accounts for the phase shift between the reflected component and the direct ray of $E_0(n)$ in (6).

The received E-field at the nth element is obtained by adding the reflected components to the direct ray component

$$G_R(n,\theta) = E_0(n) + \sum_{l=1}^{L} E_R(l,n) \cdot \exp\left[-\frac{2\pi i d}{\lambda}(n - \bar{n})\sin(\psi - \theta)\right] \quad (12)$$

where $E_0(n)$ is given by (10) and $E_R(l,n)$ by (11).

The E-field input to the nth array element on the mth pulse due to a unit scatterer at θ_k is given by

$$\text{GAIN}\,(m,n,k) = G_T(\theta_k)G_R(n,\theta)\exp\left[4\pi i(m-1)\frac{S}{\lambda}\cos\theta_k\right] \quad (13)$$

where S is the distance the radar moves between pulses.

An array of these GAIN functions is computed in the program for M pulses (m), N array elements (n), and K far-field clutter scatterers at uniformly spaced angular intervals (θ_k) within ±90° from the array normal. This array of GAIN's can be used in simulating an adaptive AMTI radar or in computing the covariance matrix of the clutter field.

In the results presented here, simulation of a random clutter field was not used. Earlier studies have shown that the response of the system can be calculated from the covariance matrix more efficiently and that the computed response follows a simulation very closely. Elements of the covariance matrix are obtained by averaging products of the GAIN's

$$M_{mnm'n'} = E\{v_{mn}{}^* v_{m'n'}\} \cdot \alpha \sum_{k=1}^{K} \text{GAIN}^*\,(m,n,k)\,\text{GAIN}\,(m',n',k) \quad (14)$$

assuming uniformly distributed clutter, i.e., equal intervals between the θ_k.

The steering signals used in the analysis and the assumed echo from a target in the main beam did not include the near-field scattering effects. This is a good approximation in most practical cases, where the near-field scattering has a minor effect on the main beam gain, but significantly changes the sidelobe structure.

References

[1] L. E. Brennan and I. S. Reed, "Theory of adaptive radar," *IEEE Trans. Aerosp. Electron. Syst.*, vol. AES-9, no. 2, pp. 237–252, March 1973.

[2] S. P. Applebaum, "Adaptive arrays," this issue, pp. 585–598; also Syracuse University Research Corporation, Syracuse, NY, SPL-709, June 1964.

[3] L. E. Brennan, E. L. Pugh, and I. S. Reed, "Control loop noise in adaptive array antennas," *IEEE Trans. Aerosp. Electron. Syst.*, vol. AES-7, March 1971.

[4] I. S. Reed, J. D. Mallett, and L. E. Brennan, "Rapid convergence rate in adaptive arrays," *IEEE Trans. Aerosp. Electron. Syst.*, pp. 853–863, Nov. 1974.

[5] J. M. Shapard, D. Edelblute, and G. Kinnison, "Adaptive matrix inversion," Naval Undersea Research and Development Center, NUC-TN-528, May 1971.

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

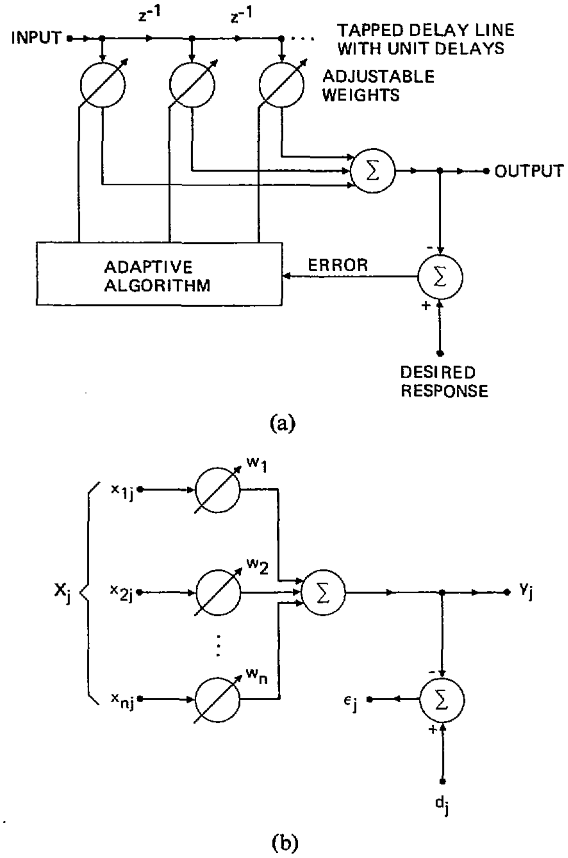


Fig. 1. Adaptive filter consisting of tapped delay line connected to adaptive linear combiner. (a) Adaptive filter configuration. (b) Adaptive linear combiner with input and output terminology.

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

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.

[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 $\boldsymbol{W}_k$. Squaring and expanding (4) and letting $\boldsymbol{W} = \boldsymbol{W}_k$ yields

$$\varepsilon_j^2 = d_j^2 - 2d_j\boldsymbol{X}_j^T\boldsymbol{W}_k + \boldsymbol{W}_k^T\boldsymbol{X}_j\boldsymbol{X}_j^T\boldsymbol{W}_k. \tag{5}$$

Now assume an ensemble of identical adaptive linear combiners, each having the same weight vector $\boldsymbol{W}_k$ at the kth iteration. Let each combiner have individual inputs $\boldsymbol{X}_j$ 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

$$E[\varepsilon_j^2]_{\boldsymbol{W}=\boldsymbol{W}_k} = E[d_j^2] - 2E[d_j\boldsymbol{X}_j^T]\boldsymbol{W}_k + \boldsymbol{W}_k^TE[\boldsymbol{X}_j\boldsymbol{X}_j^T]\boldsymbol{W}_k. \tag{6}$$

Defining the vector $\boldsymbol{P}$ as the cross correlation between the desired response (a scalar) and the $\boldsymbol{X}$-vector then yields

$$\boldsymbol{P}^T \triangleq E[d_j\boldsymbol{X}_j^T] = E[d_jx_{1j}\; d_jx_{2j} \cdots d_jx_{nj}]^T. \tag{7}$$

The input correlation matrix $\boldsymbol{R}$ is defined in terms of the ensemble average

$$\boldsymbol{R} \triangleq E[\boldsymbol{X}_j\boldsymbol{X}_j^T] = E\begin{bmatrix} x_{1j}x_{1j} & x_{1j}x_{2j} & \cdots & \\ x_{2j}x_{1j} & x_{2j}x_{2j} & \cdots & \\ \vdots & \vdots & & \\ & & \cdots & x_{nj}x_{nj} \end{bmatrix}. \tag{8}$$

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

$$\xi_k \triangleq E[\varepsilon_j^2]_{\boldsymbol{W}=\boldsymbol{W}_k} = E[d_j^2] - 2\boldsymbol{P}^T\boldsymbol{W}_k + \boldsymbol{W}_k^T\boldsymbol{R}\boldsymbol{W}_k. \tag{9}$$

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 $\boldsymbol{W} = \boldsymbol{W}_k$ is obtained by differentiating (9):

$$\nabla_k \triangleq \left\{\begin{matrix} \dfrac{\partial E[\varepsilon_j^2]}{\partial w_1} \\ \vdots \\ \dfrac{\partial E[\varepsilon_j^2]}{\partial w_n} \end{matrix}\right\}_{\boldsymbol{W}=\boldsymbol{W}_k} = -2\boldsymbol{P} + 2\boldsymbol{R}\boldsymbol{W}_k. \tag{10}$$

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

$$\boldsymbol{W}^* = \boldsymbol{R}^{-1}\boldsymbol{P}. \tag{11}$$

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:

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

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

$$\xi_k = \xi_{\min} + \boldsymbol{V}_k^T\boldsymbol{R}\boldsymbol{V}_k \tag{13}$$

where

$$\boldsymbol{V}_k \triangleq \boldsymbol{W}_k - \boldsymbol{W}^*. \tag{14}$$

The gradient may be expressed in terms of $\boldsymbol{V}_k$ as

$$\nabla_k = 2\boldsymbol{R}\boldsymbol{V}_k. \tag{15}$$

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

$$\boldsymbol{R} = \boldsymbol{Q}\Lambda\boldsymbol{Q}^{-1} \tag{16}$$

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

$$\boldsymbol{Q}^{-1} = \boldsymbol{Q}^T. \tag{17}$$

Note further that the inverse of $\boldsymbol{R}$ is

$$\boldsymbol{R}^{-1} = \boldsymbol{Q}\Lambda^{-1}\boldsymbol{Q}^{-1}. \tag{18}$$

The mean square error may thus be expressed as

$$\xi_k = \xi_{\min} + \boldsymbol{V}_k^T\boldsymbol{Q}\Lambda\boldsymbol{Q}^T\boldsymbol{V}_k. \tag{19}$$

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

$$\boldsymbol{V}' = \boldsymbol{Q}^T\boldsymbol{V} = \boldsymbol{Q}^{-1}\boldsymbol{V} \tag{20}$$

and

$$\boldsymbol{V}'^T = \boldsymbol{V}^T\boldsymbol{Q}. \tag{21}$$

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

$$\xi_k = \xi_{\min} + \boldsymbol{V}_k'^T\Lambda\boldsymbol{V}_k'. \tag{22}$$

The transformation $\boldsymbol{Q}$ projects $\boldsymbol{V}$ into $\boldsymbol{V}'$—that is, projects $\boldsymbol{V}$ 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

$$\nabla_k' = 2\Lambda\boldsymbol{V}_k'. \tag{23}$$

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 $\boldsymbol{R}$ and $\boldsymbol{P}$. Furthermore, this process would have to be continually repeated in most circumstances, where the input

[2] This can always be done when $\boldsymbol{R}$ 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.

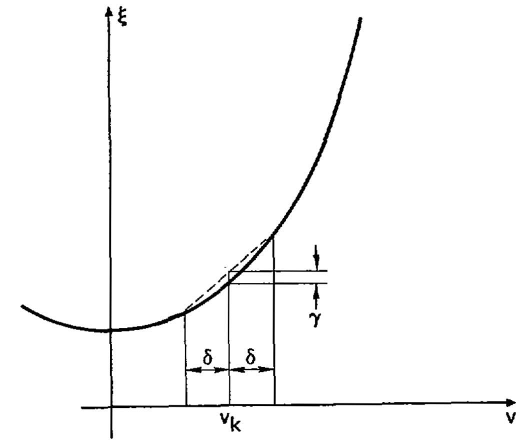


Fig. 2. Gradient estimation by derivative measurement.

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

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

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 $\boldsymbol{R}$-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

$$P = \frac{\delta^2 \lambda_{\text{av}}}{\xi_{\min}}. \tag{56}$$

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:

$$\hat{\xi} = \frac{1}{N} \sum_{j=1}^{N} \varepsilon_j^2. \tag{57}$$

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

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

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

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

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

$$\operatorname{var}[\hat{\xi}] = \frac{\kappa \xi^2}{N} \tag{60}$$

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

$$\operatorname{var}[\hat{\xi}] = \frac{2\xi^2}{N}. \tag{61}$$

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

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

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 $\hat{\nabla}_k$:

$$\hat{\nabla}_k \triangleq \nabla_k + N_k. \tag{63}$$

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

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

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

$$N_k' = Q^{-1} N_k. \tag{65}$$

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':

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

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

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

Substituting (15) and combining terms yields

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

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}:

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

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]

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

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

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

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):

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

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

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

Thus (77) can be rewritten

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

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

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

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

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:

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

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

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

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:

$$\boldsymbol{N}_j = \hat{\boldsymbol{\nabla}}_j = -2\varepsilon_j \boldsymbol{X}_j. \tag{100}$$

The covariance of this noise is given by

$$\operatorname{cov}[\boldsymbol{N}_j] = E[\boldsymbol{N}_j\boldsymbol{N}_j^T] = 4E[\varepsilon_j^2\boldsymbol{X}_j\boldsymbol{X}_j^T]. \tag{101}$$

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

$$\operatorname{cov}[\boldsymbol{N}_j] = 4E[\varepsilon_j^2]E[\boldsymbol{X}_j\boldsymbol{X}_j^T] = 4\xi_{\min}\boldsymbol{R}. \tag{102}$$

In primed coordinates the covariance is

$$\operatorname{cov}[\boldsymbol{N}_j'] = \boldsymbol{Q}^{-1}\operatorname{cov}[\boldsymbol{N}_j]\boldsymbol{Q} = 4\xi_{\min}\boldsymbol{\Lambda}. \tag{103}$$

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 $\boldsymbol{N}_j'$—that is, to both the DSD algorithm and the LMS algorithm. For the LMS algorithm, using (74) and (103), one may write

$$\operatorname{cov}[\boldsymbol{V}_j'] = \frac{\mu}{4}\boldsymbol{\Lambda}^{-1}(4\xi_{\min}\boldsymbol{\Lambda}) = \mu\xi_{\min}\boldsymbol{I}. \tag{104}$$

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

$$\operatorname{cov}[\boldsymbol{V}_j] = \mu\xi_{\min}\boldsymbol{I}. \tag{105}$$

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

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

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

$$M = \frac{E[\boldsymbol{V}_j'^T\boldsymbol{\Lambda}\boldsymbol{V}_j']}{\xi_{\min}} = \mu\operatorname{tr}\boldsymbol{R}. \tag{107}$$

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

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

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 $\boldsymbol{R}$. 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 $\boldsymbol{R}$ 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:

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

This formula is also a good approximation in many cases when the eigenvalues of $\boldsymbol{R}$ 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:

$$\mathbf{W}_{k+1} = \mathbf{W}_k + \tfrac{1}{2}[1 + \operatorname{sgn}\{\hat{\xi}(\mathbf{W}_k) - \hat{\xi}(\mathbf{W}_k + \mathbf{U}_k)\}]\mathbf{U}_k \tag{113}$$

where $\mathbf{U}_k$ is a random vector; $\hat{\xi}(\mathbf{W}_k)$ is an estimate of mean square error based on N samples of ε_j with $\mathbf{W} = \mathbf{W}_k$; $\hat{\xi}(\mathbf{W}_k + \mathbf{U}_k)$ is an estimate of mean square error based on N samples of ε_j with $\mathbf{W} = \mathbf{W}_k + \mathbf{U}_k$; 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 $\mathbf{U}_k$ 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:

$$\mathbf{W}_{k+1} = \mathbf{W}_k + \beta[\hat{\xi}(\mathbf{W}_k) - \hat{\xi}(\mathbf{W}_k + \mathbf{U}_k)]\mathbf{U}_k \tag{114}$$

where $\mathbf{U}_k$ is a random vector from a random vector generator designed to have a covariance of $\sigma^2\mathbf{I}$; $\hat{\xi}(\mathbf{W}_k)$ and $\hat{\xi}(\mathbf{W}_k + \mathbf{U}_k)$ 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 $\mathbf{U}_k$ to $\mathbf{W}_k$ is given by

$$(\Delta\xi)_k \triangleq \xi(\mathbf{W}_k + \mathbf{U}_k) - \xi(\mathbf{W}_k). \tag{115}$$

The corresponding estimated change in mean square error is

$$(\widehat{\Delta\xi})_k \triangleq \hat{\xi}(\mathbf{W}_k + \mathbf{U}_k) - \hat{\xi}(\mathbf{W}_k). \tag{116}$$

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

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

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 $\hat{\xi}(\mathbf{W}_k)$, with the weight vector set at its nominal value, and N samples to obtain $\hat{\xi}(\mathbf{W}_k + \mathbf{U}_k)$. 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

$$\begin{aligned}E\left[\xi(\mathbf{W}_k) - \frac{\xi(\mathbf{W}_k) + \xi(\mathbf{W}_k + \mathbf{U}_k)}{2}\right]\\ = \tfrac{1}{2}E[\xi(\mathbf{W}_k) - \xi(\mathbf{W}_k + \mathbf{U}_k)].\end{aligned} \tag{120}$$

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

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

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

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

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

$$\mathbf{W}_{k+1} = \mathbf{W}_k + \beta[-(\Delta\xi)_k + \zeta_k]\mathbf{U}_k \tag{123}$$

or

$$\mathbf{V}_{k+1} = \mathbf{V}_k + \beta[-(\Delta\xi)_k + \zeta_k]\mathbf{U}_k. \tag{124}$$

If one lets σ^2 be small by design, so that $\mathbf{U}_k$ is always small, one can write

$$(\Delta\xi)_k = \mathbf{U}_k^T\nabla_k = 2\mathbf{U}_k^T\mathbf{R}\mathbf{V}_k. \tag{125}$$

Substituting (125) into (124) then yields

$$\begin{aligned}\mathbf{V}_{k+1} &= \mathbf{V}_k + \beta\mathbf{U}_k[-2\mathbf{U}_k^T\mathbf{R}\mathbf{V}_k + \zeta_k]\\ &= (\mathbf{I} - 2\beta\mathbf{U}_k\mathbf{U}_k^T\mathbf{R})\mathbf{V}_k + \beta\zeta_k\mathbf{U}_k.\end{aligned} \tag{126}$$

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:

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

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

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

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

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

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

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

and

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

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. Equation (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:

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

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

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

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

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

and

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

Inserting (148) into (145) yields

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

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

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

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

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

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, $T_{p_{mse}}$	$\frac{Nn}{2\mu\lambda_p}$	$\frac{1}{4\mu\lambda_p}$	$\frac{N}{2\sigma^2\beta\lambda_p}$

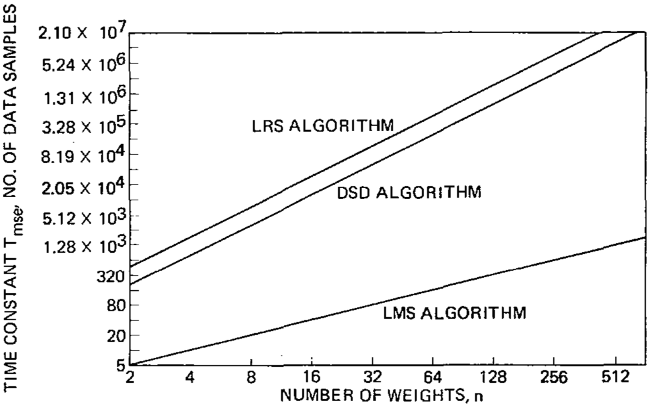


Fig. 3. Time constant of adaptive process as function of number of weights with total misadjustment M_{tot} fixed at 10 percent (perturbation P optimized for DSD and LRS algorithms).

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

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

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:

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

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

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

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

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

[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

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

This is a much more 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

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

which yields a value of the time constant of

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

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 $\xi_{\min}$ 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.

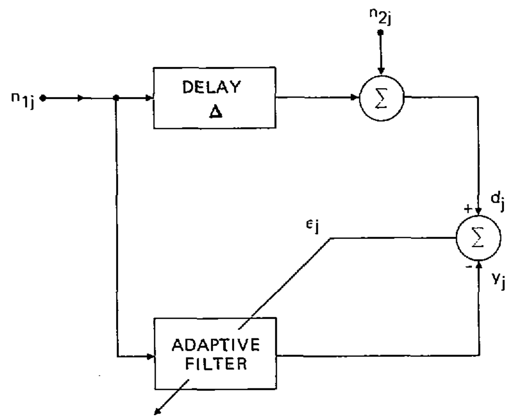


Fig. 4. Modeling a fixed delay with an adaptive filter.

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 $\xi_{\min}$ 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 $\xi_{\min}$ the difference between the average value of asymptotic mean square error and $\xi_{\min}$.

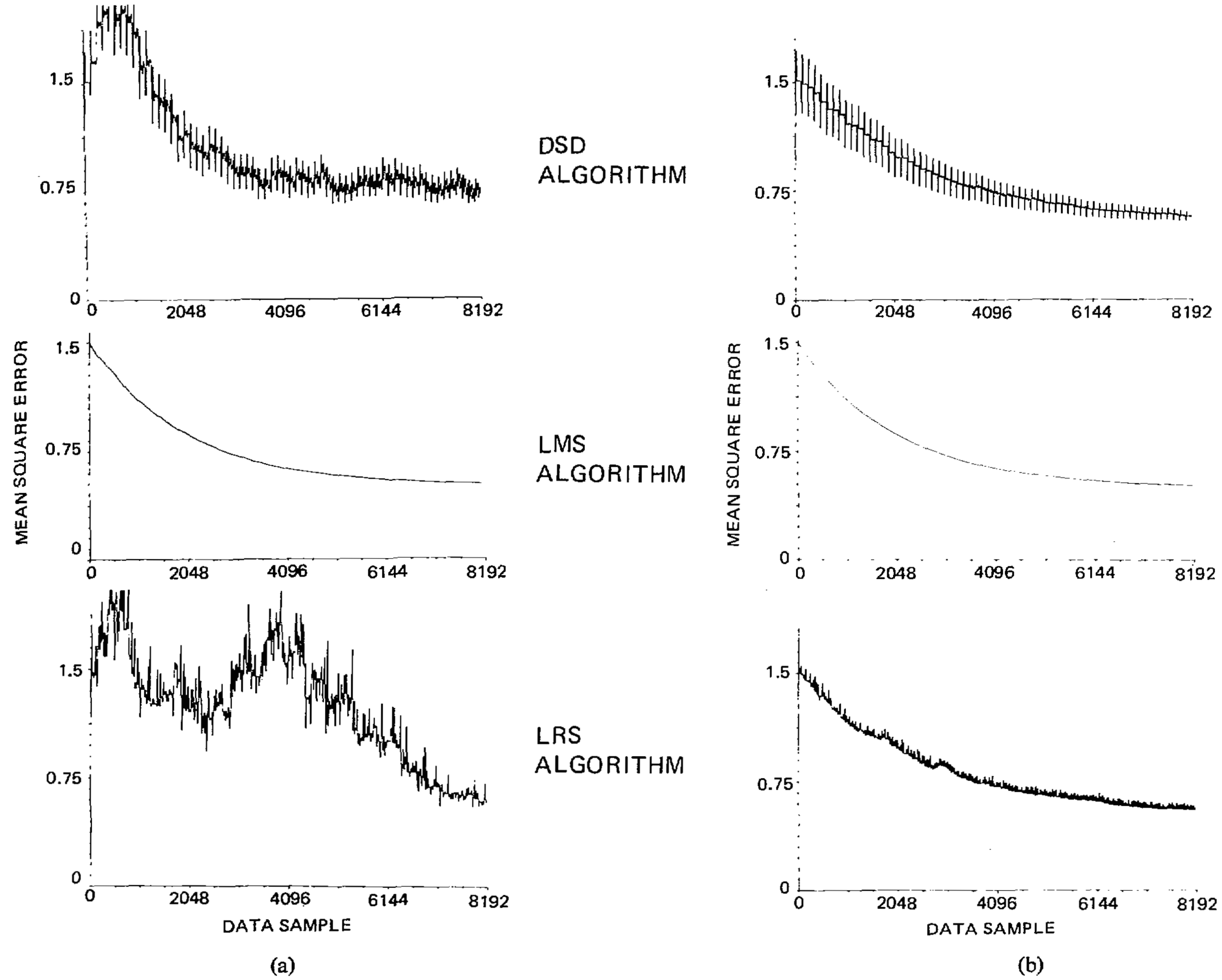


Fig. 5. Results of fixed delay modeling experiment with theoretical time constant T_{mse} fixed at 2048 data samples. (a) Individual learning curves. (b) Ensemble averages of 32 learning curves.

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

	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	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

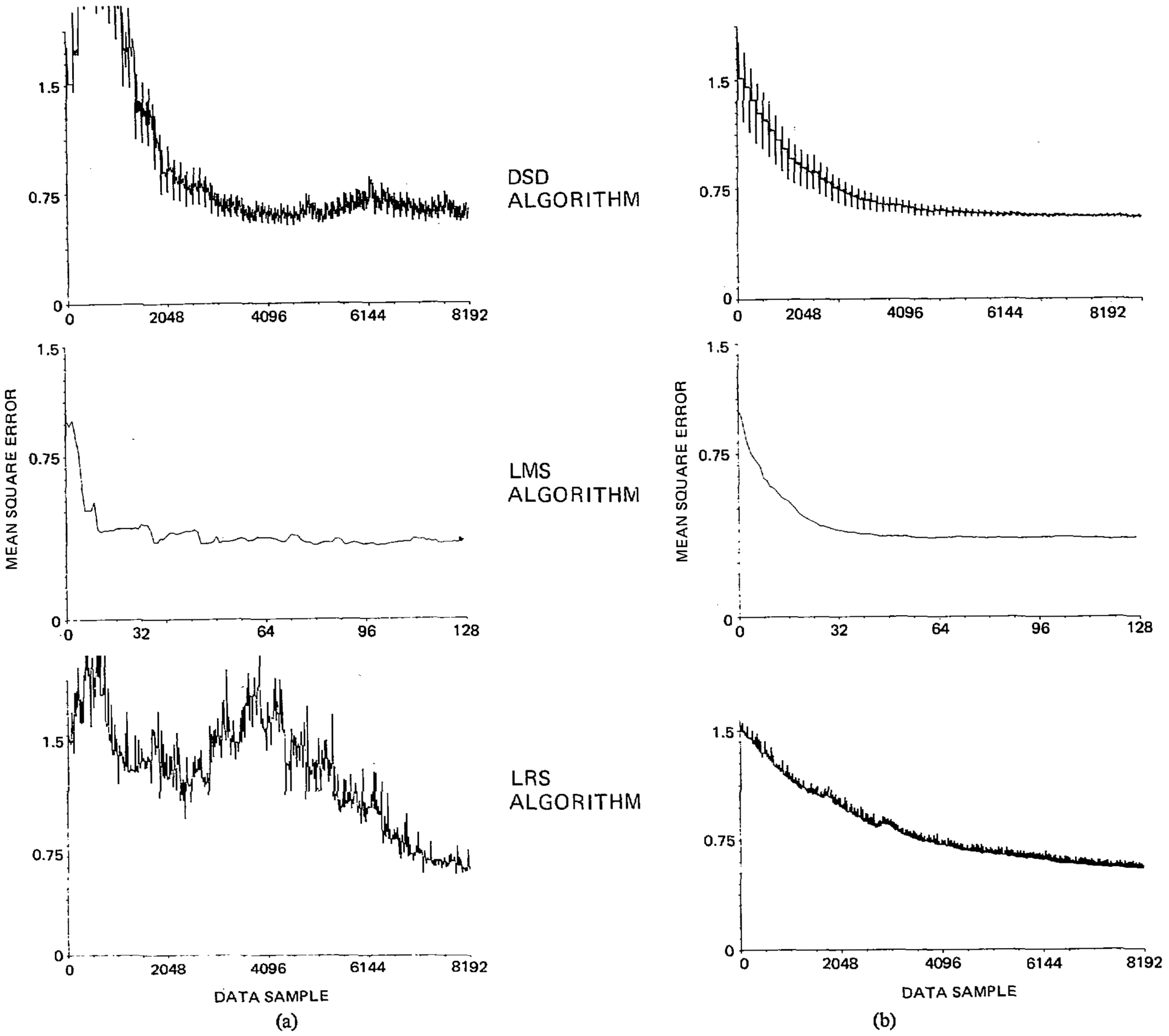


Fig. 6. Results of fixed delay modeling experiment with theoretical total misadjustment M_{tot} fixed at 9.375 percent. (a) Individual learning curves. (b) Ensemble averages of 32 learning curves.

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

								Total		
				Perturbation P,		Misadjustment M,		misadjustment M_{tot},		Theoretical time
	Convergence constants			percent		percent		percent		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.

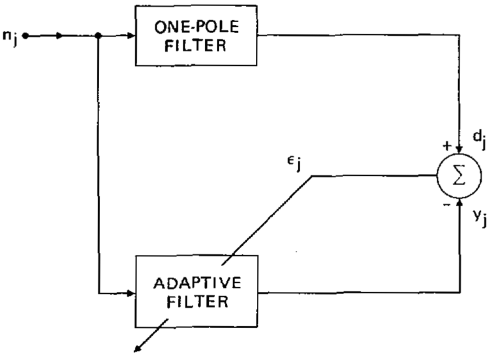


Fig. 7. Modeling a one-pole recursive filter with an adaptive filter.

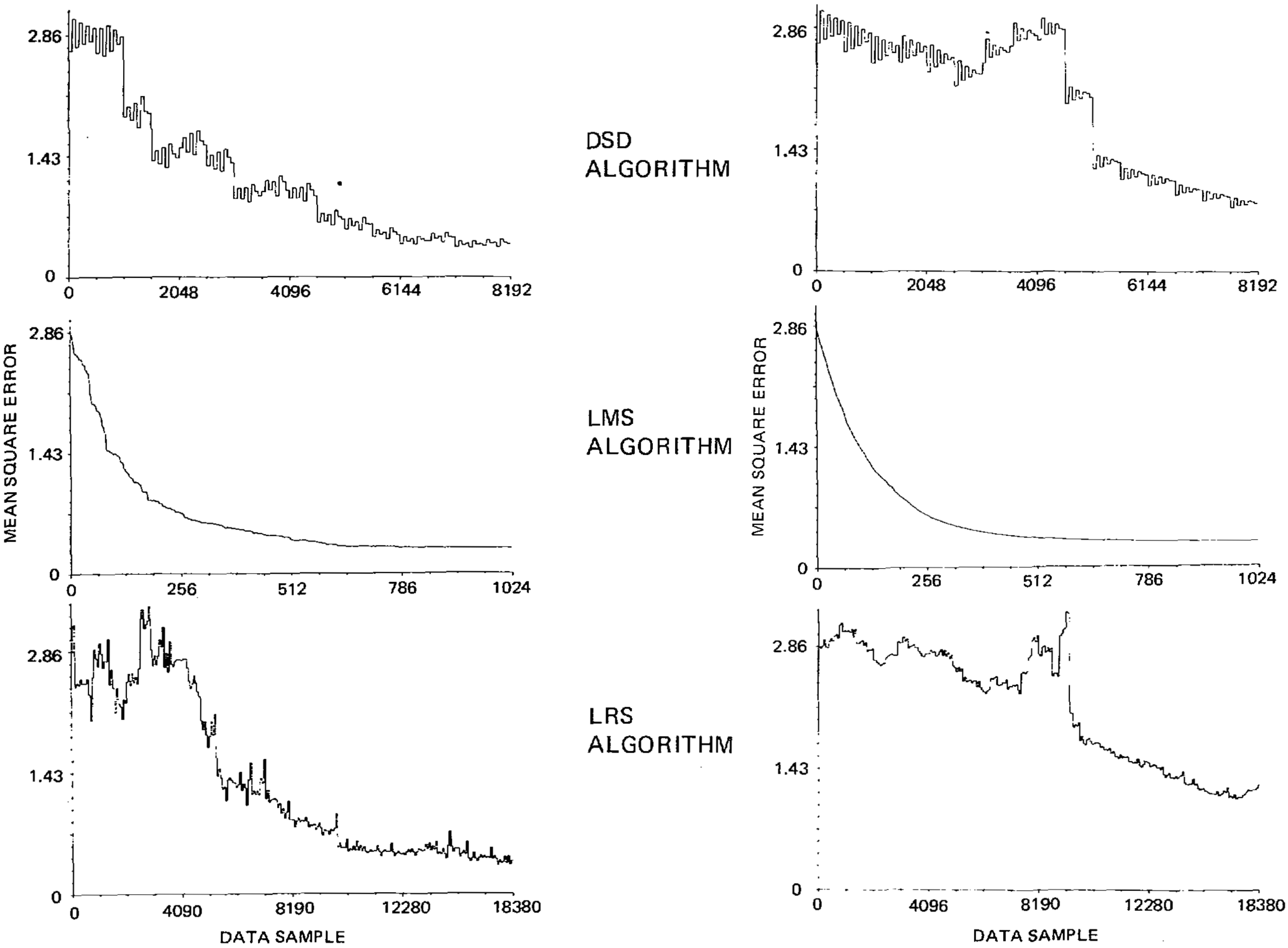


Fig. 8. Results of one-pole filter modeling experiment with theoretical total misadjustment M_{tot} fixed at 7.5 percent for DSD and LRS algorithms and at 0.75 percent for LMS algorithm. (a) Individual learning curves. (b) Ensemble averages of 32 learning curves.

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 ment 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

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	2.8675	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.

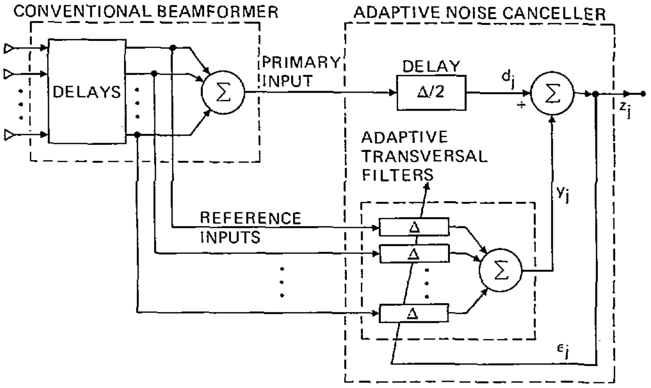


Fig. 9. Block diagram of null-constrained adaptive beamformer tolerant of array element gain and phase errors.

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 Δ/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)

$$\begin{bmatrix} w \cdots w\ 0\ 0\ 0\ 0\ 0\ w \cdots w \\ w \cdots w\ w\ 0\ 0\ 0\ w\ w \cdots w \\ \cdot \qquad 0 \qquad \cdot \\ \cdot \qquad\qquad \cdot \\ \cdot \qquad 0\ 0\ 0 \qquad \cdot \\ w \cdots w\ 0\ 0\ 0\ 0\ 0\ w \cdots w \end{bmatrix}$$

(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.

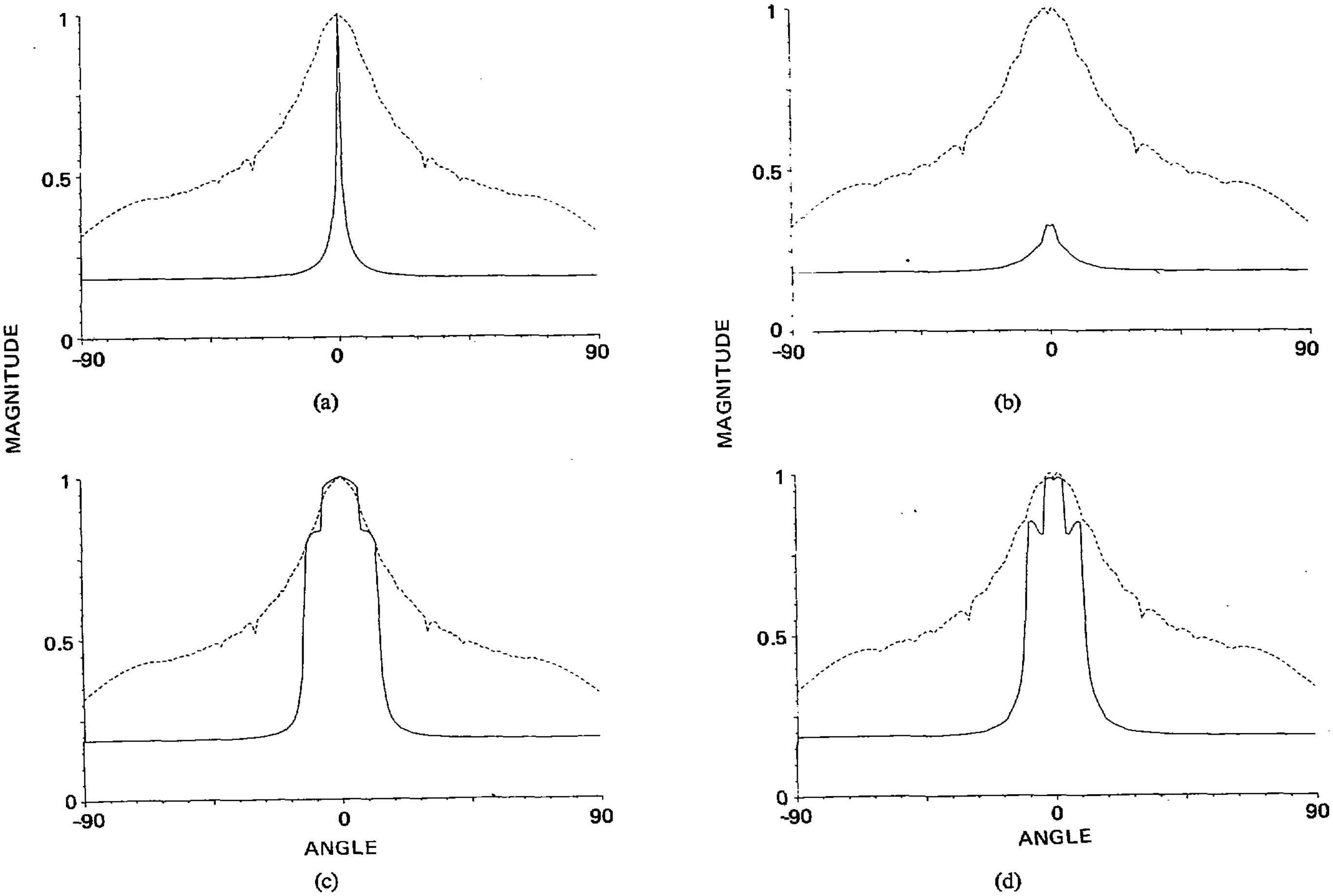


Fig. 11. Results of adaptive beamforming experiment. (a) Single column-of-zeros constraint, ideal array. (b) Single column-of-zeros constraint, nonideal array. (c) Triple column-of-zeros constraint, ideal array. (d) Triple column-of-zeros constraint, nonideal array.

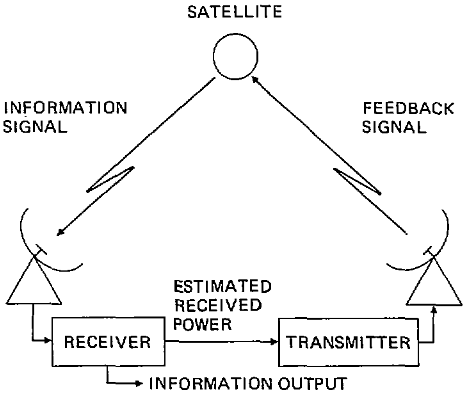


Fig. 12. Satellite transmitting information to receiver on earth.

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

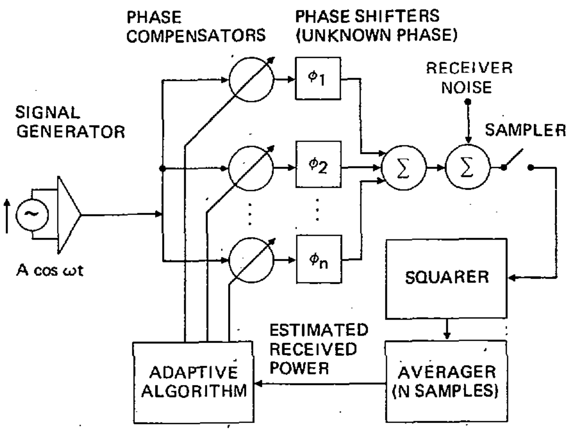
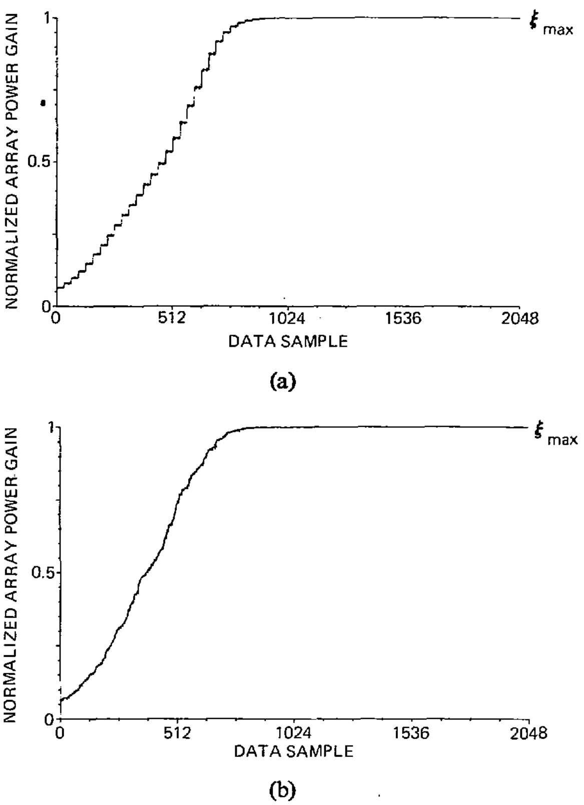


Fig. 13. Digital simulation of adaptive transmitting antenna.

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.



(a)

(b)

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

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 "$\boldsymbol{R}$-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. 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.

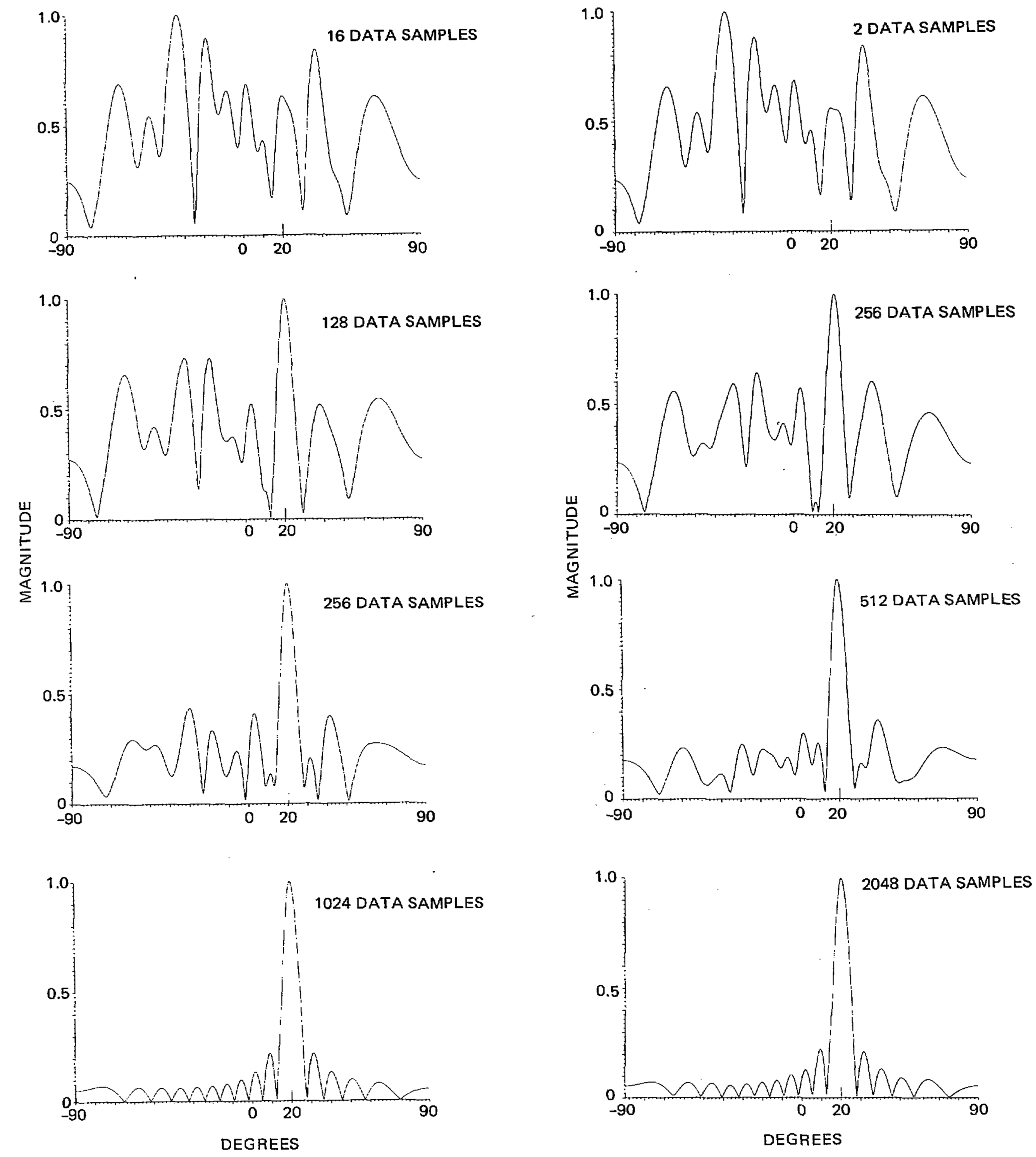
Fig. 16 shows learning curves of the adaptive process when independent samples of white noise with a power of 0.01 were injected into the simulated received signal. Array configuration and adaptive parameters are the same as in the noiseless case represented by Fig. 14. As well as can be determined by 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 "$\boldsymbol{R}$-matrix" of the performance surface can be shown to be

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

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.



(a)

(b)

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 $\xi_{\max}$. 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.

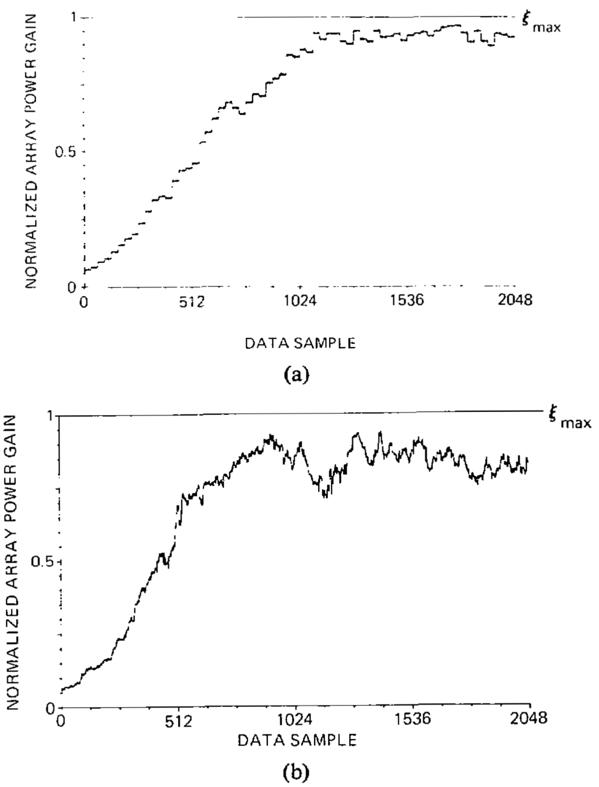


Fig. 16. Learning curves of simulated adaptive transmitting antenna with noise. (a) DSD algorithm. (b) LRS algorithm.

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. Daniell, "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.