16

APPLICATION OF CONJUGATE GRADIENT METHOD FOR OPTIMUM ARRAY PROCESSING

S. Choi

16.1 Introduction

The objective is to find an optimal set of weights for each element of an array for enhancing the reception of the desired random signal that is processed in a signal environment containing numerous interfering components and thermal noises. In section 16.2, a procedure of formulating a communication problem or electromagnetic wave interaction as a matrix equation of the form, $\mathbf{A}\,\mathbf{W} = \mathbf{Y}$, is shown where $\mathbf{A}$ is a known matrix and $\mathbf{W}$ is the weight vector to be computed for a known desired vector **Y**.* It will be shown that the characteristic of the matrix **A** is determined by the nature of input signals and system environments. A theoretical background for the adaptive signal processing is briefly reviewed in section 16.3. Procedures of refining the weight vector **W** in conventional methods such as the LMS algorithm and the method of steepest descent are shown in section 16.3. A design of an adaptive linear predictor is simulated in Example 1 of this section. Starting from the basic philosophy of iteration methods in section 16.3, adaptive procedures of the conjugate gradient method are derived in section 16.4 for many different fields of applications. The performance measure will be determined properly depending upon the matrix **A**, i.e., whether or not it is real, whether or not positive semidefinite, and whether or not symmetric (or Hermitian for the case of complex matrix). The linear predictor is redesigned using the conjugate gradient method in Example 2 and the performances are compared to the case of the LMS method and the method of steepest descent. In Example 3, the conjugate gradient method is applied to the design of an adaptive echo canceller for a telephone system. For the case when the matrix is arbitrary, a generalized conjugate gradient method is introduced. The generalized conjugate gradient method is applied to the design of an adaptive array for multipath telecommunications in Example 4.

16.2 Formulation of the Problem

Consider an array of Fig. 16.1 which consists of N isotropic antennas equally spaced with a separation d. A plane wave is incident upon the array with an incident angle ϕ, where ϕ is the arrival angle from the broadside of the array as shown in Fig. 16.1. There exists a difference in time of arrival between the adjacent elements as follows:

$$\tau = (d/c)\sin(\phi) \tag{1}$$

where c is the speed of the wave. From (1), one can find that the signal induced at each antenna element can be expressed as a delayed version of the signal induced at the reference element. Thus, the array output $y(t_0)$ at time t_0 can be represented as

$$y(t_0) = \sum_{i=0}^{N-1} x\,(t_0 - i\tau)\; w_i \tag{2}$$

* In this text, normal letter denotes a scalar quantity whereas bold letter denotes a matrix or vector quantity.

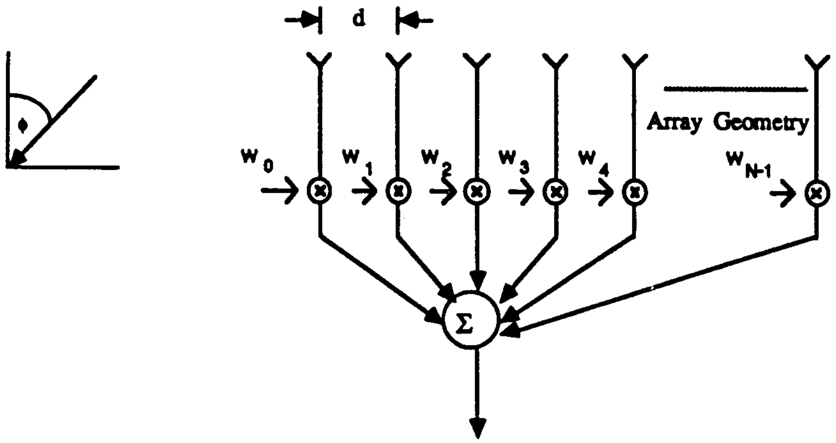


Figure 16.1 Antenna array with N elements.

where $x(t_0)$ is the signal induced at the reference antenna element at time t_0, N is the number of array elements, τ is given in (1),and w_i is the weight of the ith element.

From (1) and (2), the estimation problem can be formulated as follows:

Determine the value of w_i for each element, $i = 0, 1, \ldots, N-1$, in such a way that the array output $y(t_0)$ at t_0 best approximates the desired signal $d(t_0)$.

Note that the observation time interval can be set to be the spatial time interval τ without loss of generality. If the observation time interval is equal to τ, then the estimation problem can be formulated much more simply as shown in (3). In (3), i and k have been used as indices for the spatial time interval and observation time interval, respectively.

$$\sum_{i=0}^{N-1} x_{i+k} w_i = y_k \tag{3}$$

where y_k denotes the estimation at time k for each observation time, $k = 0, 1, 2, \ldots, M-1$. Note that the estimation y_k may not be same as the desired value. More details about the error between the estimation and the desired value will be discussed in section 16.3 and

16.4. If the number of observation times is set to be the number of array elements N, we can represent (3) as an $N \times N$ matrix equation with N unknowns as follows:*

$$\mathbf{A}\,\mathbf{W} = \mathbf{Y} \tag{4}$$

where $\mathbf{A}$ denotes the signal matrix and $\mathbf{W}$ denotes the weight vector for a desired signal vector $\mathbf{Y}$. Note that the element of the matrix $\mathbf{A}$ at the kth column of the ith row is the input signal x_{i+k-2}. $\mathbf{A}, \mathbf{W}$, and $\mathbf{Y}$ are shown in (5), (6), and (7) as

$$\mathbf{A} = \begin{bmatrix} x_0 & x_1 & \cdots & x_{N-1} \\ x_1 & x_2 & \cdots & x_N \\ \vdots & \vdots & \ddots & \vdots \\ x_{N-1} & x_N & \cdots & x_{2N-2} \end{bmatrix} \tag{5}$$

$$\mathbf{W}^t = [w_0, w_1, \ldots, w_{N-1}] \tag{6}$$

$$\mathbf{Y}^t = [y_o, y_1, \ldots, y_{N-1}] \tag{7}$$

where the superscript t denotes the transpose operator.

It is well known that there exists a unique set of solutions for a matrix equation only when the rank of the matrix is equal to the number of unknowns. For the matrix equation shown in (4), however, the number of antenna elements cannot always be the same as the rank of the matrix $\mathbf{A}$ because the number of antenna elements is a fixed value in an array whereas the rank of the matrix $\mathbf{A}$ is determined by the number of frequency components of the signal and various random noises. Therefore, the solution of (4) is in general not unique. When the rank of the matrix is less than the number of antenna elements, the solution does not even exist. In practice, however, this will not occur because of the interference and noise components. This means that the actual problem setting is in an infinite dimensional space, which in simple terms means that we have an infinite number of unknowns to be solved. Since the matrix $\mathbf{A}$ is in general formed as an ill-conditioned matrix as discussed above, an iterative method rather than a direct method such as the Gaussian elimination must be used for solving the matrix equation given in (4).

* Here we are trying to simplify the problem by forming the square matrix. In general, the matrix $\mathbf{A}$ does not have to be a square matrix.

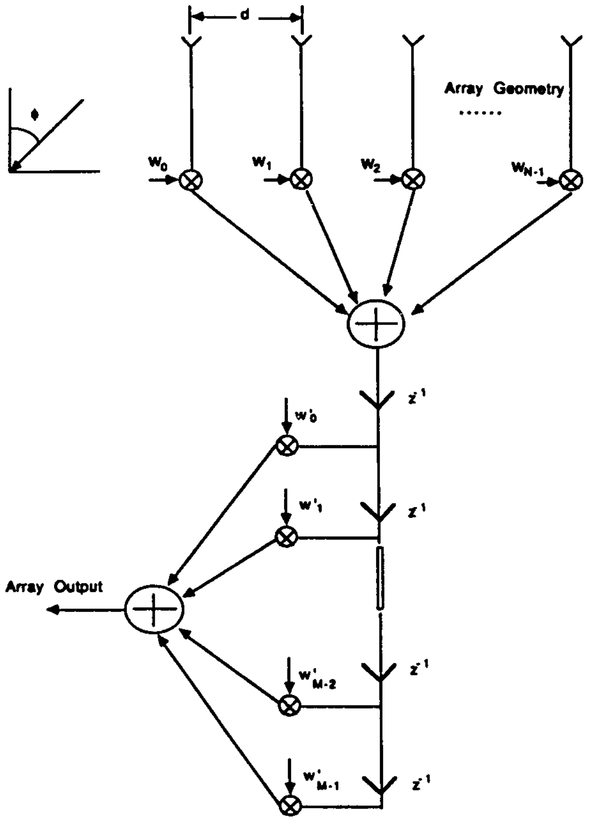


Figure 16.2 Antenna array with N elements in the detection part and M elements in the estimation part. z^{-1} in this figure corresponds to the delay τ in (1).

In (3), it is important to note that the induced signal, x_{i+k}, for each element, i.e., for $i = 0, 1, \ldots, N-1$, at each observation time, i.e., at $k = 0, 1, \ldots, N-1$, must be reasonably reliable to produce a meaningful estimation y_k. This means that the left-hand-side of (3) must not include large interference and/or noise components for a good estimation. Therefore, the matrix $\mathbf{A}$ in (4) must be formed with the output signals of a detecting network which employs properly selected weights as its gains to resolve the undesired signal components. More details about detection problems for resolving interference signals and noises are discussed in [3]. This is illustrated in Fig. 16.2.

16.3 Theoretical Background for Adaptive Signal Processing

This section surveys conventional adaptive algorithms to help readers understand the basic philosophy of the term, "adaptation." When an adaptive method is used for solving a matrix equation of the form $\mathbf{A}\,\mathbf{W} = \mathbf{Y}$, starting from an arbitrary initial guess $\mathbf{W}_0$ we refine the weight vector $\mathbf{W}$ by

$$\mathbf{W}_{n+1} = \mathbf{W}_n + t_n \mathbf{P}_n \tag{8}$$

where the index n denotes the iteration number, t_n is a scalar quantity related to the step size*, and $\mathbf{P}_n$ is a vector which determines the direction of search. Therefore, there always arise two questions to be answered whenever an adaptive method is used for solving a matrix equation: one is about the value for the step size, t_n, and the other is about the value for the search direction $\mathbf{P}_n$. Therefore, an adaptive algorithm is characterized depending on the way of setting the step size t_n and the search directions $\mathbf{P}_n$.

In this section, we review the classical method of setting t_n as a fixed value. When a fixed value is to be selected for all through the iterations, a special care has to be taken that the value for the step size ensure the convergence of the adaptive procedure. The boundary of the value for t_n which guarantees the convergence will be derived and discussed in the third article of this section. On the other hand, we can also try to optimize the value for the step size at each iteration. When the method of variable step size is used, we must select a proper functional which is to be minimized with respect to the step size, t_n, at each iteration. If the amount of additional computation required for optimizing t_n at each iteration does not exceed the speed-up due to the optimization, then the method of variable step size is preferred. It has been our experience that the speed-up obtained by the optimization of the step size becomes larger as the size (or, for the same size, the condition number) of the matrix becomes larger.

Regarding the value for the search direction $\mathbf{P}_n$ our explanations will be based on the idea of the steepest descent method in which the direction of search is the negative of the gradient of the functional.

* t_n is often called adaptive gain in other texts. In this text, t_n will be denoted as the step size

Utilizing the philosophy of the mean squared error(MSE) criterion** for the performance measure, the squared error, i.e., $\|\mathbf{A}\,\mathbf{W}-\mathbf{Y}\|^2$ can be selected as the functional. Or, alternatively, a functional can be generated in such a way that the gradient of the functional becomes the error itself, i.e., $\mathbf{A}\,\mathbf{W}-\mathbf{Y}$, so that we are actually solving the matrix equation $\mathbf{A}\,\mathbf{W}=\mathbf{Y}$ when we minimize the functional. More details about setting a functional will be discussed in Section 16.4.

16.3.1 The Gradient and the Wiener Solution

From (3), the estimation at k th observation time can be represented in a vector notation as follows:

$$y_k = \mathbf{W}^t\mathbf{X}_k \tag{9}$$

for each observation time $k = 0, 1, \ldots, N-1$ where $\mathbf{X}_k$ denotes the input vector at kth observation time. Once again, the superscript t denotes the transpose operator. If the desired value at an observation time k is denoted by d_k, the error signal e_k can be obtained as

$$e_k = y_k - d_k = \mathbf{W}^t\mathbf{X}_k - d_k \tag{10}$$

The square of this error is

$$e_k^2 = d_k^2 - 2d_k\mathbf{W}^t\mathbf{X}_k + \mathbf{W}^t\mathbf{X}_k\mathbf{X}_k^t\mathbf{W} \tag{11}$$

Therefore the expectation of the squared error is given by

$$E\{e_k^2\} = E\{d_k^2\} - 2\boldsymbol{\phi}_k(x,d)^t\,\mathbf{W} + \mathbf{W}^t\,\boldsymbol{\phi}_k(x,x)\mathbf{W} \tag{12}$$

where $E\{\ \}$ denotes the statistical expectation operator, $\boldsymbol{\phi}_k(x,d)$ is defined as a column vector of the cross-correlations between the

** Besides the MSE criterion, there exists various performance[1] measures which govern the operation of the adaptive procedure such as SNR (Signal to Noise Ratio) criterion, ML (Maximum Likelihood) criterion, and MV (Minimum Variance) criterion. However, for using SNR, ML, or MV criterion, we must assume the availability of the statistics of both signals and noises. In practice, these values may not be available. Another reason to avoid using those criteria is that we concentrate on solving the matrix equation rather than treating the signals and noises in a statistical manner.

input vector $\mathbf{X}_k$ and desired signal d_k, and $\phi_k(x,x)$ is defined as an $N \times N$ matrix of the autocorrelations of the input vector $\mathbf{X}_k$. Thus, $\phi_k(x,d) = E\{d_k\mathbf{X}_k\}$ and $\phi_k(x,x) = E\{\mathbf{X}_k\mathbf{X}_k^t\}$.***

From (12), the gradient of the MSE can be computed as

$$\text{Grad. } \left[E\left\{e_k^2\right\}\right] = -2\phi(x,d) + 2\phi(x,x)\mathbf{W} \tag{13}$$

where Grad. denotes the gradient operator with respect to $\mathbf{W}$. Therefore, the optimum choice for the weights which minimizes the MSE, i.e., $E\{e_k^2\}$, can be written as follows:

$$\mathbf{W}_{\text{opt}} = \phi(x,x)^{-1}\,\phi(x,d) \tag{14}$$

The expression for $\mathbf{W}_{\text{opt}}$ given in (14) is the Wiener-Hopf equation in a matrix form and is consequently referred to as the "Optimum Wiener Solution". Note that the autocorrelation matrix $\phi(x,x)$ must be invertible for (14) to be valid.

16.3.2 The Method of Steepest Descent

Starting from an arbitrary guess for the initial weights $\mathbf{W}$, we obtain a refined value by making a change in the direction of the negative of the gradient vector of the functional. This means that the direction vector $\mathbf{P}_n$ in the method of steepest descent is simply the negative of the gradient of the functional. The method of steepest descent can be expressed as

$$\mathbf{W}_{n+1} = \mathbf{W}_n - \mu\,\text{Grad. } \left[E\left\{e_n^2\right\}\right] \tag{15}$$

where the index n denotes the iteration number and μ is a positive constant which is related to the step size as t_n in (8). Note that the step size is a fixed value for all through the iterations. Also note that the functional employed in the method of steepest descent is the expected value of squared error, i.e., $E\left\{e_n^2\right\}$.

Substituting (13) in (15), we can rewrite (15) as

$$\mathbf{W}_{n+1} = \mathbf{W}_n - \mu\left(-2\phi(x,d) + 2\phi(x,x)\mathbf{W}_n\right) \tag{16}$$

*** In this text, the signals are assumed to be stationary unless mentioned otherwise. Therefore, the subscript, k, of $\phi_k(x,x)$ and $\phi_k(x,d)$ will be deleted unless the signals are nonstationary.

Here, the magnitude of μ must be less than $1/\lambda_{\mathrm{MAX}}$ for the convergence of the procedure where λ_{MAX} denotes the maximum eigenvalue of the autocorrelation matrix $\phi(x,x)$.[14] The restriction on the boundary of μ can be explained as follows:

Using (16), we can represent the weights vector at each iteration as

$$\begin{aligned} \mathbf{W}_1 &= [I - 2\mu\phi(x,x)]\,\mathbf{W}_0 + 2\mu\phi(x,d) \\ \mathbf{W}_2 &= [I - 2\mu\phi(x,x)]\,\mathbf{W}_1 + 2\mu\phi(x,d) \\ &\cdots \\ \mathbf{W}_N &= [I - 2\mu\phi(x,x)]\,\mathbf{W}_{N-1} + 2\mu\phi(x,d) \end{aligned}$$

where I denotes the $N \times N$ identity matrix. Then, $\mathbf{W}_N$ can be represented as a power series of the initial guess $\mathbf{W}_0$ as

$$\mathbf{W}_N = [\mathbf{I} - 2\mu\,\phi(x,x)]^N\,\mathbf{W}_0 + 2\mu\sum_{i=0}^{N-1}[\mathbf{I} - 2\mu\,\phi(x,x)]^i\,\phi(x,d) \tag{17}$$

For the power series of (17) to be convergent, the magnitude of each element in $[\mathbf{I} - 2\mu\phi(x,x)]$, must be less than unity, i.e.,

$$|\mathbf{I} - 2\mu\phi(x,x)| < 1 \tag{18}$$

which can be satisfied by the restriction, $0 < \mu < 1/\lambda_{\mathrm{MAX}}$ where λ_{MAX} denotes the maximum eigenvalue of the autocorrelation matrix $\phi(x,x)$.

16.3.3 The Least Mean Square (LMS) Algorithm

As shown in (16), one can use the method of steepest descent only when the crosscorrelation vector $\phi(x,d)$ and autocorrelation matrix $\phi(x,x)$ are perfectly known. In many cases, however, the signal statistics may not be known apriori. When the signal statistics are not available, the exact values for the gradient of the performance surface cannot be computed. Widrow[4] has shown that the LMS algorithm can be used to find the optimal set of weights in an adaptive way when the gradient must itself be estimated. The LMS algorithm updates the values for the weights as follows:

$$\mathbf{W}_{n+1} = \mathbf{W}_n - \mu\mathrm{Grad}_{\cdot\mathrm{est}}\left[E\left\{e_n^2\right\}\right] \tag{19}$$

where $\text{Grad.}_{est}\left[E\left\{e_n^2\right\}\right]$ denotes the estimated gradient vector of the MSE w.r.t. $\mathbf{W}_n$.

One method for obtaining the estimated gradient of the mean square error function is to take the gradient of a single time sample of the squared error, i.e.,

$$\text{Grad.}_{est}\left[E\left\{e_n^2\right\}\right] = \text{Grad.}\left[\left\{e_n^2\right\}\right] = 2e_n\text{Grad.}\left[e_n\right] \tag{20}$$

From (10), $\text{Grad.}\left[e_n\right] = \text{Grad.}\left[\mathbf{W}_n^t\mathbf{X}_n - d_n\right] = \mathbf{X}_n$. Thus,

$$\text{Grad.}_{est}\left[E\left\{e_n^2\right\}\right] = 2e_n\,\mathbf{X}_n = 2\left[\mathbf{W}_n^t\,\mathbf{X}_n - d_n\right]\,\mathbf{X}_n \tag{21}$$

Substituting (21) into (19), we have the update equation of the weight vector for the LMS algorithm as follows:

$$\mathbf{W}_{n+1} = \mathbf{W}_n - 2\mu\,e_n\,\mathbf{X}_n = \mathbf{W}_n - 2\mu\left[\mathbf{W}_n^t\,\mathbf{X}_n - d_n\right]\mathbf{X}_n \tag{22}$$

Note that LMS algorithm requires the error signal e_n given in (10) to be generated. This means that we have to know the desired signal. The exact value for the desired signal is generally not available apriori. In practice, the condition of requiring the desired signal can be satisfied by generating an approximation of the actual desired signal. A signal, for example, consisting of the carrier frequency only can be used as a desired signal d_n in many applications of narrow band signal processings.[10] In some other applications such as system identification, the desired signal d_n is available and can be used as a reference signal during the adaptation. More details about the application of system identification is shown with computer simulations in section 16.4. The most attractive feature of the LMS algorithm is the simplicity of the procedure. Since the single time value of the squared error is directly used as an estimation of the average value of the squared error, we do not have to store the previous values of the signals for computing the average. This particularly means that the adaptive procedure is done for each row rather than for the whole matrix. In other words, no matrix needs to be stored and processed in the LMS algorithm.

Example 1.

Suppose a signal, $x_n = \sin(0.5\pi n) + \sin(0.6\pi n)$, is to be processed in an array consisting of N elements. The LMS algorithm is to be used for designing the array. The objective is to determine the value of the weight for each antenna element in such a way that the array output estimates the next value of the signal. Assume the sampling period and the spatial delay due to the antenna separation are the same.

Simulation results are shown in Table 16.1 with two typical values for the step size, μ. The adaptive procedure is terminated when the normalized error, which is defined as $|\mathbf{W}^t\mathbf{X}-y|/|y|$ becomes less than 10^{-5}. Figure 16.3 illustrates the relation between the magnitude of the step size and the required number of iterations for the convergence. Statistical analysis of the LMS algorithm that characterizes the mean and mean-squared behavior of the weight vector were given in [14]. It is shown that the mean behavior of the adaptive procedure converges to the optimum solution given in (14) if the step size μ is selected to be

$$0 < \mu < 1/\lambda_{\mathrm{MAX}} \tag{23}$$

In many practical applications, however, not only the mean behavior of the adaptive procedure but also the mean squared error of the weight vector must be convergent. When the input signals are Gaussian, following two conditions must be satisfied for the mean squared error to be convergent.[14]

$$0 < \mu < 1/2\lambda_{\mathrm{MAX}} \tag{24}$$

$$\sum_{i=1}^{N} \frac{\mu\lambda_i}{1-2\mu\lambda_i} < 1 \qquad \text{for } i = 1, 2 \ldots N \tag{25}$$

where λ_i denotes the eigenvalue of the autocorrelation matrix of the input signal, correspondingly. Note that the above conditions were obtained under the independence assumption[15] and the assumption that the input signals involved were zero-mean Gaussion. However, it will be still interesting to check the validity of (24) and (25) in this example. For this, we compute the autocorrelation matrix of $x(t)$ as follows:

$$\boldsymbol{\phi}(x,x) = \begin{bmatrix} r_0 & r_1 & \cdots & r_{N-1} \\ r_1 & r_0 & \cdots & r_{N-2} \\ \vdots & \vdots & \ddots & \vdots \\ r_{N-1} & r_{N-2} & \cdots & r_0 \end{bmatrix}$$

where r_i for $i = 0, 1, \ldots N-1$ is computed as

$$r_i = E\left[x_m\, x_{m-i}\right] = 0.5\left\{\cos(0.5\pi i) + \cos(0.6\pi i)\right\} \tag{26}$$

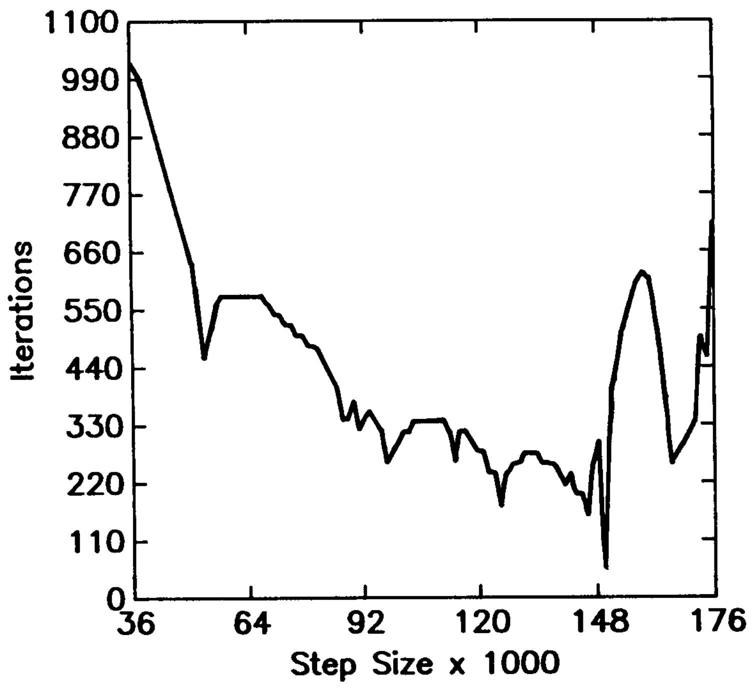


Figure 16.3 Step size vs. Required Iterations.

Note that the autocorrelation matrix can be fully determined from (26) because it is a Toeplitz matrix. From Table 16.2, we compute the eigenvalues of the matrix as shown in Table 16.3 when the array consists of 6 antenna elements. *

From Fig. 16.3, we can see that the step size μ must not exceed 0.176 for the mean square error to converge. Note that convergence conditions shown in (24) and (25) are more strict than the upper bound of μ shown in Fig. 16.3. The reason is that the derivation of (24) and (25) have been based upon the independence assumption of the

* The reason why we set the number of antenna elements to six will be discussed later in this example.

Step size	Norm. error	# of iter's	CPU time	W
0.01	$< 10^{-5}$	2757	2.11 sec	0.996, 1.391, 1.473, 1.550, -0.523, 0.158
0.1	$< 10^{-5}$	259	0.21 sec	0.996, 1.390, 1.473, 1.550, -0.522, 0.160

Table 16.1 Numerical Results of the LMS Algorithm.

$$\begin{bmatrix} 1.0000 & -0.1545 & -0.9045 & 0.4045 & 0.6545 & -0.5000 \\ -0.1545 & 1.0000 & -0.1545 & -0.9045 & 0.4045 & 0.6545 \\ -0.9045 & -0.1545 & 1.0000 & -0.1545 & -0.9045 & 0.4045 \\ 0.4045 & -0.9045 & -0.1545 & 1.0000 & -0.1545 & -0.9045 \\ 0.6545 & 0.4045 & -0.9045 & -0.1545 & 1.0000 & -0.1545 \\ -0.5000 & 0.6545 & 0.4045 & -0.9045 & -0.1545 & 1.0000 \end{bmatrix}$$

Table 16.2 Autocorrelation matrix $\phi(x, x)$ for 6 array elements.

0.0000	0.0000	0.1221	0.2797	2.4703	3.1279

Table 16.3 Eigenvalues of the autocorrelation* matrix.

Gaussian signals [15]. **

For determining the number of antenna elements, we have to consider the rank of the autocorrelation matrix of the input signal. In general, there exist $2N$ nontrivial eigenvalues in the autocorrelation matrix for an input signal consisting of N distinct frequency components. Recall that there were four nontrivial eigenvalues in the autocorrelation matrix of this example as shown in Table 16.3 and there were two distinct frequency components in the signal, $x(t)$. This particularly means that we need at least four antenna elements to estimate the

* IMSL routine has been used for computing eigenvalues of the matrix.

** The independence assumption states that $E\left\{\mathbf{X}_k, \mathbf{X}_1^t\right\} = 0$ for $k \neq 1$ where $\mathbf{X}_i$ denotes a column vector of the input signal formed at an observation time i and 0 denotes the null matrix. More details about the independence assumption is discussed in [15].

signal with a reasonable error, which has been set to 10^{-5}. Therefore, the solution for the weight vector becomes unique when the number of antenna elements is $2N$ (which is four in our case). If the number of antenna elements is greater than $2N$, then the solution is not unique and varies upon the initial choice for the weights, $\mathbf{W}_0$. The solution shown in Table 16.1 has been computed with initial choices being all 1's, i.e., $\mathbf{W}_0 = (1, 1, \ldots, 1)$. If the number of antenna elements is less than $2N$, then the mean square error of the weights vector cannot converge within a reasonable bound, e.g., 10^{-5}. Above discussions regarding the number of elements can be summarized as follows:

$$\begin{aligned}
&(1)\ \text{If } N_R = N && \text{then there is a unique solution,}\\
&(2)\ \text{If } N_R < N, && \text{then the solution is nonunique,}\\
&(3)\ \text{If } N_R > N, && \text{then the solution does not exist,}
\end{aligned}$$

where N and N_R denote the number of antenna elements and nontrivial eigenvalues of the autocorrelation matrix, respectively.

In practice, however, the rank of autocorrelation matrix is not available apriori and, furthermore, interfering components and thermal noise components are randomly included in the system environment. Because of these random components, the actual problem setting is in an infinite dimensional space, which physically means that an infinite number of antenna elements is required to solve the problem if the error is to be converged within a reasonable bound, e.g., 10^{-5}.

16.4 The Conjugate Gradient Method

The conjugate gradient method[2] was initially presented in electromagnetics[5] as an efficient way of solving a matrix equation of the form, $\mathbf{A}\ \mathbf{W} = \mathbf{Y}$[11]. In this section, the update equation for the weight vector of the conjugate gradient method is presented.

First, the method of steepest descent with variable step size is introduced. So far, the step size t_n in (8), has been treated as a constant, i.e., μ in (15), all through the iterations. In this section, we introduce a method of variable step size in which the step size of the adaptive algorithm is optimized at every iteration. As mentioned briefly at the beginning of Section 16.3, by finding the minimum of the properly selected functional, one can obtain an optimal value for the step size t_n at each iteration. Therefore, to use the method of variable step size, we have to first generate a proper functional, which is to be minimized with respect to the step size at each iteration. Once a functional is generated, then the objective becomes to find a value for t_n which yields the minimum of the functional at the next iteration, i.e., to find t_n such that

$$\partial f\left(\mathbf{W}_{n+1}\right)/\partial t_n = 0 \tag{27}$$

where $f(\mathbf{W})$ is a functional and $\partial/\partial t_n$ denotes the partial differential operator with respect to t_n.

16.4.1 The Conjugate Gradient Method

Let's first assume the matrix is real. Then, a functional can be properly generated such that the adaptive procedure of the conjugate gradient method is summarized very simply, i.e., simple expressions for the step size t_n and the search direction $\mathbf{P}_n$. In the later part of this section, i.e., 16.4.2, we introduce a generalized functional and adaptive procedure to eliminate the necessity of the assumption that the matrix is real.

Consider the following scalar quantity as a functional:

$$f(\mathbf{W}) = 0.5\langle \mathbf{A}\,\mathbf{W};\ \mathbf{W}\rangle - \langle \mathbf{Y};\ \mathbf{W}\rangle \tag{28}$$

where $\langle\,;\,\rangle$ denotes the Euclidean inner product. It can be found that minimizing the functional, $f(\mathbf{W})$ shown in (28), is equivalent to solving the matrix equation, $\mathbf{A}\,\mathbf{W} = \mathbf{Y}$ shown in (4), under the assumption that the matrix $\mathbf{A}$ is symmetric and positive semidefinite. To check the validity of the equivalence mentioned above, we compute the gradient of the functional, $f(\mathbf{W})$ shown in (28), with respect to $\mathbf{W}$ as

$$\text{Grad. } [f(\mathbf{W})] = \mathbf{A}\,\mathbf{W} - \mathbf{Y} \tag{29}$$

assuming that the real matrix $\mathbf{A}$ is symmetric. Equation (29) states that minimizing the functional $f(\mathbf{W})$ is itself solving the matrix equation, $\mathbf{A}\,\mathbf{W} = \mathbf{Y}$. Once again, the real matrix $\mathbf{A}$ must be symmetric (for (29) to be valid) and positive semidefinite (for the procedure to be a minimization procedure). Note that if the matrix $\mathbf{A}$ is negative semidefinite, then the procedure becomes a maximization problem.

Now, let us compute the step size t_n which minimizes the functional at each iteration, i.e., to find t_n such that

$$\begin{aligned}
&\frac{\partial f\,(\mathbf{W}_{n+1})}{\partial t_n} \\
&= \left.\frac{\partial\,[0.5\,\langle\,\mathbf{A}\,\mathbf{W}_{n+1};\mathbf{W}_{n+1}\,\rangle - \langle\,\mathbf{Y};\;\mathbf{W}_{n+1}\,\rangle]}{\partial t_n}\right|_{\mathbf{W}_{n+1}=\mathbf{W}_n+t_n\,\mathbf{P}_n} \\
&= (\partial/\partial t_n)\left[0.5\langle\mathbf{A}\,\mathbf{P}_n;\;\mathbf{P}_n\rangle t_n^2 + \langle\mathbf{R}_n;\;\mathbf{P}_n\rangle t_n\right. \\
&\qquad \left. + \,0.5\langle\mathbf{A}\mathbf{W}_n;\;\mathbf{W}_n\rangle - \langle\mathbf{Y};\;\mathbf{W}_n\rangle\right] \;=\; 0 \qquad (30)
\end{aligned}$$

where the residue vector $\mathbf{R}_n$ is defined as

$$\mathbf{R}_n = \mathbf{A}\,\mathbf{W}_n - \mathbf{Y}. \tag{31}$$

Note that (30) is valid only when the matrix $\mathbf{A}$ is symmetric because $\langle\mathbf{A}\,\mathbf{P}_n;\;\mathbf{W}_n\rangle$ is equal to $\langle\mathbf{A}\,\mathbf{W}_n;\;\mathbf{P}_n\,\rangle$ only when a real matrix $\mathbf{A}$ is symmetric. Since (30) is given in a quadratic form with respect to t_n, the coefficient of t_n^2, i.e., $0.5\,\langle\,\mathbf{A}\,\mathbf{P}_n;\;\mathbf{P}_n\,\rangle$, must be positive for the solution of (30) to be the minimum of the functional $f\,(\mathbf{W}_{n+1})$. Since the matrix has been assumed to be positive semidefinite, this condition can be satisfied. Therefore, the scalar quantity, t_n, at each iteration can be written from (30) as

$$t_n \;=\; -\frac{\langle\mathbf{R}_n;\;\mathbf{P}_n\rangle}{\langle\mathbf{A}\,\mathbf{P}_n;\;\mathbf{P}_n\rangle} \;=\; \frac{||\mathbf{R}_n||^2}{\langle\mathbf{A}\,\mathbf{P}_n;\;\mathbf{P}_n\rangle} \tag{32}$$

Now, let's derive the expression for the search direction $\mathbf{P}_n$. From (8), we first derive the expression for $\mathbf{R}_{n+1}$ by multiplying the matrix $\mathbf{A}$ on both sides of (8) and subtracting $\mathbf{Y}$ from the results of the multiplications. Then, we get

$$\mathbf{R}_{n+1} \;=\; \mathbf{R}_n + t_n\,\mathbf{A}\,\mathbf{P}_n. \tag{33}$$

The direction vector is initially set to be the negative of the initial residue $\mathbf{R}_0$, i.e., $\mathbf{P}_0 = -\mathbf{R}_0$. Then, starting from an initial guess $\mathbf{P}_0$, we refine the search direction as

$$\mathbf{P}_{n+1} \;=\; -\mathbf{R}_{n+1} + q_n\,\mathbf{P}_n. \tag{34}$$

The scalar quantity q_n is chosen for the conjugate gradient method such that $\mathbf{P}_i$ are $\mathbf{A}$-orthogonal with respect to the inner product as follows:

$$\langle\,\mathbf{P}_i;\;\mathbf{A}\,\mathbf{P}_j\,\rangle \;=\; \langle\,\mathbf{P}_j;\;\mathbf{A}\,\mathbf{P}_i\,\rangle \;=\; 0 \quad \text{for } i \neq j. \tag{35}$$

Applying the relation of (35) for $\mathbf{P}_{n+1}$ and $\mathbf{A}\,\mathbf{P}_n$ in (34), i.e., $\langle \mathbf{P}_{n+1}; \mathbf{A}\,\mathbf{P}_n \rangle = 0$, we can compute the value for q_n as follows:

$$q_n = \frac{\langle \mathbf{R}_{n+1}; \mathbf{A}\,\mathbf{P}_n \rangle}{\langle \mathbf{A}\,\mathbf{P}_n; \mathbf{P}_n \rangle} = \frac{\| \mathbf{R}_{n+1} \|^2}{\| \mathbf{R}_n \|^2} \tag{36}$$

When the search directions $\mathbf{P}_i$ satisfy the condition (35), it is guaranteed that the adaptive procedure converges in a finite number of steps.[11],[16]

Example 2.

The conjugate gradient method is used for designing the same array discussed in *Example 1*. The simulation results are shown in Table 16.2.1. In the simulation, to guarantee the positive semidefiniteness of the matrix, we square the signal matrix as follows:

$$\mathbf{A} = (\mathbf{A}')^t (\mathbf{A}') \tag{37}$$

where the signal matrix $\mathbf{A}'$ is defined as

$$\mathbf{A}' = \begin{bmatrix} x_1 & x_2 & \cdots & x_6 \\ x_2 & x_3 & \cdots & x_7 \\ \vdots & \vdots & \ddots & \vdots \\ x_{10} & x_{11} & \cdots & x_{15} \end{bmatrix} \tag{38}$$

Note that we do not compute the autocorrelation matrix of the input signal, $x_n = \sin(0.5\pi n) + \sin(0.6\pi n)$. This means that we do not store many input data to compute the time averages of them for forming the autocorrelation matrix and crosscorrelation vector. Instead, instantaneous values of the input signal is used to form the matrix, $\mathbf{A}'$. Compare the results shown in Table 16.2.1 to that of Table 16.2.2, which have been computed with the acceleration factor, q_n of (34), being zero. Note that when the acceleration factor q_n is zero the procedure becomes nothing but the method of steepest descent because if q_n is zero then the search direction would be simply the negative gradient of the functional. It has been our experience that the required number of iterations in the method of steepest descent, i.e., $q_n = 0$, becomes less as the number of rows of $\mathbf{A}'$, which is set to 10 in (38), increases. This means that the adaptive procedure for the method of

Norm. Error	#-iter's	CPU time	W
$< 10^{-5}$	6	0.05 sec	0.997, 1.392, 1.473, 1.550, -0.524, 0.158

Table 16.2.1 Numerical Results of the Conjugate Gradient Method.

Norm. Error	#-iter's	CPU time	W
$< 10^{-5}$	202	0.11 sec	0.996, 1.391, 1.473, 1.550, -0.523, 0.158

Table 16.2.2 Numerical Results of the Steepest Descent Method.

9.997605	−1.545082	−7.140579	2.898278	3.447232	−1.968863
−1.545082	6.193379	−0.398282	−4.045115	1.013947	1.516346
−7.140579	−0.398282	5.847889	−1.331557	3.486099	1.601732
2.898278	−4.045115	−1.331557	3.329220	0.207282	−1.918485
3.447232	1.013947	−3.486099	0.207282	2.770206	−0.986067
−1.968863	1.516346	1.601732	−1.918485	−0.986067	1.940121

Table 16.2.3 Elements Values for A.

−0.328	−0.067	0.957	0.045	−1.573	0.357

Table 16.2.4 Elements Values for Y.

steepest descent in which the acceleration factor q_n is zero can converge faster as we store the more number of input data for computing the matrix, **A**. In the conjugate gradient method, however, the required number of iterations for the convergence is always less than or equal to six, which is the number of unknowns, whether or not we store more than six rows of input data for forming the matrix **A**$'$. This will be discussed in more details in *Example 3.*

Example 3.

An adaptive echo canceller for a telephone communication channel is to be designed using the conjugate gradient method. The objective is to determine the values for the coefficients of an FIR filter in such a way that the impulse response of the FIR filter approximates the impulse response of the echo path connected to the telephone line shown in Fig. 16.4. Then, the reflected echo which can be thought as being the

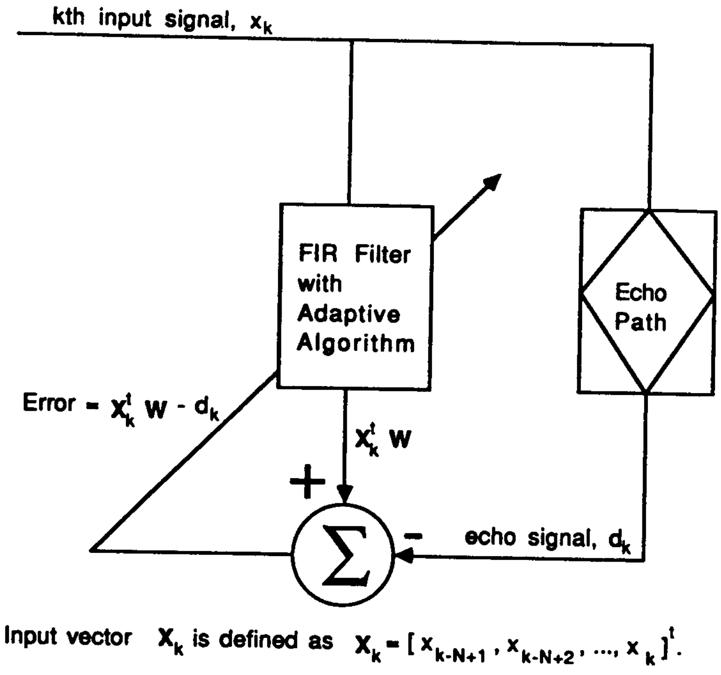

kth input signal, x_k
FIR Filter with Adaptive Algorithm
Echo Path
Error = $X_k^t W - d_k$
$X_k^t W$
+
-
echo signal, d_k
Σ
Input vector X_k is defined as $X_k = [x_{k-N+1}, x_{k-N+2}, \ldots, x_k]^t$.

Figure 16.4 Block diagram for example 3.

output of the echo path will be cancelled with the output of the FIR filter.

We assume the echo path can be modeled as a finite order linear system, i.e, an FIR filter. Suppose the system function of the echo path, $H\left(e^{i\Omega}\right)$, is modeled as an 8th order FIR filter as follows:

$$H\left(e^{i\Omega}\right) = \sum_{i=1}^{8} h_i z^{-i}\big|_{z=\exp(j\Omega)} \tag{39}$$

where the coefficients, h_i, are arbitrarily selected as

$$0.1,\ 0.4,\ -0.5,\ 0.3,\ -0.1,\ 0.7,\ -0.2,\ 0.1. \tag{40}$$

For simplicity, we set the number of unknowns, i.e., the order of FIR filter to be designed, as the same as the order of echo path, i.e., 8. Therefore, it is expected that the final solution for the filter coefficients converges, at the conclusion, to the values of h_i, *for* $i = 1, 2, \ldots 8$ shown in (40), correspondingly. If the order of the filter to be designed is higher than the actual order of the echo path, then the final solution for the redundant orders will converge to all zeros. Input training signal x_k shown in Fig. 16.5 is a zero mean uniformly distributed random signal which has been generated by a 66th order MLSR (Maximum Length Shift Register).

Let x_k denote the input signal with the observation time index k and $\mathbf{W}$ denote the vector of the filter coefficients to be determined, then the matrix equation can be formed as follows:

$$\begin{bmatrix} x_1 & x_2 & \ldots & x_8 \\ x_2 & x_3 & \ldots & x_9 \\ & & \ldots & \\ x_i & x_{i+1} & \ldots & x_{i+7} \\ & & \ldots & \\ & & & \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ \ldots \\ w_8 \\ \ldots \\ \end{bmatrix} = \begin{bmatrix} d_1 \\ d_2 \\ \ldots \\ d_i \\ \ldots \\ \end{bmatrix} \tag{41}$$

where d_i denotes the echo signal reflected from the telephone line. Note that the echo signal d_i can explicitly be calculated at any observation time i by taking a convolution of the input signal, $x_i, x_{i+1}, \ldots x_{i+7}$ with the 8th order FIR filter, h_i which have been assumed as (40). Recall that in most practical applications the desired signal, which corresponds to the echo signal in this particular example, was usually not available as discussed in 3.1.

To apply the conjugate gradient method for determining the values for $\mathbf{W}$, we have to first form a positive semidefinite matrix from the matrix shown in the left-hand-side of (41). Two different situations in forming a positive semidefinite matrix are considered as shown below.

< case #1 > When the averages for the autocorrelation and cross-correlation are computed.

Suppose the autocorrelation of the input training signal and the crosscorrelation between the input and echo signal are available. This situation can be represented as follows:

$$E\left\{\begin{bmatrix} x_1 & x_2 & \dots & x_8 \\ x_2 & x_3 & \dots & x_9 \\ & & \dots & \\ x_i & x_{i+1} & \dots & x_{i+7} \\ & & \dots & \\ & & & \end{bmatrix}^t \begin{bmatrix} x_1 & x_2 & \dots & x_8 \\ x_2 & x_3 & \dots & x_9 \\ & & \dots & \\ x_i & x_{i+1} & \dots & \\ & & \dots & \\ & & & \end{bmatrix}\right\} \begin{bmatrix} w_1 \\ w_2 \\ \dots \\ w_8 \end{bmatrix}$$

$$= E\left\{\begin{bmatrix} x_1 & x_2 & \dots & x_8 \\ x_2 & x_3 & \dots & x_9 \\ & & \dots & \\ x_i & x_{i+1} & \dots & x_{i+7} \\ & & \dots & \\ & & & \end{bmatrix}^t \begin{bmatrix} d_1 \\ d_2 \\ d_3 \\ \dots \\ d_i \\ \dots \end{bmatrix}\right\} \tag{42}$$

where E denotes the expectation operator. Equation (42) can be rewritten in a matrix form, $\mathbf{A}\,\mathbf{W} = \mathbf{Y}$ where $\mathbf{A}$ and $\mathbf{Y}$ denote the autocorrelation and crosscorrelation, respectively. $\mathbf{A}$ and $\mathbf{Y}$ can be computed by taking the time averages of the signal, i.e., by averaging enough number of blocks of 8 sampled signals. In the simulation, 3500 blocks are averaged as follows:

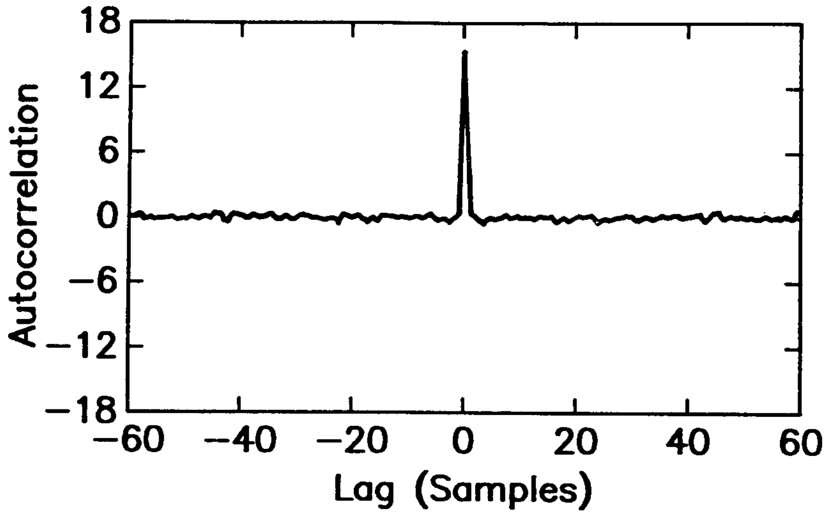
$$\mathbf{A}_k = (1/3500)\sum_{i=1}^{3500} s_i s_{i-k} \qquad \text{for } k = 0, 1, \dots 7 \tag{43}$$

$$\mathbf{Y}_k = (1/3500)\sum_{i=1}^{3500} d_i s_{i-k} \qquad \text{for } k = 0, 1, \dots 7 \tag{44}$$

Since the autocorrelation matrix is a Toeplitz matrix, we represent $\mathbf{A}$ with a single variable k in (43). The values for $\mathbf{A}_k$ for $k = 0$, $1, \dots 7$ and $\mathbf{Y}_k$ for $k = 0, 1, \dots 7$ are shown in Table 16.3.1. As shown in Table 16.3.2, the final solution for the filter coefficients, $\mathbf{W}_k$ for $k = 0, 1, \dots 7$, almost coincide with the values for the coefficients of the echo path. Note that the echo path has been modeled as an 8th order FIR filter with the coefficients defined as shown in (41). The method of steepest descent, in which the acceleration factor q_n of (34) is zero, results in similar values for the final solution for $\mathbf{W}$ as shown in Table 16.3.3. The procedures in both methods were terminated in 3 iterations with the normalized error being less than 10^{-5}. This means that the method of steepest descent can work as efficiently as the conjugate gradient method when we compute the autocorrelation matrix and crosscorrelation vector with enough number of input data. However, it is not realistic to assume the availability of that many input data (3500 blocks in our simulation). Even if the data are available, the statistics of the signals and system environments may change while we store the data and compute the averages of the data for the autocorrelation and crosscorrelation functions. If the statistics change then the final solution of the matrix equation is not optimal any more. Therefore, this application is valid only when (1) enough number of input data for computing the autocorrelation and crosscorrelation with reasonable accuracy are available, (2) the time for computing the averages is allowed, and (3) the statistics of the system is stationary (in wide sense). It is apparently not realistic to assume that above three conditions can be satisfied together. In the next application, we observe the difference of efficiency in both methods, i.e., the conjugate gradient and steepest descent methods, when the autocorrelation and crosscorrelation functions are not computed.

< case #2 > When the instantaneous values are used.

In this application, we take $2N-1$ samples for forming the matrix $\mathbf{A}$ where N denotes the order of the FIR filter to be designed. Thus, instead of taking the averages of many samples, we use the instantaneous values of the input training signals for the elements values of the matrix $\mathbf{A}$. Recall that we used 3500 blocks of samples for computing the averages in the previous application. Also note that $2N-1$ is the minimum number of samples to form an $N \times N$ matrix. The simulation results are shown in Table 16.3.4. The solutions for the weights $\mathbf{W}$ have been computed by the conjugate gradient method and the method of steepest descent in Table 16.3.4. and 16.3.5, respectively. What is interesting is that the acceleration factor q_n affects the convergence of the procedure very conspicuously in this case. In other words, when the instantaneous values of the input training signals rather than the time averages are used for the elements of the matrix $\mathbf{A}$ and vector $\mathbf{Y}$, the method of steepest descent converges much more slowly than the conjugate gradient method. Since, as discussed previously, the autocorrelation and crosscorrelation are generally not available in practice, the instantaneous values must be used to form the matrix $\mathbf{A}$ and vector $\mathbf{Y}$. Therefore, it is strongly recommended that the conju-



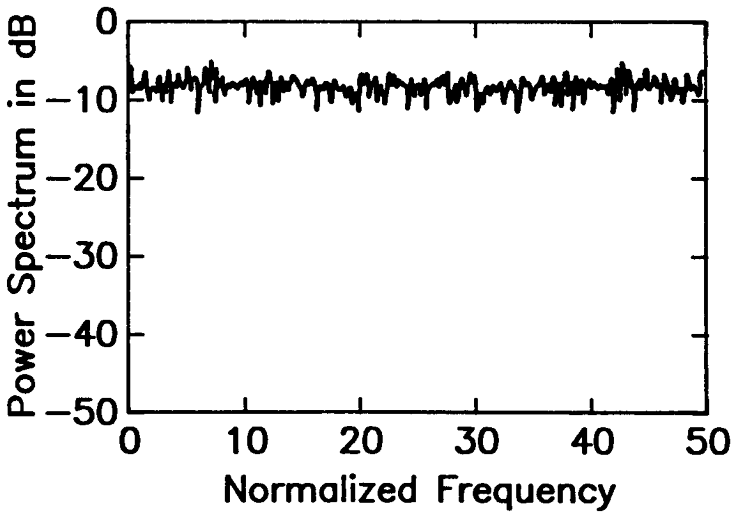


Figure 16.5 Input signal used in example 3.

k = 0	k = 1	k = 2	k = 3	k = 4	k = 5	k = 6	k = 7
0.154475	0.002768	0.001649	−0.003169	−0.000562	−0.001429	−0.000527	0.002409

Values for Autocorrelation Matrix (Toeplitz).

k = 0	k = 1	k = 2	k = 3	k = 4	k = 5	k = 6	k = 7
0.014022	0.061347	−0.077551	0.046811	−0.015469	0.109181	−0.030136	0.016941

Values for Crosscorrelation Matrix Vector.

Table 16.3.1 Autocorrelation and crosscorrelation for Example 3.

W_{actual}	0.1	0.4	−0.5	0.3	-0.1	0.7	-0.2	0.1
W_{comptd}	0.098971	0.400100	−0.500012	0.3000332	−0.099972	0.700002	−0.199974	0.100017

Table 16.3.2 Simulation results by the CG method with averaged data

W_{actual}	0.1	0.4	−0.5	0.3	-0.1	0.7	-0.2	0.1
W_{comptd}	0.098958	0.400094	−0.500012	0.3000323	−0.099965	0.700007	−0.199972	0.100028

Table 16.3.2 Simulation results by the CG method with averaged data

gate gradient method be used rather than the steepest descent method for efficiently solving the matrix equation which arises in a telephone communication problem due to the reflected echo. The normalized error, $\| \mathbf{A}\,\mathbf{W} - \mathbf{Y}\| / \|\mathbf{Y}\|$, is shown in Fig. 16.6 for both methods. Note that the error reduces very rapidly in the conjugate gradient method. In the simulation, the normalized error becomes less than 10^{-8} in 8 iterations which is the number of unknowns in the FIR filter.

From the simulations for case #1 and #2, we conclude that the conjugate gradient method works very efficiently in solving a matrix equation no matter whether the matrix has been formed with average values or instantaneous values. On the contrary, the steepest descent

Iterations	Norm. Error
1	7.59824e-2
2	2.50764e-3
3	5.03112e-4
4	6.37421e-4
5	1.47509e-4
6	1.09022e-4
7	2.15068e-4
8	2.95511e-10

Final Sol. for Coeff. Values
0.100000
0.400000
-0.499999
0.300000
-0.100001
0.700001
-0.200001
0.100000

Table 16.3.4 Simulation results by the CG Method with instantaneous data.

method, in which the acceleration factor q_n is zero, works much less efficiently than the conjugate gradient method when the average values are not available, i.e., when the instantaneous values are used as elements of the matrix $\mathbf{A}$ and vector $\mathbf{Y}$.

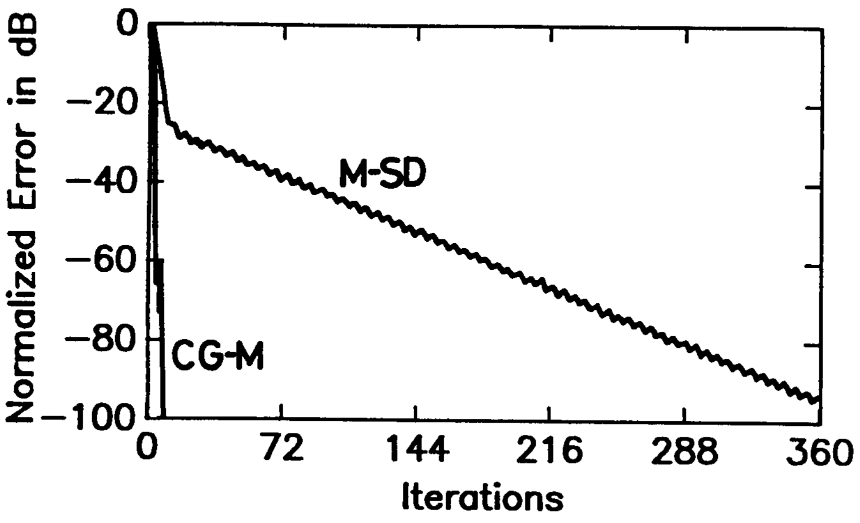
16.4.2 A Generalized Conjugate Gradient Method

In Section 16.4.2, for the procedure to be valid the real matrix should be symmetric and positive semidefinite, which forced us to square the matrix as shown in *Examples 2* and *3*. However, it is well known that if a matrix is squared, then the condition number of the matrix is also squared, which consequently worsens the convergence of the adaptive procedure. In this section, the adaptive procedure of the conjugate gradient method will be modified in such a way that the procedure can be used when the matrix is not real, positive semidefinite, or symmetric (Hermitian for the case of complex matrix). The conjugate gradient method will be further modified such that it can be used even when the complex matrix is not a square matrix.

For solving a complex matrix equation $\mathbf{A}\,\mathbf{W} = \mathbf{Y}$ in which the matrix $\mathbf{A}$ consists of complex elements, we introduce L^2 norm of the residue vector as a functional $f(\mathbf{W})$ as follows:

$$f(\mathbf{W}) = \langle \mathbf{R}_n ; \mathbf{R}_n \rangle = \|\mathbf{R}_n\|^2 \tag{45}$$

where the residue vector $\mathbf{R}_n$ is defined as $\mathbf{R}_n = \mathbf{A}\,\mathbf{W}_n - \mathbf{Y}$ and $\langle ; \rangle$ denotes the Euclidean inner product[16]. The conjugate gradient



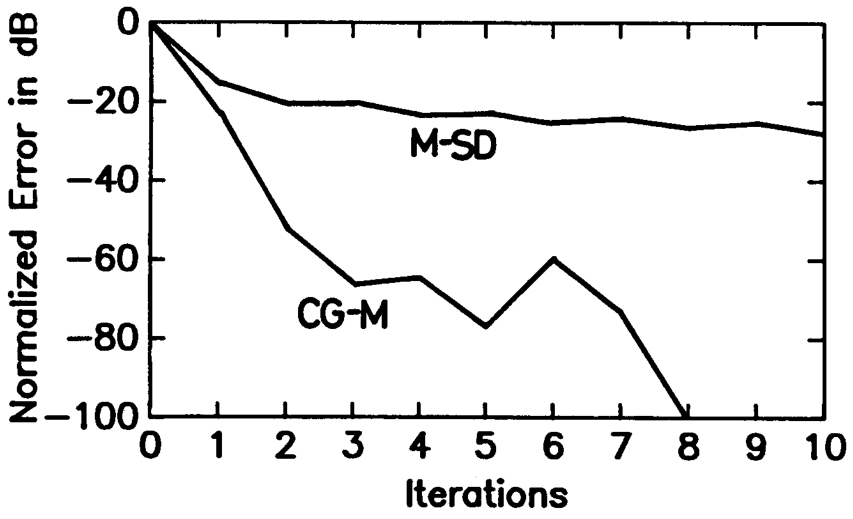


Figure 16.6 Normalized error vs. iterations.

Iterations	Norm. Error
1	7.59824e-2
2	7.59825e-3
5	4.87984e-4
37	9.93395e-5
143	9.66463e-6
247	9.81738e-7
351	9.97682e-8
457	9.71090e-9

Final Sol. for Coeff. Values
0.098832
0.399595
-0.498824
0.297811
-0.098094
0.696928
-0.199003
0.097400

Table 16.3.5 Simulation results by the SD Method with instantaneous data.

method minimizes $f(\mathbf{W})$ by generating the following sequence of solutions:

$$\mathbf{W}_{n+1} = \mathbf{W}_n + t_n\mathbf{P}_n \tag{46}$$
$$t_n = \|\mathbf{A}^H\mathbf{R}_n\|^2/\ \|\mathbf{A}\mathbf{P}_n\|^2 \tag{47}$$
$$\mathbf{R}_{n+1} = \mathbf{R}_n + t_n\,\mathbf{A}\,\mathbf{P}_n \tag{48}$$
$$\mathbf{P}_{n+1} = -\mathbf{A}^H\mathbf{R}_{n+1} + q_n\mathbf{P}_n \tag{49}$$
$$q_n = \|\mathbf{A}^H\mathbf{R}_{n+1}\|^2/\ \|\mathbf{A}^H\mathbf{R}_n\|^2 \tag{50}$$

where the superscript H denotes the Hermitian operator. Note that the matrix $\mathbf{A}$ does not have to be positive semidefinite for the functional of (45) to be minimized. Recall that, for the functional defined in (28) to be minimized during the adaptive procedure, the real matrix $\mathbf{A}$ had to be positive semidefinite.

Example 4.

An array is to be designed for a multipath telecommunication by utilizing the generalized conjugate gradient method. We consider a $(2N-1)$-element adaptive array of the structure shown in Fig. 16.1. Suppose that two multipath components are incident upon the array with incident angles ϕ_s and ϕ_m, respectively. Thermal noise component is also included at each antenna element of the array. The complex-valued signal induced at the kth element can be represented as follows:

$$x_k = s_k + m_k + n_k \qquad \text{for } k = 1, 2, \ldots 2N-1 \tag{51}$$

where s_k, m_k, and n_k denote the signal component, interference component, and thermal noise component, respectively, at *kth* element. Note that the required number of antenna elements for forming an $N \times N$ matrix is $2N-1$. The reason is that all the elements of the matrix, the number of which is $2N-1$, must be obtained at one observation time so that each element can be represented as a function of incident angles of the multipath components. Assuming the antenna elements are isotropic and a half wavelength apart, we can express s_k and m_k as follows:

$$s_k = s_0 \exp\left[-j\pi(k-1)\sin(\phi_s)\right] \tag{52}$$

$$m_k = m_0 \exp\left[-j\pi(k-1)\sin(\phi_m)\right] \tag{53}$$

for $k = 1, 2, \ldots N$ where s_0 and m_0 denote the signal component to be received and the interference component due to a multipath, respectively, induced at the reference antenna element. Then, the matrix equation can be established as $\mathbf{A}\,\mathbf{W} = \mathbf{Y}$ where

$$\mathbf{A} = s_0 \begin{bmatrix} 1 & \exp[-j\pi\sin(\phi_s)] & \cdots & \exp[-j\pi(N-1)\sin(\phi_s)] \\ \exp[-j\pi\sin(\phi_s)] & \exp[-j2\pi\sin(\phi_s)] & \cdots & \exp[-j\pi N\sin(\phi_s)] \\ \cdots\cdots & \cdots\cdots & & \cdots\cdots \\ \exp[-j\pi(N-1)\sin(\phi_s)] & \exp[-j\pi N\sin(\phi_s)] & \cdots & \exp[-j\pi(2N-2)\sin(\phi_s)] \end{bmatrix}$$

$$+ m_0 \left[\begin{matrix} 1 \\ \exp[-j\pi\sin(\phi_s)] \\ \cdots\cdots \\ \exp[-j\pi(N-1)\sin(\phi_s)] \end{matrix} \right.$$

$$\begin{array}{c} \exp[-j\pi\sin(\phi_s)] \\ \exp[-j2\pi\sin(\phi_s)] \\ \cdots\cdots \\ \exp[-j\pi N\sin(\phi_s)] \end{array}$$

$$\left.\begin{array}{cc} \ldots & \exp[-j\pi(N-1)\sin(\phi_s)] \\ \ldots & \exp[-j\pi N\sin(\phi_s)] \\ & \cdots\cdots \\ \ldots & \exp[-j\pi(2N-2)\sin(\phi_s)] \end{array}\right]$$

$$+\,\mathbf{n} \tag{54}$$

and

$$\mathbf{Y}^t = s_0\,[1 \quad \exp[-j\pi\sin(\phi_s)] \;\ldots\; \exp[-j\pi(N-1)\sin(\phi_s)]] \tag{55}$$

The noise matrix n will be described later in this example (57). Note that when there exist more than one multipath interference components we simply repeat the second term of the right-hand-side of (54) with different incident angles and references (m_0).

The matrix $\mathbf{A}$ shown in (54) consists of two multipath components, s and m, and the thermal noise n. In general, when there exist L multipath interferences, the element at the ith column of the jth row of the matrix $\mathbf{A}$ can be represented as

$$\mathbf{A}_{i,j} = s_{i+j-1} + \sum_{l=1}^{L} m^l_{i+j-1} + \underline{n} \qquad \begin{array}{rcl} \text{for } i & = & 1,2,\ldots N \\ \text{and } j & = & 1,2,\ldots N \end{array} \tag{56}$$

where the superscript l is used as an index for the corresponding interference component and a random quantity $\underline{\mathrm{n}}$ denotes the thermal noise component at each array element. In the simulation, the random component $\underline{\mathrm{n}}$ is generated as follows:

$$\underline{n} = (3)^{1/2}(S/SNR)^{1/2}\,\underline{a}\,\exp(-j2\pi\underline{\phi}) \tag{57}$$

where S is the average power of the signal to be received, $\underline{a}$ is a uniformly distributed random quantity in the interval [-1, 1], and $\underline{\phi}$ is another (independent) random quantity uniformly distributed between $-\pi$ and π.

The adaptive procedure of the generalized conjugate gradient method shown in (46)–(50) is applied to this problem. The simulation results are shown in Table 16.4 for the different values of signal-to-multipath interference ratio (SMR) and signal-to-noise ratio (SNR). For the simulations, we considered four interference components due to the multipaths of which the incident angles are, −48°, −24°, 30°, and 60°, respectively, while the desired signal is assumed to be incident along the direction of 0°. The values for (SMR,SNR) have been set to (0dB, 20dB), (20dB, 20dB), (0dB, 40dB), and (−20dB, 20dB) in Table 16.4, (a), (b), (c), and (d), respectively. The final solutions for the weights are shown in Table 16.4 together with the magnitude of the error at each iteration. The array pattern shown in Fig. 16.7 has been computed utilizing the following equation.

$$\text{Array pattern for incident angle } \phi = \sum_{i=1}^{N} \mathbf{W}_i \exp(-j\pi(i-1)\sin(\phi)) \qquad (58)$$

Iteration	Error
1	0.2
2	2.21e-2
3	2.91e-3
4	1.12e-4
5	1.86e-6
6	2.62e-37

[Re(W), Im(W)]
[0.15118195633698, −5.3331990647804e-2]
[0.21700717450047, −3.1222094826687e-2]
[0.24592750674078, −7.4339596236216e-5]
[0.21805807807816, 3.1042507446225e-2]
[0.15466003272798, 5.4484958246424e-2]

(a) SMR = 0dB, SNR = 20dB

Iteration	Error
1	0.2
2	1.76e-4
3	1.81e-5
4	3.93e-6
5	6.29e-8
6	5.98e-21

[Re(W), Im(W)]
[0.13642150675143, −4.9501015851098e-2]
[0.22244807808097, −4.6627397971111e-2]
[0.23422790312294, 1.7345718303866e-3]
[0.22840605633478, 4.0157297779941e-2]
[0.16553643540934, 5.4653081862584e-2]

(b) SMR = 20dB, SNR = 20dB

Iteration	Error
1	0.2
2	2.35e-2
3	3.21e-3
4	8.42e-5
5	1.53e-6
6	1.56e-37

[Re(W), Im(W)]
[0.15496157953842, −5.4723919105755e-2]
[0.21907776944136, −3.0211840535102e-2]
[0.25011709167830, −1.2106797072577e-5]
[0.21919013363437, 3.0196826031389e-2]
[0.15532117843457, 5.4846483158216e-2]

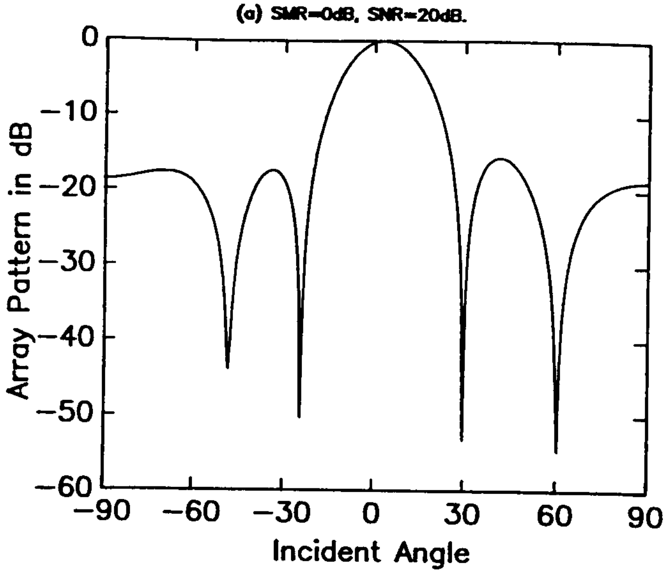
(c) SMR = 0dB, SNR = 40dB

Iteration	Error
1	0.2
2	0.18
3	0.17
4	0.11
5	4.42e-3
6	8.34e-34

[Re(W), Im(W)]
[0.15317135902711, −5.3945528178160e-2]
[0.21650728478971, −2.9670112625984e-2]
[0.24715349593322, 1.9782253228295e-4]
[0.21656764977023, 3.0022873479564e-2]
[0.15343479480437, 5.4326889765988e-2]

(d) SMR = −20dB, SNR = 20dB

In Fig. 16.7, array patterns are illustrated as a function of incident angle in the interval [−90°, 90°]. Note that there exist pattern nulls along the incident angles of the multipath interferences. As can be predicted, the pattern nulls are generated more deeply as SNR becomes larger. This is shown in Fig. 16.7(c). What is interesting is that the pattern nulls become deeper as the value for SMR become smaller as shown in Fig. 16.7(a), (b), and (d). This means that the array can generates pattern nulls which become deeper as the interferences become larger. Note that in Table 16.4 and Fig. 16.7(d) the pattern nulls are deep enough to resolve the interferences even when the magnitude of the interferences are 20dB larger than the signal to be detected.



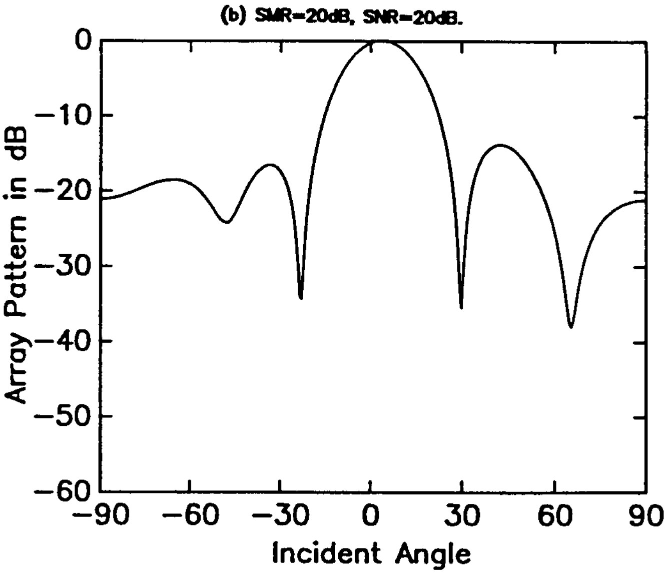
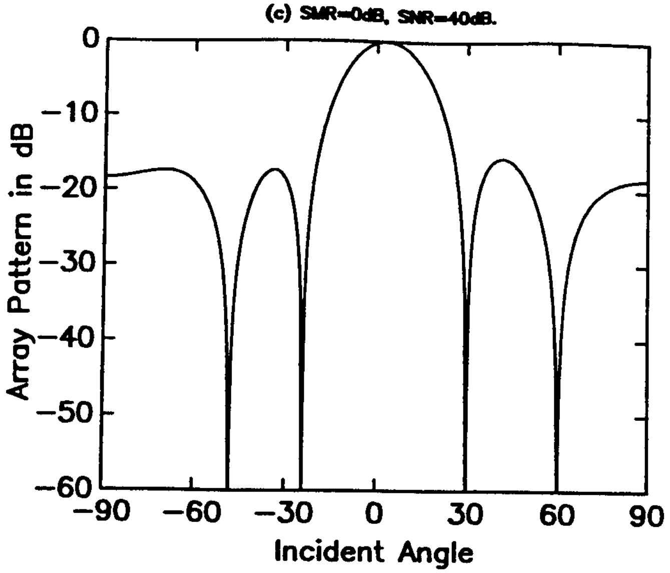


Figure 16.7 Continued.



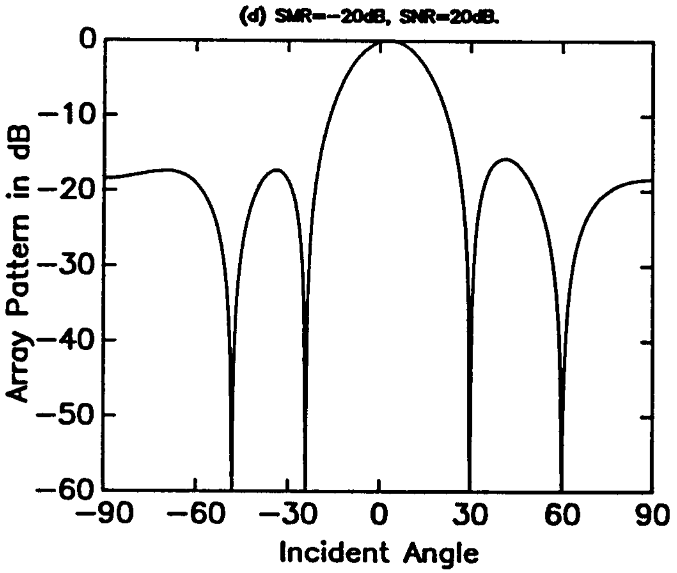


Figure 16.7 Array pattern by the optimal weights. Incident angles; $M1 = -48°$, $M2 = -24°$, $M3 = 30°$, and $M4 = 60°$.

16.5 Summary

The conjugate gradient method is presented as an efficient method for solving an ill-conditioned matrix equation which arises in communication problems and electromagnetic wave interactions. Comparisons of the conjugate gradient method to the other conventional adaptive algorithms are also presented showing the strong and weak points of each method. The conjugate gradient method is generalized and simplified to treat an arbitrary complex matrix equation.

The following two features of the conjugate gradient method become evident: (1) The convergence of the adaptive procedure is guaranteed by setting a proper functional which monotonically decreases at each iteration. This means that, no matter what kind of matrix equation we are facing, the conjugate gradient method can solve it in a finite number of steps for any initial guesses. (2) The conjugate gradient method converges faster than any other iterative method. Since we cannot use any direct matrix solution techniques when the matrix is very ill-conditioned, one can say that the conjugate gradient method solves the matrix equation arising in practical situations in the most efficient way.

What we have to pay for the above two features is an added complexity of the procedure which will result in additional hardware of the system. However, the speed-up gained by the efficiency of the conjugate gradient method compensates for the added complexity in the hardware.

Acknowledgments

The author thanks Dr. Tapan K. Sarkar for his continuous advice and helpful discussions during the research. The appreciation is extended to Mr. Young-Wha Kim and Dr. Sung Ho Cho for their extending valuable assits. The careful review by Dr. Hideichi Sasaoka is also gratefully acknowledged.

References

[1] Monzingo, R. A. and T. W. Miller, "Introduction to Adaptive Arrays," John Wiley & Sons 1980.

[2] Hestines, M. "Conjugate Direction Methods in Optimization," Springer-Verlag, 1980.

[3] Choi, S. "Design and Application of Adaptive Tseng Window for One & Two Dimensional Radio Direction Finding and Digital Signal Processing," *Ph.D. Dissertation, Dept. of ECE, Syracuse University, Syracuse, NY.*

[4] Widrow, B., P. E. Mantey, L. J. Griffiths, and B. B. Goode, "Adaptive Antenna Systems," *Proc. IEEE* **55**, 2143–2159, Dec. 1967.

[5] Sarkar, T. K., K. Siarkiewicz, and R. F. Stratton, "Survey of Numerical Methods for Solving of Large Systems of Linear Equations for Electromagnetic Field Problems," *IEEE Trans. on Antennas and Propagations,* **AP–29**, Nov. 1981.

[6] Hestenes, M. R. "Applications of the Theory of Quadratic Forms in Hilbert Space to the Calculus of Variations," *Pacific J. Math.,* **1**, 525–581, 1951.

[7] Allen, J. L. *The Theory of Array Antennas (with Emphasis on Radar Applications),* M.I.T. Lincoln Lab., Lexington, Mass., Tech. Rep. 323, Memo DDC AD-422945, Jul. 1963.

[8] Dudgeon, D. E., "Fundamentals of Digital Array Processing," *Proc. IEEE,* **65**, 898–903, Jun. 1977.

[9] Elliott, R. S. *Antenna Theory and Design,* Prentice- Hall, Englewood Cliffs, NJ, 1981.

[10] Ogawa, Y., M. Ohmiya, and K. Itoh, "An LMS Adaptive Array for Multipath Fading Reduction", *IEEE Trans. on Aer. & Elec. Sys.,* **AES-23**, Jan. 1987.

[11] Sarkar, T. K., X. Yang, and E. Arvas, "A Limited Survey of Various Conjugate Gradient Methods for Solving Complex Matrix Equations Arising in Electromagnetic Wave Interactions," *Wave Motion,* 527–546, Elsevier Science Publishers B. V. (North Holland), 1988.

[12] Widrow, B., and S. D. Stearns, *Adaptive Signal Processing,*

Prentice-Hall Inc.,Englewood Cliffs, NJ, 1985.

[13] Sarkar, T. K., and N. Sangruji, "An Adaptive Nulling System for a Narrow-Band Signal with a Look-Direction Constraint Utilizing the Conjugate Gradient Method," *IEEE Trans. on Antennas & Propagations,* **37**, 940–944, 1989.

[14] Feuer, A., and E. Weinstein, "Convergence Analysis of LMS Filters with Uncorrelated Gaussian Data," *IEEE Trans. on ASSP,* **ASSP-33**, 222–229, Feb. 1985.

[15] Cho, S. H., "Convergence Analysis for Efficient Adaptive Digital Filtering Algorithms and Structures," *Ph.D. Dissertation,* The University of Utah, 1989.

[16] Anton, H., *Elementary Linear Algebra,* fourth edition, John Wiley & Sons, NY, 1984.

[17] Orfanidis, S. J., *Optimum Signal Processing: An Introduction,* Macmillan Publishing Company, 1985.

[18] Chen, H., T. K. Sarkar, S. A. Dianat, and J. D. Brule, "Adaptive Spectral Estimation by the Conjugate Gradient Method," *IEEE Trans. on ASSP,* **ASSP-34**, 272–284, Apr. 1986.