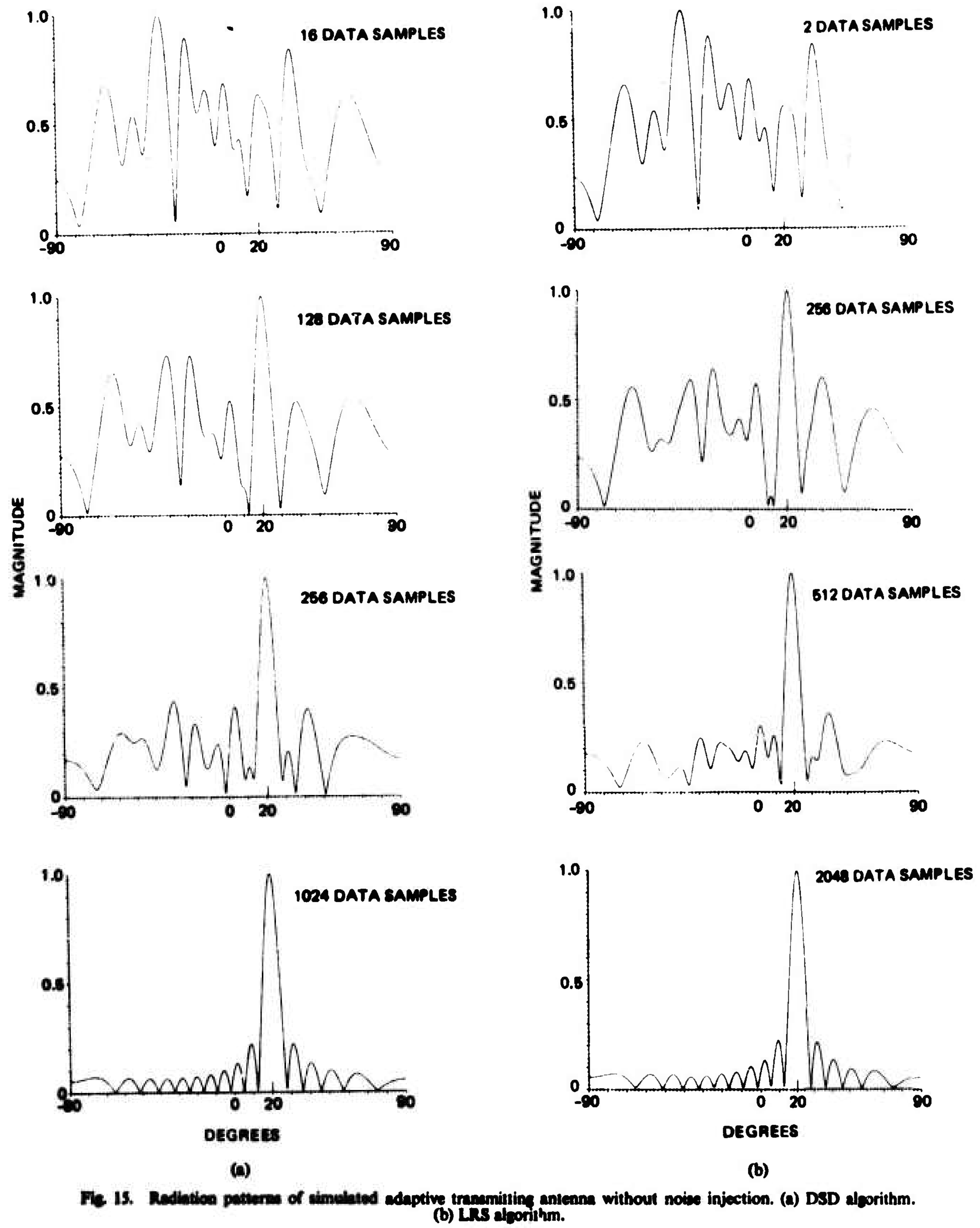


Fig. 15. Radiation patterns of simulated adaptive transmitting antenna without noise injection. (a) DSD algorithm. (b) LRS algorithm.

for both algorithms are also approximately the same as in the noiseless case.

Noise in the adaptive phase control process, as evident in Fig. 16, causes a steady-state average loss of array power gain. One can define for this case a form of misadjustment that is a ratio of the loss in power to the peak power ξ_{max}. Though the appropriate formulas have not yet been derived, the formulas for stochastic inputs and quadratic performance surfaces would suggest that with equal theoretical time constants the misadjustment of the LRS algorithm would be greater than that of the DSD algorithm. This expectation is confirmed by the results obtained in this experiment.

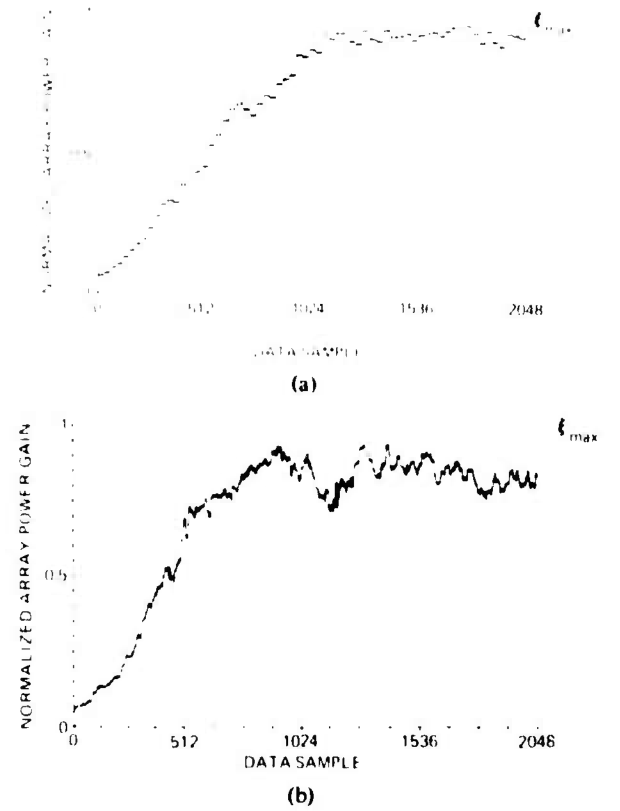


Fig. 16. Learning curves of simulated adaptive transmitting antenna with noise. (a) DSD algorithm. (b) LRS algorithm.

VII. Conclusion

The theoretical and experimental results presented in this paper show that the LMS algorithm is the most efficient by a large factor of the three algorithms compared and indicate that it should be used whenever circumstances permit. The DSD algorithm is less efficient than the LMS but more efficient by a factor of two than the LRS algorithm. Its use is appropriate where technical or economic considerations preclude use of the LMS algorithm or where a high speed of adaptation is not required. Use of the LRS algorithm may be appropriate in cases where the performance surface for the adaptive process is not well behaved and has both local and global optima. Further experience is required, however, to confirm that the random weight vector changes associated with this algorithm can provide an advantage in the presence of local optima that may slow or prevent global convergence of algorithms based on the method of steepest descent. Further work is also required to extend the theoretical derivations for time constant and misadjustment of the three algorithms to applications other than those entailing stochastic inputs and quadratic performance surfaces.

Acknowledgment

The authors wish to acknowledge the contributions of Dr. O. L. Frost of Argo Systems, Inc., and Prof. L. J. Griffiths of the University of Colorado to the concept of the constrained adaptive sidelobe canceller presented in this paper. Thanks are also due to R. Fraser of the Naval Undersea Center for assistance in editing the paper.

References

[1] P. Howells, "Intermediate frequency side-lobe canceller," U.S. Patent 3 202 990, Aug. 24, 1965.
[2] S. P. Applebaum, "Adaptive arrays," Special Projects Lab., Syracuse Univ. Res. Corp., Rep. SPL TR 66-1, Aug. 1966.
[3] J. Capon, R. J. Greenfield, and R. J. Kolker, "Multidimensional maximum likelihood processing of a large aperture seismic array," *Proc. IEEE*, vol. 55, pp. 192–211, Feb. 1967.
[4] B. Widrow, P. Mantey, L. Griffiths, and B. Goode, "Adaptive antenna systems," *Proc. IEEE*, vol. 55, pp. 2143–2159, Dec. 1967.
[5] L. J. Griffiths, "A simple adaptive algorithm for real-time processing in antenna arrays," *Proc. IEEE*, vol. 57, pp. 1696–1704, Oct. 1969.
[6] O. L. Frost, III, "An algorithm for linearly constrained adaptive array processing," *Proc. IEEE*, vol. 60, pp. 926–935, Aug. 1972.
[7] A. H. Nuttall and D. W. Hyde, "A unified approach to optimum and suboptimum processing for arrays," Navy Underwater Sound Laboratory, Rep. 992, April 1969.
[8] R. Riegler and R. Compton, Jr., "An adaptive array for interference rejection," *Proc. IEEE*, vol. 61, pp. 748–758, June 1973.
[9] W. F. Gabriel, "Adaptive arrays—An introduction," *Proc. IEEE*, vol. 64, pp. 239–272, Feb. 1976.
[10] A. M. Vural, "An overview of adaptive array processing for sonar applications," in *IEEE EASCON Conv. Rec.*, pp. 34A–34M, 1975.
[11] B. Widrow, "Adaptive filters," in *Aspects of Network and System Theory*, R. Kalman and N. DeClaris, Eds. New York: Holt, Rinehart, and Winston, 1971, pp. 563-587.
[12] N. Wiener, *Extrapolation, Interpolation and Smoothing of Stationary Time Series, with Engineering Applications.* New York: Wiley, 1949.
[13] H. Bode and C. Shannon, "A simplified derivation of linear least squares smoothing and prediction theory," *Proc. IRE*, vol. 38, pp. 417–425, April 1950.
[14] T. Kailath, "A view of three decades of linear filtering theory," *IEEE Trans. Inform. Theory*, vol. IT-20, pp. 145–181, March 1974.
[15] R. V. Southwell, *Relaxation Methods in Engineering Science*. New York: Oxford, 1940.
[16] D. J. Wilde, *Optimum Seeking Methods.* Englewood Cliffs, NJ: Prentice-Hall, 1964.
[17] B. Widrow and M. Hoff, Jr., "Adaptive switching circuits," in *IRE WESCON Conv. Rec.*, pt. 4, pp. 96–104, 1960.
[18] N. Nilsson, *Learning Machines*. New York: McGraw-Hill, 1965.
[19] J. Koford and G. Groner, "The use of an adaptive threshold element to design a linear optimal pattern classifier," *IEEE Trans. Inform. Theory*, vol. IT-12, pp. 42-50, Jan. 1966.
[20] L. J. Griffiths, "Rapid measurement of instantaneous frequency," *IEEE Trans. Acoustics, Speech, and Signal Processing*, vol. ASSP-23, pp. 209–222, April 1975.
[21] K. Senne, "Adaptive linear discrete-time estimation," Stanford Electronics Laboratories, Stanford Univ., Rep. SEL-68-090, June 1968 (Ph.D. dissertation).
[22] T. Danieli, "Adaptive estimation with mutually correlated training samples," Stanford Electronics Laboratories, Stanford Univ., Rep. SEL-68-083, Aug. 1968 (Ph.D. dissertation).
[23] Y. P. Lin, "Adaptive models for natural selection," E.E. Thesis, Department of Electrical Engineering, Stanford Univ., Aug. 1972.
[24] C. Karnopp, "Random search techniques for optimization problems," *Automatica*, vol. 1, pp. 111-121, Aug. 1963.
[25] G. J. McMurty and K. S. Fu, "A variable structure automaton used as a multimodal searching technique," *IEEE Trans. Automat. Contr.*, vol. AC-11, pp. 379–387, July 1966.
[26] R. L. Barron, "Self-organizing control: The elementary SOC—Part I," *Contr. Engr.*, Feb. 1968.
[27] ——, "Self-organizing control: The general purpose SOC—Part II," *Contr. Engr.*, March 1968.
[28] M. A. Schumer and K. Steiglitz, "Adaptive step size random search," *IEEE Trans. Automat. Contr.*, vol. AC-13, pp. 270–276, June 1968.
[29] R. A. Jarvis, "Adaptive global search in a time-variant environment using a probabilistic automaton with pattern recognition supervision," *IEEE Trans. Syst. Sci. Cybern.*, vol. SSC-6, pp. 209-217, July 1970.
[30] A. N. Mucciardi, "Self-organizing probability state variable parameter search algorithms for systems that must avoid high-penalty operating regions," *IEEE Trans. Systems, Man, and Cybernetics*, vol. SMC-4, pp. 350–362, July 1974.
[31] R. A. Jarvis, "Adaptive global search by the process of competitive evolution," *IEEE Trans. Systems, Man, and Cybernetics*, vol. SMC-5, pp. 297–311, May 1975.
[32] B. Widrow *et al.*, "Stationary and nonstationary learning characteristics of the LMS adaptive filter," *Proc. IEEE*, vol. 64, Aug. 1976 (forthcoming).
[33] ——, "Adaptive noise cancelling: Principles and applications," *Proc. IEEE*, vol. 63, pp. 1692–1716, Dec. 1975.